Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 7-16 E-ISSN 2587-3636

CZU: 372.8004 DOI: 10.36120/2587-3636.v18i4.7-16

TEACHING COMPUTER PROGRAMMING IN SCHOOLS

Krassimir MANEV*, professor, doctor

New Bulgarian University, Sofia, Bulgaria

*Lecturer in Discrete Mathematics, Programming and Algorithms in New Bulgarian University, Sofia, Bulgaria; co-founder of International Olympiad in Informatics (IOI); Team Leader of Bulgarian team; 11 years member of International Committee of IOI (including President of IOI '2009) and President of IOI for the period 2014-2017; in 2017 he co-founded European Junior Olympiad in Informatics and is its first President.

Summary. Starting with a brief historical incursion of the introduction of computer science into the school from the author's perspective, the article outlines some theoretical aspects regarding the modern trends in the formation and development of programming skills of students and their involvement in computer competitions.

Key words: computer programming, competitions in Programming, teaching Informatics in secondary school, task.

STUDIEREA PROGRAMĂRII ÎN ȘCOLI

Rezumat. Începând cu o scurtă incursiune istorică a introducerii informaticii în școală din perspectiva autorului, în articol se elucidează unele aspecte teoretice cu privire la tendințele moderne în formarea și dezvoltarea competențelor de programare la elevi și implicarea acestora în concursurile de informatică.

Cuvinte cheie: programare, concursuri de programare, predarea informaticii în școli, sarcină.

Introduction

I have started to learn computer programming more than 50 years ago – during the far 1966, in the National mathematical school in Sofia. My first computer was "Minsk 2" – a machine with 2K 36 bits words memory, practically without input/output devices – only punched tape reader and writer, and primitive printer that was able to print a single float point number on a line. There was no OS, no IDE and even no compiler from some used in that time language. Something more – there was no even Assembly language and Assembler. We created our first programs in the machine language, punched them on tape and operators executed them manually from the control console of the computer and brought us a short piece of paper with one-two numbers that was the result of our programs. Usually that was solution of some equation.

In my first years at the Sofia University we started to learn FORTRAN and PL/1 but without the possibility to execute our programs at all. Meanwhile the "modern" machine "Minsk 32" was installed in the Institute of Mathematics of the Bulgarian Academy of Sciences and we had the possibility to write our diploma thesis in Assembler. This machine had some simple OS functions. The programs were punched on much more comfortable 80 columns card and results were printed on 128 columns printers. The

machine had 8 magnetic tapes reading/writing devices and we could save our programs to tapes and execute them many times from tape without reading the cards and reassembling for each execution. One researcher of the Institute created in that time, not so complex, plotter and I graduated with a diploma thesis on a system that could, by given from the cloth constructor cuts of man shirts, to create ant draw on the plotter any asked by the user size of a shirt cut.

When I have started my academic career as an assistant professor at the university the things changed totally. First the Eastern Europe's IBM 360 compatible machines – so called "ES series" – were installed everywhere, with a true OS, disk devices, Assembler and compilers from FORTRAN and PL/1. Teaching of Assembly language of IBM 360 (the best I have ever seen) goes in parallel with teaching of the procedural languages. Anyway the cycle of compiling a program, finding the obtained bugs of compilation or errors of the logic, punching replacing cards and submitting the deck of cards for new compilation/execution lasted usually one day. When a bit later the terminals were attached to our ES 1040 with a corresponding system for distant creation (what revolution was creating and editing the code on the screen of the terminal) and execution of the programs started to looks, more or less, like the nowadays technology of programming.

At the end of 70's and the beginning the 80's years of the past century computer programming entered the secondary schools of Bulgaria – exceptionally in the profiled mathematical school. Methodology of teaching (if something as methodology existed at that moment) was directly moved from the universities to the school. And university professors participated in this process en mass because there were no teachers in programming at all. Even in this early moment it was clear that secondary school students with mathematical abilities very quickly perceive the conceptions of the computer programming and very quickly get significant skills in solving tasks with computer programs. It was clear also that, for these talented students, the very limited school program in Informatics is not a challenge at all and we had to support their ambitions for development in the domain in some other way.

At the end of 70's, almost the same time when in USA started the Collegiate Programming Contest for university students (famous nowadays all over the world as ICPC) we started the first competitions in Programming for secondary school students. Organizing programming contests on ES (IBM 360) machines was very hard. I could write a separate paper about difficulties we have to overcome. But here I will limit myself to a simple example – at that time we had to evaluate programs of the contestants reading the source code. The time of 4 hours was too short at that times, the contestants did not succeed to finish their programs and checking them by test cases was impossible.

The things changed revolutionary with appearing of the personal computer. Bulgaria started producing own Apple II compatible personal computer Pravetz 8 and

any secondary school was able to install as many of these machines as the school needs. Any school student with corresponding interest could start to learn programming. In the middle of 80's we created the National Olympiad in Programming for school students. Each contestant used its own machine and could create and debug in depth her/his program. In 1987, by an idea of prof. Blagovest Sendov, we organized the first international contest in programming with participation of 7 countries, and with two age groups. Enthusiasm from the results was enormous and prof. Sendov asked from UNESCO approval to organize the first International Olympiad in Informatics (IOI) in Bulgaria. This year IOI has its 31st edition in Baku, Azerbaijan with participation of 327 students from 87 countries.

Almost 40 years of my life are closely related with the competition on programming. I started as a task creator and coach, was President of the National team for Olympiads in Informatics, Leader and Deputy Leader of the National team for many IOI and Balkan Olympiads in Informatics. I have served 11 years as a member of IC of IOI and from 2014 to 2017 was a President of IOI. After the end of my presidency I have created with a team of colleagues the first international programming contest for school students aged under 15,5 years – the European Junior Olympiad in Informatics (EJOI) [5]. The 3rd edition of EJOI took place this year in Maribor, Slovenia, with 90 contestants from 24 European countries.

During all this years I did not stop my work as University professor, mainly in Sofia University, but also in American University in Bulgaria, New Bulgarian University and other Bulgarian universities too, teaching Discrete Mathematics, Programming and Algorithms. As University professor I was coaching long years the team of Sofia University for Regional ICPC contest and 4 times participated in ICPC World finals. My participation in the International programming contests give me the opportunity to meet colleagues from many countries of the world and to share opinions with them on teaching programming and related topics.

During my academic career I have published more than 75 scientific and methodological papers; I have led a team that created more than 25 textbooks for Bulgarian school in Informatics and Information Technologies and published 2 textbooks for university students – in Discrete mathematics and in Algorithms in graphs.

I decided to start this paper with such extensive description of my academic career because, I suppose, I know almost all history of teaching programming in my country and the big part of the history of teaching programming in the world. This is crucial because in the next lines I would like to stress some negative trends in the process of preparing programmers – an activity that I consider extremely important for the future of the Humanity!

Teaching programming in the secondary schools and the universities

Teaching Informatics came in Secondary schools relatively later – in 70's – 80's years of past century. It came in the moment when Mathematics and the other natural sciences have their strong place in the school curriculum. And it happens that there is no room for teaching Informatics. That is why teaching the elementary things in Informatics – what is computer, what is OS and program, and how to create programs – is taught in out of class or in some elective forms. More serious teaching of Informatics we have only in some profiled schools (so called mathematical schools) but not in all. With wide spread of computers and computerized devices another discipline – Information technologies – came in the curricula and there is a tendency to almost totally displace teaching of Informatics.

For example, let see nowadays the curriculum of the Bulgarian school. We have elective classes (one hour per week) in Information technologies for 1^{st}-4^{th} degrees of schools that have the possibilities to organize such education. Then we have mandatory classes (one hour in week) in Information technologies for (5^{th}-10^{th} degrees). And mandatory classes (two hours in week) in Informatics for 8^{th} degrees only and not in all schools! More Informatics is taught in mathematical schools when there are well prepared teachers. On this misbalanced background Bulgarian ministry in education introduced in 3^{rd}-4^{th} degrees new (third) discipline with the pretentious name Computer modeling! The idea was in this discipline classes to be introduced some programming environment for children (Skratch-like) but we strongly doubt that it happens.

This colorful and misbalanced process is implemented under the conditions of permanent lack of qualified teacher – not only in Informatics but in Information technologies and new discipline Computer modeling, too. The lack of teachers in Informatics and related disciplines is easy explainable when we have in mind that the software industry "sucks" almost each qualified young professional proposing 5-10 time higher salaries. That is why some of the teachers are not prepared to teach the material in assigned classes, especially the teachers in primary schools who have to taught many disciplines. So, "education" pass under the slogan "computer is yours, make what you wish".

Classes in Information technology are too much, and in these classes, in a "spiral", are repeated exercises with the applications from MS Office. Some of the students know well these programs and the other have no interest and teaching loses sense. Classes in Informatics are not enough even for introducing the main conceptions of the domain. If some programming is included in the Curriculum it is dedicated to creating very small applications with graphical interface, which is rather technology (creating screen forms and linking some standard function events of the form) but not real programming. And especially when the teachers are not prepared enough they leave students to play games and to surf in the net. What happens in the classes of Computer modeling is full mystery.

We have information that some teachers are reading fairy tales in these classes (this is 3rd-4th grade any way and there is not too much to expect).

In other papers, discussing the situation, we proposed a reasonable solution of this not normal state of the art. We would like to repeat it here. The school needs only one discipline – Informatics and Information technology – but from the first to last grade and with the existing number of class hours of the three disciplines. In primary school the student have to be taught the main skills for working with the computer – for example, correct work with the keyboard (10-fingers typing) and the mouse, principles of the OS (executing programs, works with the file systems) and some introductory abilities to use application (typing and editing of text, for example).

In middle grades (5th-7th) students have to be taught, as in the moment, to more deep usage of the applications (formatting big documents, structuring data in tables, presenting data, and drawing with a graphic editor). But in parallel some lessons in Informatics have to be introduced in these years, increasing each year the number of class hours. In 8th-12th degrees of the regular schools the main part of the time has to be dedicated to topics from Informatics – carefully chosen parts of Discrete Mathematics (not such a thing in the secondary school at all), Operating Systems (role and functions of OS are hidden under the graphic interface and students do not understand well them), Programming and Algorithms, Networking. Lessons in programming have to repeat the ontogenesis of the discipline. The students have to be able to write procedural code, before to pass to Object-Oriented programming and making applications with graphical interface and network applications. Some technology classes in this period have to be included too – on Data Bases and Information systems. In profiled mathematical school the number of classes could be 2-3 times bigger and teaching could be more intensive and comprehensive [4].

Only with such ultimate discipline in the secondary school it is possible to expect that the students accepted in university programs will be able to study successfully and to graduate in time. And most of the university programs in Informatics and IT did not consider this particularity of the day – now the students following programs in Informatics and IT are much more than in time when the corresponding Curricula are created. And the average preparation level of the students that come from the Secondary schools is below the level in that time.

Students have not mathematical culture (they just know to solve some typical tasks of the school mathematics but not to reproduce proof of a theorem or to construct their own proof), so they do not understand the lectures in Discrete Mathematics [1]. As a result they are not able to understand and use complex abstract data type and to structure data. As well as they are not able to understand more complex algorithms (because each complex algorithm is based on some more or less deep theorem). So the universities produced mainly Web programmers and programmers of the routine programs with

graphical interface. The same are doing many “academies” created by big software companies of societies of professionals.

The university could not fill the “whole” in the education in Informatics of secondary schools created during of years neglecting this education in the school. To tear down significantly the level of the university programs means to turn the universities in professional schools. No one university will do such suicide. Because the constant increasing of necessity of programmers, the solution is to mobilize the efforts of the community for reforming the education in programming and related topics in the secondary school. And the universities could make some compromise with the level of education till the moment when the gap between the school and university is minimized enough. Another way to fill this gap is creating an intermediate level of education, which to prepare students for the university, or to work in a software business as developers of low qualification (the business need such developers).

Programming contests for secondary school students

At the end, I would like to share some things that happen in the programming contests for secondary school students – mainly in IOI but in all related competitions where contestants are prepared for IOI – National Olympiads and Regional Contests too [2]. Because these contests are the place where the young programmers fund the base of their future career in programming, they could help the contestants to elaborate skills and habits that could be very helpful in their future work – to plan carefully the implementation, to analyze in details the necessary algorithm and the most efficient structuring of the data, to structure and type the code in such way to be easy readable, to debug efficiently the code and so on. But these contests could also create habits that are not so helpful and even harmful. Some of these negative habits that the nowadays contest in IOI style nor simply encourage but make them inevitable I would like to comment here.

The programming contest task usually states some algorithmic problem. The statement of the task describes in details what kind of data will the program receive on the input, how this data will be formatted, what kind of results it has to produce and how the results have to be formatted on the output. In order to solve the task the contestant has to write corresponding piece of programming code and to submit it to the grading system of the contest. The grading system performs the task – append some piece of code from the author, compile the obtained program and execute it against a set of test cases. Efficiency of the used algorithm is estimated by corresponding time limit for each execution. The contestant receives grading marks depending of the quantity and quality of the test cases that the program solved.

The first trend that I would like to comment is so called *feedback*. When the contestant submit a solution then the grading system could evaluate the solution without

telling to the contestant the result of the evaluation – such grading is called *without feedback.* But it is possible that grading system reports to the contestant the result from evaluation on a part of the test cases – *partial feedback* or on all test cases – *full feedback*. Giving partial or full feedback was introduced in IOI and related contests less than 10 years ago because the tasks became very difficult and the assumption is that with some feedback contestants will have more chances to solve such difficult tasks.

We could accept such an assumption and consider it reasonable. But, unfortunately, some colleagues decided that it is necessary to grade each task with full feedback. I would not like to discuss the fact that, having full feedback, some contestants started to make attempts to guess answers of the test cases instead to write a program that solve them. And responsible committees had to search how to prevent such possibilities. Much more dangerous is the behavior of almost all contestants to stop debugging their programs. Having full feedback the contestant just submits the code without debugging it. If the submit is successful the contestant moves to next task. But if the full feedback says that the solution is wrong then the contestant makes some changes and without debugging the code submits it again. And so on till the limited number of submissions.

As it was mentioned above debugging of the code is one of the main activities of the developers – debugging needs much more time than writing the code. And when one is developing commercial software there is no feedback at all. Giving full feedback in the competition of programming denude contestant of the possibility to train one of the most important professional activity in their future career and could be very big damage. Reasonable compromise in contest round with 3 tasks is simple – one task with full feedback, one task with partial feedback, and one task without feedback at all.

The second trend that I would like to discuss is connected with the form of supplying the input data of the competition tasks. Because of the constant increasing of the speed of modern processors authors of the tasks started to increase the size of the input (and/or output) data in order to be able to estimate more precise the speed of the used algorithm. As a result the time for execution of the solutions and giving the promised full feedback starts to increase to and this overloaded the grading system. The proposed solution of the problem was that instead of submitting complete program solution contestant had to submit, **for all tasks**, some functions to be called by the main function of the author. The necessary data respectively structured, are provided through a pointer argument and the results are passed back to the main function through some pointer too.

I would not comment the mass of technical difficulties that such approach creates. A negative trend in the preparation of young programmers in this case is that the contestant stopped to parse input data. We could not deny that interface of nowadays commercial software is organized through the screen forms where parsing the input data is almost not necessary. But ability of parsing text input is important for the qualified

developers because of existence of applications in which parsing text input is inevitable and will be such forever. The compilers are very good examples for such applications.

The normal solution is obvious in this case – for tasks with extremely large input/output will be natural the statement of the task to demand creation of sub-function. But for task in which input is not so large it is normal to demand that the program is able to parse the input.

The last negative trend that we would like to stress is splitting of competitive tasks in *subtasks*. The idea is that not so experienced contestant to be able to solve some of the easy subtasks that correspond to their level of preparation. We still not deny this idea in principle. Before the era of subtasks the authors gave such possibilities to contestants making the test case in such way that that solution of complexity $O(n^3)$ to earn 10% of the points, a solution of complexity $O(n^2)$ to earn 20% of the points, a solution of complexity $O(n\log n)$ to earn 30% of the points and a solution of complexity $O(n)$ to earn 100% of the points. In that time the contestant was writing a single program trying to implement the best algorithm they could invent and if there is time then to search other, more efficient.

What happen now? For some tasks 5, 6 or more subtasks are formulated, some of them so different that it is impossible to solve one of subtasks ameliorating the algorithm that solved another subtask. Something more, to receive a grade for some subtasks it is necessary to solve all test cases for this task. It is quite possible that the contestant has a single unsolved test case in each of the tasks and instead of usual 80% of the points to obtain zero. And more, sometime authors create complex relations among the tasks – even if the contestant solved all test cases for subtask *D* she/he could not obtain a mark if was not solved subtask *C,* but grades for correct solving subtask *C* are assigned if and only if the subtask *B* or subtasks *A* is solved. In such a way the normal solving a competitive task by searching better universal algorithm is replaced by considering a set of not coherent algorithms and difficult navigating through the relations of the subtasks.

We consider such trend very negative because it is not teaching young programmers to some abilities that will be necessary in their professional career. In this format of the tasks sometime tasks appear which have no natural splitting of subtask. Even interesting such task is rejected or is split formally in some subtasks just for the principle that task has to be in subtasks. In this format IOI tasks are, the same are the tasks of ICPC and related contests. But for comparison, one ICPC-style contest tasks set is usually composed of 12 tasks that has to be solved by a team of 3 university students for 5 hours. In IOI-style contests a single secondary school student is put in front of 15 or more subtasks (sometime with inner dependencies among them) that she/he has to solve for the same 5 hours.

The full feedback leads to very negative behavior of some contestants. Without attempting to make a perfect solution of some of the tasks, which has to be the main goal

of programming contest in educational prospective, they submit 6 routine solutions of the contest tasks and won bronze medals. In IOI'2019 a bronze medal was earned with 41,7% of the points, in IOI'2018 – with 31,17% and in IOI'2017 – with 22,86%. It is clear that the smartest children in our domain will learn how to survive in the complex competition conditions. But the question is "Why they need to do it?"

One of the most important features of the nowadays programming contests is the using of grading systems. Contemporary grading systems are high quality instrument for automatic testing of programs. They are not created to be used in education in programming and have not some functions that are important for teaching – to show some of the tests cases, to show answer of some of the test case, to provide teaching material concerned solved task etc. Extending a classic grading system with such functionality will make it priceless assistant of the teachers, making the educational process much more intensive and efficient [3].

In addition to programming contest in "Olympic" style, we have to mention another kind of contest – in them the contestants present software projects made "in home". We consider these contests also very important, because not each student has the psychological quality of a contestant – high mathematical culture, deep knowledge of Algorithmics and libraries of predefined abstract data types, ability to work under time pressure and so on. There are students who are very good in working on a large project that needs months for implementing. These students are very big reserve of future programmers and their efforts have to be encouraged, too.

But let us see what happen in this very large domain of education of young programmers. Instead of directing their efforts to the important topics and instruments of Informatics and creating applications, many tutors push their talented pupils to create presentations, visualizations, games interfaces, static web sites or web sites with trivial access to simple data base and so on. Something more, the tutors and members of the Juries of such competitions of projects pretend that the results of these students are of high quality and importance and deviate in such way some of the students from the development of the serious professional skills in programming.

Conclusions

Education of young programmers is one of the most important tasks of the global education process of the planet, because the necessity of programmers will increase in the future. In contrast with this the conservative secondary education system still does not find an adequate place for teaching Informatics in secondary school almost all over the world. Instead the authorities try to replace teaching of Informatics with teaching of Information technology where many things are included in the curriculum that student know from their everyday use of mobile devices.

We deeply believe that secondary schools have to find enough room in the curriculum for an integrated discipline *Informatics and Information technologies* (in a similar way as integrated disciplines *Philosophy and Society, Biology and Human health, Chemistry and Human environment*, and so on). I&IT has to be mandatory and to be included in the curricula of all grades – from the first to last. In the beginning the stress will be, of course, on usage of computers and computerized devices, and on the technology. But from one moment (5^{th} grade for example) computer programming has to be introduced in the program in such way that, in the last 3-4 years in the secondary school, classes in I&IT to be dedicate to Informatics almost. Only in such way the students that continue their education in the universities will be prepared for the ambitions of the Departments of Informatics to produce well educated programmers.

When this idea becomes realistic, then the Olympiads in Informatics, as well as the Project contests, will obtain a new vision. The number of participants, especially in middle grades (4^{th}-7^{th}) will increase, the results of the countries that now participate on the principle "participation is more important than the victory" will ameliorate. This will be in accordance with needs of the society of more and more qualified programmers.

References

1. Manev Kr. Mathematics and Discrete Mathematics. Proceedings of Seventh International Conference "Discrete Mathematics and Applications". South-West University, Blagoevgrad, June 2004. p. 131-138. ISBN 954-680-363-4.
2. Manev Kr., Kelevedjiev E., Kapralov St. Programming Contests for School Students in Bulgaria. In: International Journal "Olympiads in Informatics". v.1, Vilnius, 2007. p. 112-123. ISSN 1822-7732.
3. Manev Kr. Grading Systems in Teaching Programming. Proceedings. of International Scientific Conference "Informatics in the Scientific Knowledge". Varna, June, 2014. p 230-239. ISSN 1313-4345.
4. Manev Kr., Maneva N. On a Metodology for Creating School Curricula in Computing. Olympiads in Informatics, 11, 2017. p. 93-107. ISSN:1822-7732.
5. Manev Kr., Yovcheva B. First European Junior Olympiad in Informatics, Olympiads in Informatics, 11, 2017. p. 171-173. ISSN:1822-7732.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 17-25 E-ISSN 2587-3636

CZU: 37.016.046:004 DOI: 10.36120/2587-3636.v18i4.17-25

ONE PROPOSAL FOR SYLLABUS FOR EJOI COMPETITORS PREPARATION, BASED ON SPIRAL APPROACH TO PROGRAMMING TEACHING

Biserka YOVCHEVA*, senior lecturer, PhD on specialty Methodology of informatics
Konstantin Preslavsky University of Shumen, Bulgaria

*Chairman of the Organizing Committee of IATI (International Autumn Tournament in Informatics) in Shumen, Bulgaria; initiator and organizer of European Junior Olympiad in Informatics; manager of A&B School in Shumen and teacher and trainer of some of the best competitors in informatics in the world; author of unique methodology "Spiral approach to training in informatics" for training in informatics of 10 -11 year old students.

Summary. This article presents a program for preparing students from Bulgaria for programming competitions. This program is based on the experience gained by the author in preparing the competitors for the international Olympiads: European Junior Olympiad in Informatics (EJOI), Balkan Olympiad in Informatics (BOI), International Olympiad in Informatics (IOI) etc.

Keywords: Junior Olympiad in Informatics, logical thinking, age characteristics of children, spiral approach to programming teaching.

PROIECT DE CURRICULA PENTRU PREGĂTIREA COMPETITORILOR EJOI, BAZATĂ PE ABORDAREA DE TIP SPIRALĂ ÎN PREDAREA PROGRAMĂRII

Rezumat. În articol se prezintă un proiect de curricula de pregătire a elevior din Bulgaria pentru competiții de programare. Curricula se bazează pe experiența acumulată de autor în pregătirea concurenților pentru olimpiadele internaționale: Olimpiada Europeană de Informatică pentru Juniori (EJOI), Olimpiada Balcanică de Informatică (BOI), Olimpiada Internațională de Informatică (IOI) etc.

Cuvinte cheie: Olimpiada de Informatică pentru Juniori, gândire logică, particularitățile de vârstă ale copiilor, abordarea în spirală a predării programării.

Introduction

European Junior Olympiad in Informatics (EJOI) is an international programming competition, which purpose is to enable international expression for beginners in programming competitions.

The experience is shown about 4-5 years are needed for preparation of one student to participate in International Olympiad in Informatics (IOI). In this case, the preparation includes not only teaching material, but also training and refinement of thinking of the competitors. They have to develop programming skills and habits and it takes a long time.

The knowledge the future competitors have to acquire is extremely hard and require big focusing.

In view the fact in most countries informatics is not learning in standard schools or it has been learned at the higher educational level, it becomes clear, that students have to learn programming in extracurricular or extra scholar forms. There is only one stimulus

for learning – the participation in many competitions – national and international. Of course, the final purpose is IOI, but this is a high level competition and the participation is very limited (4 competitors of a country). Thus the stimulus IOI become something unattainable and many young students (10-11-12 year old) drop out of the preparation. For a little child at this age it is very difficult to devote to one goal for several years by only one reason – the chance for national team and eventually participation to IOI.

A goal to remote and hard to reach, often makes empty efforts of otherwise talented competitors.

Exactly in this meaning EJOI is an opportunity after relatively shorter period of learning (2-3 years) the children to have an international participation. EJOI is a competition closer in time (the perspective of student who likes attainable and makes reasonable earlier preparation and efforts of little (10-11-12 age) students.

In Bulgaria, exactly for these reasons, the national competitions of informatics are organized within 5 aged groups – group E (age 10-11), D (age 12), C (age 13-15), B (age 16-17) and group A (age 17-20). The competitions give the possibility for national participation for more students and motivate them for better participation and great diligence.

Programming learning of 10-12 aged students is strongly specific and requires consideration with age characteristics of children - logical thinking, the ability to focus on a problem etc.

The syllabus for preparation to the programming competitions

The characteristics of this education and one approach for solving the problems considering them are suggested in my PhD: "Spiral approach to programming teaching 10-11 age students" as in a lot of papers on the topic too.

This paper deals with example syllabus for 3-year preparation students (age 11-12-13) for participation in EJOI. This syllabus is based on the conclusions and findings, drawn in that study.

The syllabus will be briefly provided for the first two years, because it is fully developed in the dissertation. Then it will be made a transition to teaching in 3rd year and themes included in this teaching.

1. ***The syllabus for the first year***

1.1. *Subjective*

During the first year of learning the students well meet with the concept program and programming. They have to get to know a programming on general, no matter what language they learn. Due the fact for EJOI C++ language was fixed, the syllabus will be illustrated with it.

The main purposes during the first learning year are to learn the basic language construction and the fundamental compound data types – one dimensional arrays and strings.

1.2. *Fundamental problems in basic programming learning*

- The most of the concepts related to programming are new for the students. It is not a good idea to learn a lot of concepts at once.
- Another difficulty at this stage of learning is the level of students' abstract thinking. The abstraction in the programming turns out to the higher level the little students can understand and learn. The overcoming of this problem is achieved with a gradual increasing of abstraction level.

The basic problem in abstraction is the parametrization of programming problems. For example, the most popular problem in basic programming learning is calculating the maximal elements of ***n*** numbers. There ***n*** is a parameter for number of numbers.

There are a few parameters in this problem – the numbers among which the maximum is sought and their number. If the task is offered the students in this variant for the first time, they will be confused and understanding of the task will be very difficult for them. It is therefore a good idea to propose it at two stages:

1. The first one is right after learning the test conditions and selection statements. In this case the problem is formulated for fixed number of numbers (example 3). Then the parameter ***n*** (number of numbers) is ignored and the algorithm for calculating of maximal element is learned only for 3 numbers.
2. The second return to this problem is immediately after organization of loops and loop stations. At this moment this is the second parameter – unknown number of numbers is included. And because the students already know the algorithm from the first stage, the have to focus now only how to process with unknown number of numbers.

Learning the concepts "variable" and "statement" is the next very important problem.

It is very important to explain the students that the program is a tool for manipulating the computer and that the basic devices of the computer are the memory and the processor. The programming language have to provide tools for manipulating these devices. The statements are the tools for manipulation the processor and the variables, the constants and the literals, are the tools for manipulation the memory.

This simplistic concept for a program allows learners to better understand the importance of both concepts and to differentiate them. Typical example is that in the beginning of programming learning the students do not make difference between the declaration of variable (int a;) and reading the value of variable from standard input (cin>>a;).

- Using different data types appears again the difficulty for beginners. That is why it is appropriate to include in the problems all simple data types. This should be done with all types of algorithms linear, algorithms, branching algorithm, loop algorithms etc.

1.3. *The important themes in first year of programming learning*

Here are the generic themes included in this syllabus (fig.1).

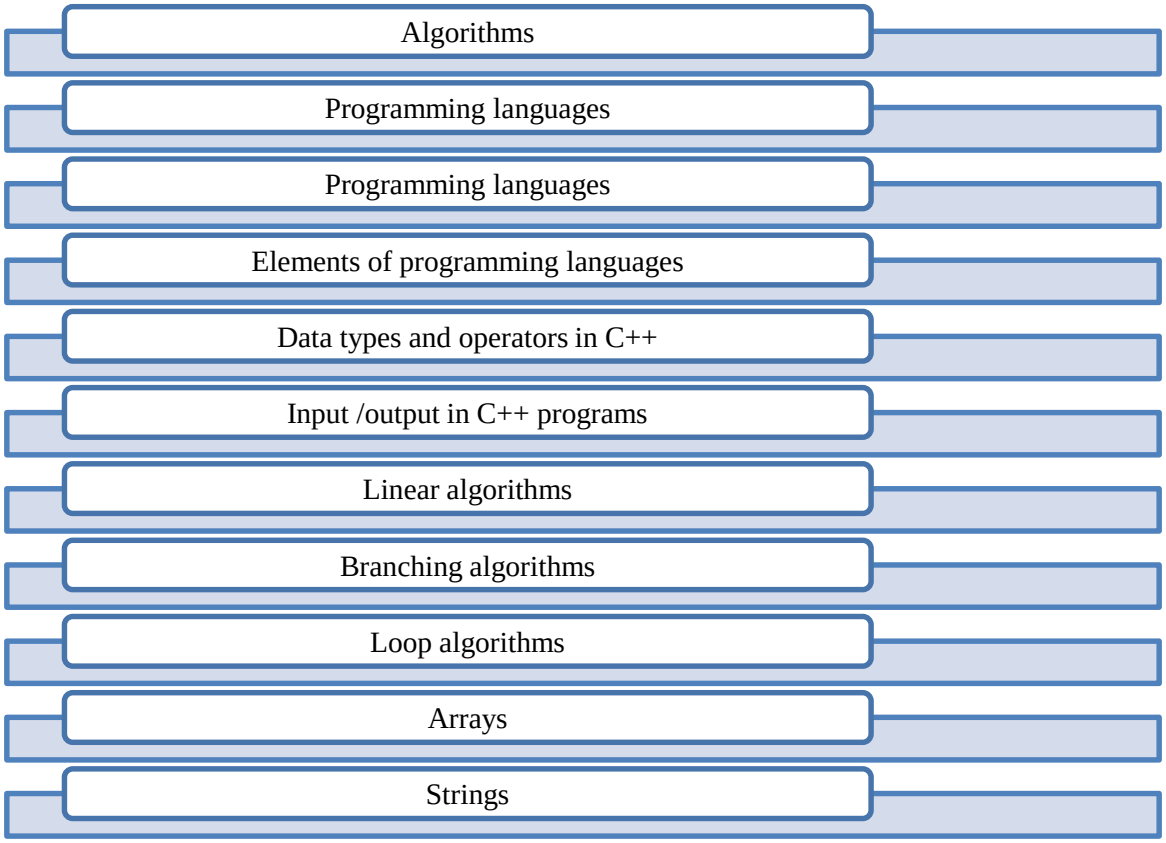


Figure 1. The important themes in first year of programming learning

1.4. *The important algorithmic constructions which have to be learned during the first year* (fig. 2)

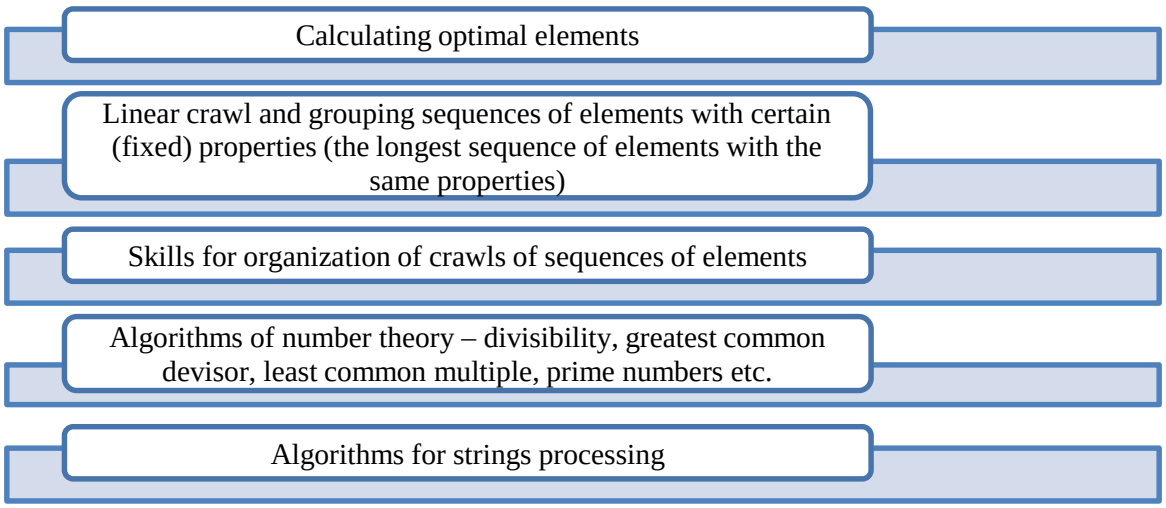


Figure 2. The important algorithmic constructions which have to be learned during the first year

2. *The syllabus for the second year*

1.1. *Subjective*

This syllabus can be found in [8], [2]. During the second year of learning the students have to learn well all C++ tools for creating algorithms. They have to learn the basic data structures like stack, queue and their use.

They need to learn what this recursion is. The students have to create the basic recursive algorithms.

During this year there should be working for improvement the students' skills – to write correctly and effectively programs. To choose correctly tools (statements, data structures, etc.)

The students have to reach a higher level for understanding and creating computer programming during this year.

2.2. *Fundamental problems*

The problems arising during the second year of study are related to the higher level of understanding the concept computer program.

- The students' ability to structure their programs is very serious problem at this stage. Now their abilities have to be built completely.

  The basic difficulties are related to the decomposition the tasks to subtasks for better and effectively code realization. Many tasks must be prepared for training of these students' skills.
- Another problem is the understanding the concept complexity of algorithm and the need of creating effectively programs. There have to explain (with a lot of proper examples) to children why the optimization of given algorithm is necessary if it works correctly for the most of tests. The motivation has to be created in students for writing the effective programs.
- Understanding compound data types is a very important problem. It is arisen from the students' incompetence for analysis and synthesis of problems and insufficient development of their abstract thinking. The students still being understood one-dimensional and two-dimensional arrays and strings, but fore them is very difficult to perceive polytypes data types like struct and class.
- The algorithms at this stage are not such intuitively, as they were in the first year. The students for the first time meet themselves with standard algorithms, which are already made up from another people. They have to learn them and use them in the problems solving (they sometimes need to be modified).

  In many cases the students are confident that they come up with a solution to a problem, therefore the find it difficult to motivate them to learn ready-made algorithms.

- Building of recursive thinking is the last problem in this part. In this year the students have to get acquainted with the recursion as a tool for solving programming problems. The students have to learn basic recursively algorithms.
  One of the biggest obstructions is to learn to find recurrent connections and to imagine how exactly works the recursive algorithm.

Every one of these problems could be the subject of a separate paper. In this paper they are marked only.

2.3. *The important themes in the second year* (fig.3)

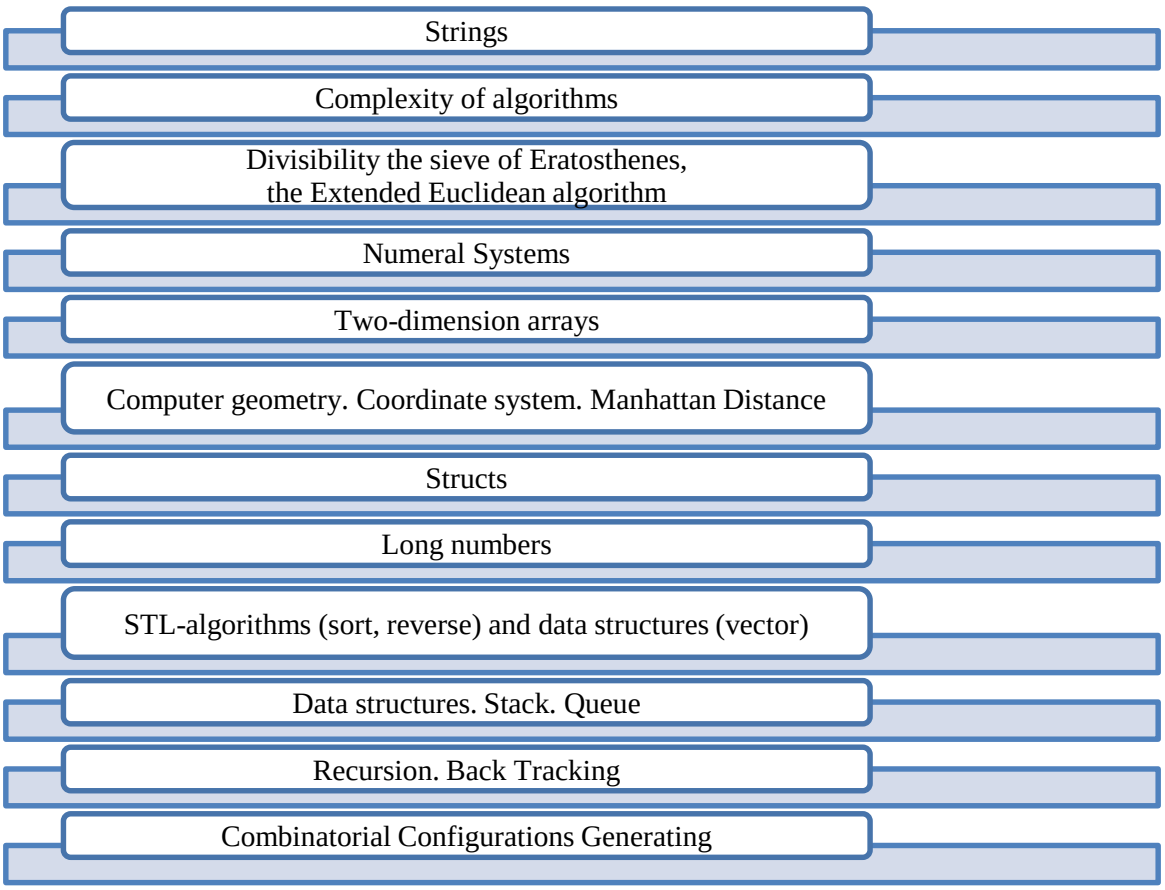


Figure 3. The important themes in the second year

2.4. *The important algorithms in the second year* (fig.4)

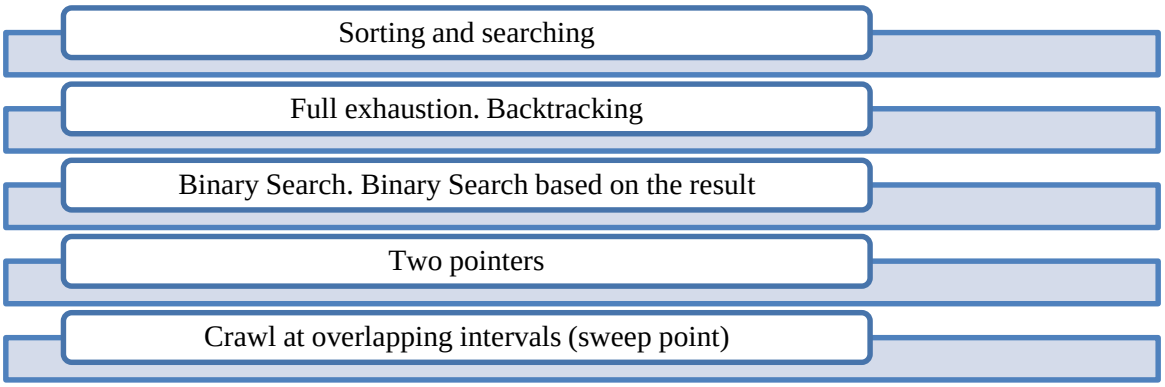


Figure 4. The important algorithms in the second year

3. *The syllabus for the third year*

No official material has been published related to this issue. In the different countries there are lists of themes, which the students have to know, but the author has not found official structured syllabus, not to mention some methodical notes on the subject.

The book "The tasks for C group" by author's team with a leader Professor Krasimir Manev will be published soon. The tasks from Bulgarian selection competition for the Bulgarian team to EJOI will be analyzed in this book. In this part will be suggested the syllabus for the third year, as in the previous parts.

3.1. *Subjective*

In the end of the third year the students have to know all the themes which are necessary for solving EJOI tasks. During this educational year the students get to know with algorithms and the programming techniques also and to know more advanced data structures.

The students have to be introduced in graph theory – the basic graph definitions, the basic algorithms for traversing of tree or graph data structures and the algorithms that solve the shortest path problem for a graph – Floyd's algorithm, Ford-Belman's algorithm.

It is still debatable whether knowledge of tree is needed (except the STL data structures set and map).

In some of the tasks, the authors require the use of segment or indexed tree for the best solution. Experience shows that it is very difficult to study this tree structures during a year.

However, the students who participated in EJOI are the best for their age. It is noticeable that the competitors who win the medals from EJOI, like Nikoloz Birkadze, Ildar Gainullin, Egor Lifar, Sofija Melnik, Martin Kopchev, Victor Kozhuharov, etc., will perform perfectly themselves in IOI in the next year.

In that sense, it is possible the best student to succeed to learn and to use trees in their problems' solving.

The theme for achievement of the best competitors may be further discussed. The subjective in this paper is to present the minimum of knowledge, which have to learn EJOI competitors.

3.2. *Fundamental problems*

- Using pointer is difficult for the students at this stage. The students have to know the management memory and pointer technical. This requires a higher level of abstract thinking, which fortunately, good competitors already have. More time is needed and the typical technique tasks are selected for using pointers and dynamical data structures – stack, queue, list, etc. Using dynamic arrays must be demonstrated,

however, it is better for students to use the class vector that provides sufficient functionality at this stage.

- The students' skills to calculate the complexity of own algorithm is another problem. Sometimes the full exhaustion is not considered resource-intensive processes, because the recursive function appears compact and loops in them implicit (invisible). The operations with STL's data structures also are not calculated with correct time for their execution.

For example, the function erase of the class string can reach complexity O (n), where ***n*** is the size of the string. But since in the students code it is called thorough the only one statement (calling the function), it is often overlooked at calculating complexity.

- At this stage the students have to mastering another important skill – to search information and to learn it alone, without the obligatory presence of a teacher (trainer).

Conclusions

Self-improvement of knowledge and skills is a very important premise for the further growth of small competitors.

The presented syllabus is based on years of author's experience in preparation competitors in Bulgaria.

The syllabus is a subject to discussion by the General Assembly of EJOI. Additional methodical guidelines have to be developed. The separation into three different years is conditional. The experience shows that it is more effective and guarantees good results. In another situation and in other academics the education may take less time.

The international discussion of the syllabus and the methodology for preparation competitors is very important for successful realization of young people and the future successes of EJOI competitors.

This paper could be a successful start of this discussion.

References

1. Yovcheva B., Ivanova I. First steps in C/C++. Sofia: KLMN, 2006.
2. Yovtcheva B., Ivanova I., Petrov P. Second steps in C/C++. Sofia: KLMN, 2014.
3. Yovcheva B. Spiral teaching programming to 10 – 11 year old pupils (based on the language C++). Mathematics and mathematical education. Sofia, 2007.
4. Yovcheva B., Mollov A. The idea for realization of spiral approach in the early training programming. The Vth Balkan congress "The Education, The Balkans, Europe", 22-24 June 2007, Stara Zagora.
5. International Olympiad of Informatics: http://olympiads.win.tue.nl/ioi/.

6. Mollov A., Yovtcheva B. Main problems connected with developing programming program of studies for 10 – 11 year old children (based on the C++ language). Collection "Covering and value of teaching". Reports from the third autumn scientific conference of the faculty of preschool and primary school education. University of Sofia, Veda Slovena –GZ Sofia, 2005.
7. Private School of Mathematics and Informatics A&B in Shumen http://ab-bg.com.
8. Yovcheva B. Spiral Teaching Of Programming To 10 – 11 Year-Old Pupils After Passed First Training (Based On The Language C++). The 3rd International Conference "Iseep Informatics in Secondary School Evolution and Perspectives Informatics Education – Contributing Across the Curriculum", Torun, Poland, 2008.
9. Mollov A., Yovcheva B., Petrov P. Internal contests as an element of the training of pupils for competitions in informatics. Математика и математическо образование. София, 2009. с. 217-233. ISBN: 978-954-8212-01-4.
10. Yovcheva B., Momcheva G., Petrov P. JBOI – one more possibility for increasing the number of competitors in Informatics. Olympiads in Informatics. An International Journal. Institute of Mathematics and Informatics. Vilnius, 2009 г. ISSN 1822-7732.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 26-36 E-ISSN 2587-3636

CZU: 372.8004 DOI: 10.36120/2587-3636.v18i4.26-36

PEDAGOGICAL MODELS FOR PREPARING GIFTED AND EXTRA-GIFTED STUDENTS FOR NATIONAL AND INTERNATIONAL COMPETITIONS IN INFORMATICS

Anatol GREMALSCHI*, professor, habilitated doctor

Tiraspol State University

*Team leader of the national team of Moldova at IOI (1997-2013); President of the Scientific Council of the Balkan Informatics Olympiad (2006, 2017); President of the Republican Olympic Council in Informatics.

Summary. It is argued the need to develop new pedagogical models for preparing gifted and extra-gifted students for national and international competitions in Informatics. The models in question have to be based on a common structure and be oriented towards the development of an extended set of competences specific to Informatics and Mathematics, competences which cannot be found in the national curriculum. The teaching strategies will be based mainly on simultaneous teaching, self-learning and the creation of simulated competition environments.

Key words: Informatics, Olympiads, specific competences, pedagogical models, didactic strategies.

MODELE PEDAGOGICE DE PREGĂTIRE A ELEVILOR DOTAȚI ȘI SUPRADOTAȚI PENTRU COMPETIȚIILE NAȚIONALE ȘI INTERNAȚIONALE DE INFORMATICĂ

Rezumat. Este argumentată necesitatea elaborării unor noi modele pedagogice de pregătire a elevilor dotați și supradotați pentru competițiile naționale și internaționale la Informatică. Modelele în cauză trebuie să se bazeze pe o structură comună și să fie orientate spre formarea unui set extins de competențe specifice Informaticii și Matematicii, competențe ce nu se regăsesc în curriculumul național. Strategiile didactice se vor baza, în principal, pe predarea simultană, învățarea de sine stătătoare și pe crearea de situații simulate de competiție.

Cuvinte cheie: Informatică, olimpiade, competențe specifice, modele pedagogice, strategii didactice.

Short history

In Republic of Moldova every year, starting with 1985, the Republican Informatics Olympiad is held. On the whole, the problems proposed to the competitors at these national competitions are algorithmic in nature and require from the competitor the development of computer programs that would conform to predetermined memory and execution time restrictions. For the most part, the knowledge and skills required to be mastered by competitors fall within the school course of Informatics, the high school level of general education. Ten years after the launch of Republican Olympics in Informatics, Republic of Moldova also started participating in the International Olympiad in Informatics (IOI), this had a profound impact on both the teaching of Informatics within our country and on the preparation processes of students for national and international competitions. Obviously, with the development of Informatics as a science and the improvement of its teaching-learning-evaluation methods as a school discipline, the need has emerged to systematize

the process of preparing gifted and extra-gifted students and to base it on the principles of constructivist learning.

Article goal

The purpose of this article is to develop a pedagogical model for the training of gifted and extra-gifted students for national and international competitions in Informatics, based on the principles of constructivist learning and the full exploitation of the opportunities offered by the new computer-assisted training systems.

Comparative analysis of the IOI Sylabus and the Informatics school curriculum in Republic of Moldova

It is a known fact that the International Olympiad in Informatics is held on the basis of a curriculum approved by the International Scientific Committee, a document which is updated practically every[1]. This document, de facto, only contains what knowledge (content) the IOI participant must possess, without making an explicit reference to his/her competences. The contents in questioner are grouped into the following areas of knowledge: Mathematics; Computing Science; Software Engineering; Computer Literacy.

Until 1999, Informatics, as a school discipline, was taught on the basis of programs which only listed the contents to be learned, without explicitly formulating learning objectives and the competences expected to be trained and developed in students. Starting with the year 2000, the teaching of Informatics in Republic of Moldova schools was carried out on an objective-based curriculum, starting with 2010 – on the basis of a competency-based curriculum. Currently, starting with the 2019/2020 school year, the teaching of Informatics in the Moldovan schools is based on a modernized curriculum, which is also based on competences [2].

Unlike IOI Syllabus, the central pillar of the National Informatics Curriculum consists of a set of digital competences, formulated according to the following definition:

> "School competence is an integrated system of knowledge, skills, attitudes and values which are: acquired, formed and developed through learning; whose mobilization allows identifying and solving different problems in diverse contexts and situations" [3].

Since the contents of the School Curriculum in Informatics are only recommendations, a direct comparison with the IOI Syllabus is difficult. However, starting from the fact that knowledge is an indispensable component of the "integrated system" that constitutes the actual competence, an indirect comparison of the documents in question is possible. For this purpose, we will treat the contents as connecting elements that lead to digital competences [4], implicitly assumed by IOI Syllabus and explicitly stated in the school curricula in Informatics and Mathematics.

Table 1. The knowledge required by IOI Syllabus and the school curricula in Informatics and Mathematics

IOI Syllabus Topics	School curricula	
	Informatics	Mathematics
Arithmetics and Geometry	-	+
Functions, relations, and sets	-	*
Basic logic	+	+
Proof techniques	-	*
Basics of counting	-	+
Graphs and trees	-	-
Programming Fundamentals	+	-
Algorithms and problem-solving	+	-
Fundamental data structures	*	-
Recursion	+	-
Event-driven programming	-	-
Basic algorithmic analysis	+	-
Algorithmic strategies	*	-
Algorithms	*	-
Data structures	*	-
Automata and grammars	+	-
Geometric algorithms	*	*
Software design	-	-
Using APIs	-	-
Software tools and environments	+	-
Software processes	-	-
Software requirements and specification	-	-
Software validation	-	-
Software project management	-	-
Formal methods	-	-
Computer Literacy	*	-

Legend: “-” – missing; “+” – is studied in full; “*” – is partially studied.

From the data presented in the table above it can be concluded that the school curricula in Republic of Moldova does not cover all the topics required by IOI Syllabus and, consequently, the preparation of students for participation in international competitions requires teaching-learning-evaluation activities that cannot be carried out only based on the classical pedagogical models, currently used in general education.

Typology of problems proposed to students in the Republican Olympics in Informatics

The analysis of problem statements proposed to the students in Republican Olympics in Informatics during the 2010-2019 years, shows that they can be classified, with small exceptions, in the following categories: Combinatorics; Computational graphics; Graphs; Computer arithmetic; Information theory; Simulation; Formal grammar.

A very small number of problems, especially those of advanced complexity (about 5% of those proposed to the competitors of said competitions), relate to two and even three of the categories listed above. Such were the problems attributed to each of the respective categories.

From the data presented in Figure 1 it is observed that most problems, 29.2% of those proposed to contestants, are in the field of combinatorics, the students having the task of implementing a programming technique (Greedy, Backtracking, Divide et Impera, etc.) which would count, generate, analyze or determine the "largest", "smallest" or "best" object from an set. We emphasize that the mathematical and informatics fundamentals of this type of problems are indeed studied within the respective school disciplines.

Formally, the problems in the following weight categories – Computational graphics (20.0% of proposed to competitors) and Graph theory (18.5%) cannot be solved based on only the subjects studied in the school disciplines of Mathematics and Informatics. Despite this fact, annually about 5-10% of contestants solve such problems in full, which once again confirms the significant role or formal, non-formal and informal extracurricular preparation of students gifted and extra-gifted in Informatics.

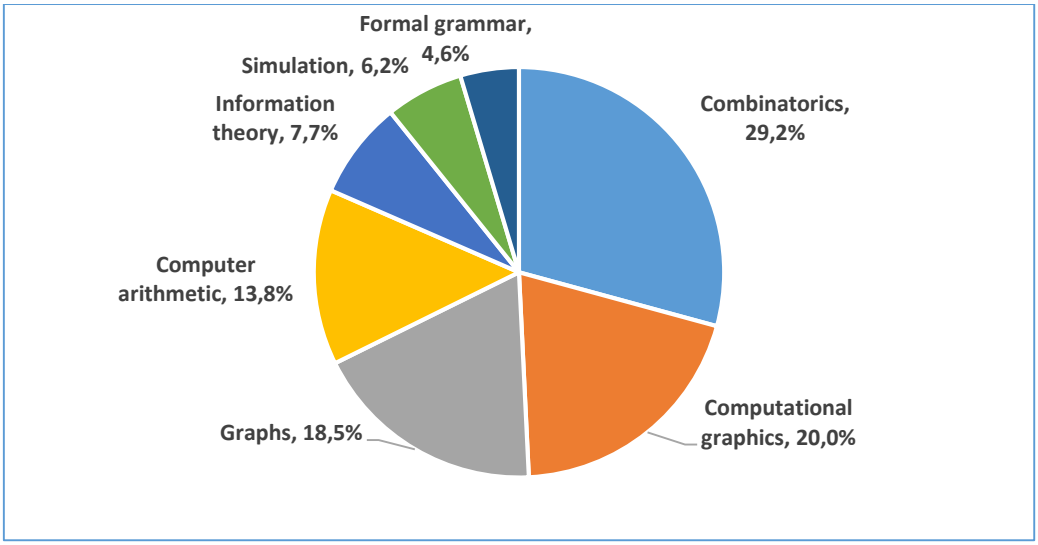


Figure 1. Distributions of problems proposed to contestants in the Republican Olympics in Informatics (years 2010-2019, high school grades)

The remaining problem categories – Computer arithmetic (13.8%), Information theory (7.7%), and Formal grammar (4.6%) – can surely be solved based on the subjects studied in the school disciplines of Informatics and Mathematics, however, as evidenced by the student results at Republican Olympics in Informatics from 2010-2019, their power

of competitor ranking is reduced. An exception to this rule is attested by the results from the Simulation (6.2%) problem category, solving which requires from competitors not only the application of traditional programming techniques, but also of creative approaches.

The skill set required to be developed as a priority in the process of preparing gifted and extra-gifted students for national and international competitions in Informatics

According to the regulation in force [5], the subjects proposed to students in general education competitions must comply with the provisions of the school programs and be related to their higher demands, both in terms of objectives volume and their degree of complexity.

We emphasize that in the framework reference of the national curriculum the term "programs" has the meaning of "curricula by disciplines". Therefore, students who would be selected only on the basis of national competitions, conducted off the "school curricula", would by definition not be prepared to participate in international competitions. Moreover, the experience of organizing and conducting the 32 Republican Olympics in Informatics clearly demonstrates that the use of only the subjects that correspond to the school programs does not even allow the classification (ranking) of students according to their performances, as such topics are solved within 20-30 minutes by most contestants. Therefore, the pedagogical model expected to be developed must be based on a broader interpretation of the term "school programs", including in them subjects studied in extracurricular activities, such as: Informatics clubs (groups) in educational institutions and student creativity centers, the numerous projects implemented by big companies in the field of information technology and communications, the online programming projects implemented by enthusiasts in the field of computer science, etc.

At the same time, in the process of preparing gifted and extra-gifted students for national and international competitions in Informatics, emphasis will also be placed on the formation and development of transversal key-competences, as explicitly indicated in both national documents [6, 7], and in European ones [6]. In particular, we will insist on two of them, as defined in Council Recommendation of 22 May 2018, specifically (1) Personal, social and learning to learn competence, and (2) Entrepreneurship competence. According to this document, the respective competencies are defined as follows:

> „*Personal, social and learning to learn competence* is the ability to reflect upon oneself, effectively manage time and information, work with others in a constructive way, remain resilient and manage one's own learning and career. It includes the ability to cope with uncertainty and complexity, learn to learn, support one's physical and emotional well-being, to maintain physical and mental health, and to be able to lead a health-conscious, future-oriented life, empathize and manage conflict in an inclusive and supportive context.
>
> *Entrepreneurship competence* refers to the capacity to act upon opportunities and ideas, and to transform them into values for others. It is founded upon

> creativity, critical thinking and problem solving, taking initiative and perseverance and the ability to work collaboratively in order to plan and manage projects that are of cultural, social or financial value" [8].

An advanced development of especially these skills would allow students to go beyond the limits imposed by the school programs and to acquire, by themselves or under guidance, the subjects that cannot be found in the Mathematics or Informatics curriculum, but explicitly assumed by the IOI Syllabus.

We emphasize that the training of gifted and extra-gifted students for national and international competitions in Informatics involves the formation and development of a set of competences required to act autonomously. The set in question is explicitly defined in the documents of OECD (Organization for Economic Co-operation and Development) [8, 9] and assumes:

- Act within the big picture;
- Form and conduct life plans and personal projects;
- Defend and assert rights, interests, limits and needs.

In the case of preparing students for national and international competitions in Informatics, a very important role belongs to the competence of *Form and conduct life plans and personal projects*, defined in the documents cited above as follows:

> "This competency applies the concept of project management to individuals. It requires individuals to interpret life as an organized narrative and to give it meaning and purpose in a changing environment, where life is often fragmented. This competency assumes an orientation toward the future, implying both optimism and potential, but also a firm grounding within the realm of the feasible. Individuals must be able, for instance, to:

- Define a project and set a goal;
- Identify and evaluate both the resources to which they have access and the resources they need (e.g. time and money);
- Prioritize and refine goals;
- Balance the resources needed to meet multiple goals;
- Learn from past actions, projecting future outcomes; and
- Monitor progress, making necessary adjustments as a project unfolds" [7].

Obviously, in the process of preparing for national and international competitions, the student must have the ability to define as specifically as possible the goals to be achieved, to stagger the activities to be carried out over time, to rationally use the available time resources, to prioritize and redefine the goals. At the same time, it requires the development of resilience to stress, awareness of own limits and objective evaluation of own achievements.

Overall, the set of skills needed to be developed as a priority in the process of preparing gifted and extra-gifted students for national and international competitions in Informatics will include:

- Scientific perception of the role and impact of Informatics phenomena in the contemporary society;
- Use of informatics and mathematical concepts, methods and algorithms in various application contexts;
- Applying informatics and mathematical reasoning in identifying and solving problems;
- Exploring problem situations through modeling, planning and conducting virtual experiments within digital environments;
- Developing strategies and designing activities for solving problems and problem-situations;
- Algorithmization of methods of analysis, synthesis and solving of problem-situations;
- Implementation of algorithms in programming environments;
- Analysis of problem solving, problem-situations in the context of correctness, simplicity, clarity and significance of results.

The taxonomies used to establish the objectives of student preparation

In the National Curriculum of the Republic of Moldova, the basic taxonomy recommended for establishing the learning objectives is that of Bloom-Anderson [10]. We'd like to remind you that in this taxonomy the following hierarchical levels of cognitive abilities are established: (1) remember; (2) understand; (3) apply; (4) analyze; (5) evaluate and (6) create. The complexity of the cognitive capacities increases with the progression from the lower hierarchical level (reproduction) to the higher one (creation). Starting from the fact that in the process of teaching-learning-evaluation the emphasis must be placed not only on the formation of the cognitive competences, but also of the functional-actionable ones, the Curriculum of Informatics for general education in the Republic of Moldova also proposes the use of Simpson's taxonomy [11], especially of the higher levels: Adaptation (adapts, alters, changes, rearranges, reorganizes, revises, varies) and Origination (arranges, combines, composes, constructs, creates, designs, originates).

Worth mentioning that on the Internet there are works that try to adapt Bloom's taxonomy to the specifics of digital skills training and development. Initially the works in question were referring to digital literacy [12], but subsequently they included the development of algorithms and their implementation on the computer [13]. Thus, in the case of the Analyzing level we will mention the action words: Calculating, Estimating, Integrating, Planning, Structuring; in the case of the Evaluating level – the action words: Assessing, Checking, Detecting, Measuring, Monitoring, Testing, Validating; in the case of the Creating level, the action words: Composing, Designing, Developing, Integrating,

Inventing, Making, Originating, Producing, Programming, Publishing, Simulating, Solving etc. Obviously, these verbs should be used to define the learning and evaluation objectives when preparing gifted and extra-gifted students for national and international competitions in Informatics.

The structure of the pedagogical models for preparing the gifted and extra-gifted students in Informatics

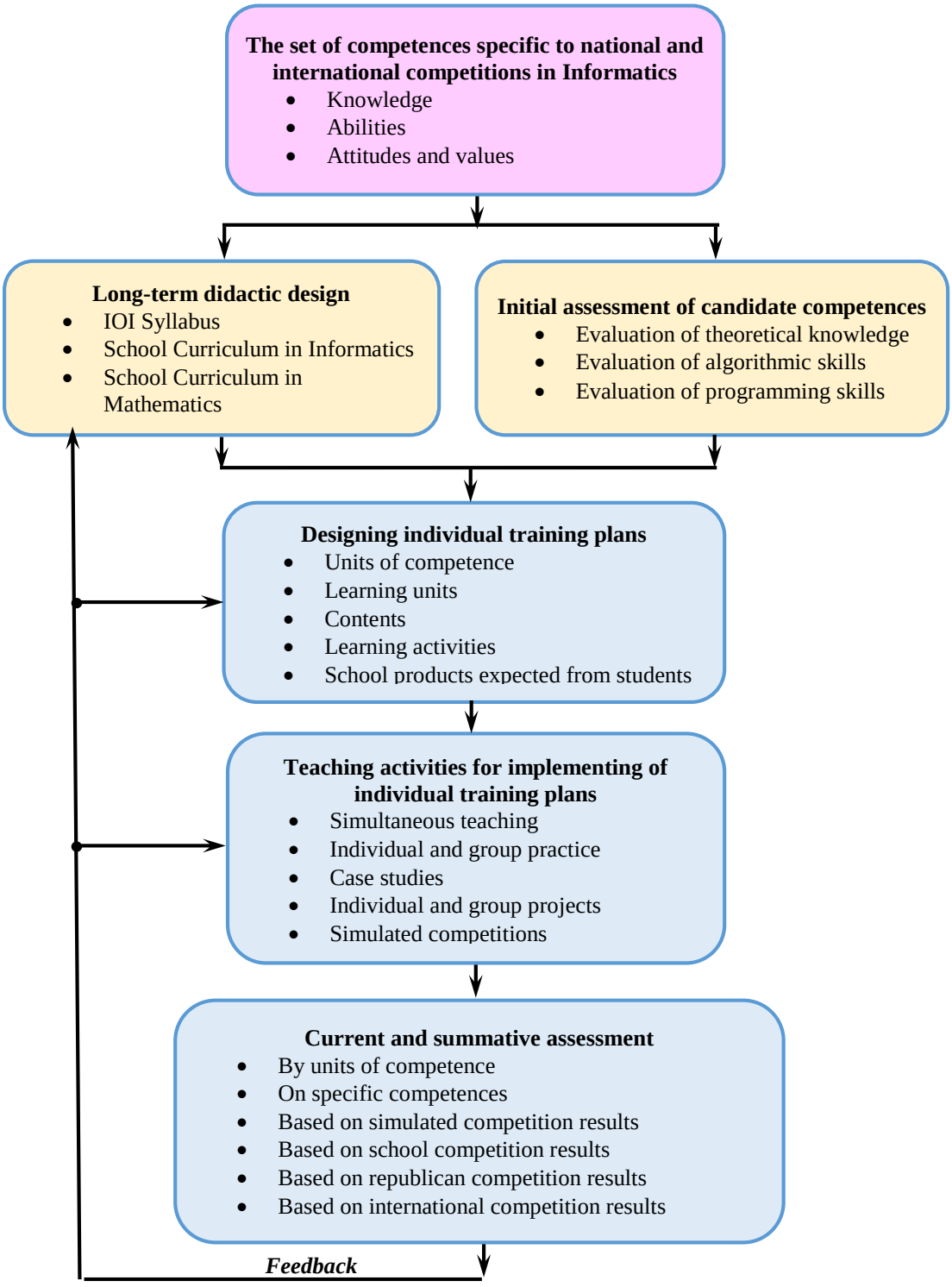


Figure 2. The structure of the pedagogical models for preparing the gifted and extra-gifted students in Informatics

The structure presented in figure 2 allows the generation of several pedagogical models for preparing gifted and extra-gifted students for national and international competitions in Informatics. The models in question are obtained through the concrete set of skills to be trained and developed in students according to their initial level of education and age groups of participants: if junior or senior for international competitions, grammar school or high school classes in the case of national competitions.

The long-term didactic design will aim to narrow down the specific skills for the initial and continuous training of gifted and extra-gifted students in Informatics, with the emphasis being on:

- Algorithmizing methods of analysis, synthesis and solving of problem-situations, educating in students creativity and perseverance;
- Implementation of algorithms in programming environments, training students in concentration and resilience;
- Exploring problem-situations through modeling, planning and conducting virtual experiments in digital environments, educating in students the analytical spirit, clarity and brevity.

Depending on the initial level of mastery of the mathematical and informatics skills required by IOI Syllabus, for each of the students an individual Training Plan will be developed, which will ensure that each of them will progress at their own pace. The teaching strategies (the forms of organization, the teaching resources, the methods of current and summative evaluation) will be based, mainly, on the simultaneous teaching, the learning on their own and on the creation of simulated competition situations. In order to develop students resilience to stress, the simulated competition environment will be similar to those that they would meet in the national and international competitions in Informatics, with the same style of problem statements, same methods of evaluation, and same style of interaction with the jury and the scientific committee.

The evaluation models used in the training of gifted and extra-gifted students in Informatics will be based on the criterion of appreciation, i.e. they will provide functional information, mobilizing students on achieving the expected objectives, to create those computer programs that comply with the restrictions imposed in the competition problem statements, while offering, at each stage, improvement solutions. The models used for the evaluation must have a corrective character, which offer the students opportunities to improve the developed products according to the feedback received from the teaching staff.

As a whole, the didactic activities required to realize the individual Training Plans must be based on the large-scale use of means and methods of computer-assisted training, using for this purpose interactive multimedia lessons, automatic knowledge testing systems, content management systems, learning management systems. It is important that the development of computer programs, their testing and debugging, the evaluation of the

programs by the teachers will be carried out in development environments similar to those used in international informatics competitions.

Conclusions

The organization of national and international competitions in Informatics only on the basis of general education school curricula does not ensure the ranking of competitors. The lessons learned during the organization of such competitions in the Republic of Moldova (years 1985-2019) and our country's participation in international competitions (years 1996-2019) confirm that the training of gifted and extra-gifted students in Informatics should be based mainly on extracurricular activities. There are significant differences between the national curriculum in Informatics and Mathematics and the IOI Syllabus, the latter including subjects characteristic of university education. This fact argues the need to develop and implement pedagogical models for the training of gifted and extra-gifted students for national and international competitions in Informatics oriented to the formation and development of an extended set of specific competences in Informatics and Mathematics and to organize the teaching-learning-evaluation process according to individual Training Plans. The teaching strategies will be based mainly on simultaneous teaching, self-learning and the creation of simulated competition situations.

References

1. The International Olympiad in Informatics Syllabus. https://ioinformatics.org/files/ioi-syllabus-2018.pdf.
2. Curriculum Național. Disciplina Informatică. MECC, 2019. https://mecc.gov.md/sites/default/files/informatica_curriculum_liceu_rom.pdf.
3. Guțu V., Bucun N., Ghicov A. ș.a. Cadrul de referință al curriculumului național. Min. Educației, Culturii și Cercet. al Rep. Moldova. – Chișinău : Liceum, 2017.
4. Communication from the Commission to the European Parliament, the Council, the European economic and social committee and the Committee of the regions on The Digital Education Action Plan. Brussels, 17.1.2018. COM (2018) 22 final.
5. Regulamentul privind organizarea și desfășurarea olimpiadelor școlare la disciplinele de studii. Aprobat prin Ordinul Ministerului Educației nr. 66 din 31.01.2012. http://aee.edu.md/sites/default/files/reg_olimp.pdf.
6. Council recommendation of 22 may 2018 on key competences for lifelong learning. Official Journal of the European Union, C 189/1.
7. Ibidem.
8. Organization for Economic and Co-operation and Development. Definition and Selection of Competences (DESECO): Theoretical and Conceptual Foundations. Strategic Paper. DEELSA/ED/CERI/CD(2002)9, 07-Oct-2002. JT00132752.

9. Organisation for Economic and Co-operation and Development. The Definition and Selection of Key Competencies. Executive Summary. http://www.oecd.org/pisa/35070367.pdf.
10. Lorin W. Anderson, David R. Krathwohl, Peter W. Airasian and more. A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. Abridged Edition, 2000.
11. Simpson E. J. The Classification of Educational Objectives in the Psychomotor Domain. Washington, DC: Gryphon House, 1972.
12. Churches A. Bloom's Digital Taxonomy, 2012. https://www.academia.edu/30868755/Andrew_Churches_-_Blooms_Digital_Taxonomy.pdf.
13. Bloom's Digital Taxonomy Verbs For 21st Century Students. https://www.teachthought.com/critical-thinking/blooms-digital-taxonomy-verbs-21st-century-students/

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 37-45 E-ISSN 2587-3636

CZU: 372.8004 DOI: 10.36120/2587-3636.v18i4.37-45

THE METHODOLOGY FOR PREPARING UNDERGRADUATE STUDENTS FOR OLYMPIADS IN INFORMATICS IN EXTENDED FORMAT

Liubomir CHIRIAC*, professor, habilitated doctor

Lilia MIHALACHE, associated professor, PhD

Tiraspol State University

*Deputy leader of the national team of Moldova at IMO (2004); President of the Scientific Council of the Informatics Olympiad within the TSU.

Abstract. This article examines didactic features for preparing university students for contests in informatics. Some methodical approaches are analyzed regarding the necessity of extending the formats of the contests.

Key words: algorithmic thinking, Olympiads in informatics, mathematical modeling.

METODOLOGIA DE PREGĂTIRE A STUDENȚILOR DE LA CICLUL I, LICENȚĂ, PENTRU OLIMPIADELE DE INFORMACĂ ÎN FORMAT EXTINS

Abstract. În acest articol se examinează aspectele didactice de pregătire a studenților de la ciclul I - licență pentru concursurile de informatică. Sunt analizate unele abordări metodice cu privire la necesitatea extin-derii formatelor concursurilor.

Cuvinte cheie: gândire algoritmică, olimpiade de informatică, modelare matematică.

1. Specific features of computer science contests

Contests in informatics represent both a method of testing knowledge and a method for classifying competitors according to their programming and algorithmic thinking skills. But at the same time, Olympiads in informatics must become a forum for algorithmic thinking that calls and urges absolutely all those who are fascinated and passionate about computer science to get involved in the process of running these competitions and to discover (rediscover) their own abilities and skills, to rebuild confidence in their own intellectual powers and to develop the "appetite" for progressing in at least some departments of computer science. The reasons for student's participation in contests in informatics are:

a) Cultivating algorithmic thinking;
b) More complex understanding of the programming skills;
c) Cultivating the ability to quickly focus and react in a short period of time;
d) Accumulating the necessary experience for the future IT programmers;
e) Elaborating an attractive CV which mentions participation in the regarding contests. These aspects are quite often making a differences in CV-s and are more taking into account in the process of employing people in IT companies;
f) Identifying weak and strong points regarding the student's theoretical and practical skills.

„Elite" computer science contests, and not only, are for „strong" students with deep knowledge, good and very good training in algorithmic, who think logically and have a speed of thinking higher than the average level. In these kind of contests, qualities determined by the reaction of thoughts, speed in relation to the decision-making process vis-à-vis the methods and approaches to be implemented, intuition and other time-dependent factors make the difference most of the time. But what do we do with those students who do not fall into that category? Thus, the experience gained over several years in the process of organizing and conducting several Olympiads in informatics highlights the following aspects:

1. There are students that are equally well trained in the field of computer science but are slower and have a slower speed in the process of logical analysis and thinking. The negative experiences, accumulated in various competitions cut them off from the desire to participate in other competitions, no longer wanting to get involved and losing confidence in their own powers.
2. Another category of students, due to objective and subjective considerations, have a lower training in the field of computer science, but the quite advanced intellectual capacities and the desire to perform would allow them to grow professionally in this field.

Taking into account the listed aspects, a special emphasis is placed on the process of preparing students of the above mentioned categories, also placed on the accomplishment of the following stages:

Stage* 1. *Appealing to the predetermined criteria, the preparation level of the students is determined.

1. The level of the algorithmic thinking of the students is identified.
2. The level of preparation from the theoretical point of view of the students is identified, including programming skills.
3. The quality regarding the speed of operational thinking is determined.
4. The level of preparation regarding working with the computer is determined.

The realization of points 1 to 4 implies organizing and conducting the tests for the students based on the mention criteria. Thus, the strengths and the weaknesses of the students will be highlighted from their results.

Stage* 2. *Elaboration and monitoring of the individual training program for students with emphasis on studying methods and techniques for solving problems.

1. An individual training route is elaborated with the realization of specific performance indicators.
2. The implementation of the elaborated program shall be monitored and evaluated.
3. Recommendations and suggestions may intervene in the training program originally developed.

Knowledge and successful application of solving strategies is the key to success for preparing students for Olympiads in informatics. Below we will examine some of the specifics.

2. Teaching aspects regarding the solution of problems in the process of preparing undergraduate students for contests in informatics

In our opinion, both in the process of preparing the students for the respective competitions and in the subjects proposed for conducting the competitions themselves, there must be problems which involve the elaboration of mathematical models from the perspective of the inter / transdisciplinarity of the exact sciences. In this way, emphasis will also be placed on the understanding of the interdisciplinarity of the methods, which have as common denominator principles and methods of investigation in various exact sciences and of the interdisciplinarity based on common concepts, fundamental for several disciplines.

One of the characteristics of the university Olympiads in informatics, organized within the Tiraspol State University, is about solving the problems that, as mentioned above, involve the development of mathematical models. In such situations, students develop their ability to develop mathematical models, and subsequently these models are solved through the studied algorithms. In order to develop mathematical models, students must be versed in different disciplines related to the exact sciences (mathematical analysis, geometry, physics, graph theory, number theory, abstract algebra, etc.) and their application, in many cases, including the concept STEM, [1-8].

The intellectual activity carried out by the student is focused on problem solving divided into the following types of problems:

A) ***Standard type problems***

The algorithm for solving these problems is known. The student only has to apply the respective algorithm or simple combinations of algorithms to solve the examined problem. This type of problem corresponds to abilities and programming skills that are determined by the content of the studied subject.

Example 1. *We have the product of 5 factors. To each of the first three factors of the respective product add the factor itself multiplied to one and the same number x. From each of the following two factors, of the product, the factor itself is multiplied by the one and the same number x. Determine the number x, if it is known that the product of these factors, after modification, remains unchanged.*

Solution. Suppose that *a, b, c, d, e* –are the product's factors, *x* – part of each factor.

According to the condition of the problem we write the relation:

$$(a+ax)(b+bx)(c+cx)(d-dx)(e-ex)=a\ b\ c\ d\ e;$$

$$a(1+x)b(1+x)c(1+x)d(1-x)e(1-x)=\ a\ b\ c\ d\ e;$$

$$a\ b\ c\ d\ e\ (1+x)^3(1-x)^2 = a\ b\ c\ d\ e;$$
$$(1+x)^3(1-x)^2=1;$$
$$(1+3x+3x^2+x^3)(1-2x+x^2)=1;$$
$$x^5+x^4-2x^3-2x^2+x=0;$$
$$x\,(x^4+x^3-2x^2-2x+1)=0$$

The mathematical model represents the following relation

$$x^4+x^3-2x^2-2x+1=0.$$

The student, by applying some methods of solving algebraic equations (Bisection method, Method of chords, Newton's method) and root localization, elaborates the program in a known programming language and determines two segments with positive roots [0, 1] and [1, 3] for the problem examined. For example, for the segment [0,1], the root by the Newton's method is 0.38939068, the root by the Method of chords is 0.38939080. And for the segment [1,3], the root obtained through the tangent method is 1.28879519, and respectively, the root by the Method of chords is 1.28912073 [4-8].

B) ***Problems that involve selecting suitable algorithms, methods or procedures for solving the problem***

In order to be solved, this type of problem requires the development of a not too complicated research process on the part of the student. In these situations, the research process involves selecting from the variety of known methods some methods and procedures or combinations of methods and timely procedures (which may require minor modifications) that would lead to solving the problem examined. In such cases, the student must have the necessary skills and abilities to choose from the set of known algorithms those that would perfectly fit the examined situation.

Example 2. *A hemispherical vessel is filled with water (Fig.1). At what angle α should the vessel be tilted so that one third of the water remains in the vessel? The result to be obtained with the error eps = 0.001.*

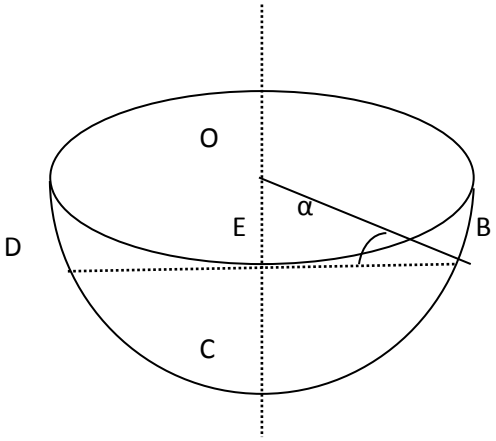


Figure 1. A hemispherical vessel

Solution. The volume of water in the hemisphere is calculated according to the formula:

$$V = 2\pi r3 / 3, (1)$$

and according to the condition of the problem we have that

$$1/3\ V = 2 / 9\pi r3\ (2)$$

Supposing that one third of the volume of the water will be situated in the hemispherical segment DBC , expressed by radius r. Therefore we will obtain:

$$V_{DBC}=\pi h^2(r-h/3)\ (3)$$

where $h=\left|OC\right|-\left|OE\right| = r - r\sin\alpha = r(1-\sin\alpha)$

$$V_{DBC}= \pi r^2(1-\sin\alpha)^2(r-r(1-\sin\alpha)/3)=$$
$$= \pi r^2(1-\sin\alpha)^2(3r-r+r\sin\alpha)/3)=$$
$$= \pi r^3/3(1-\sin\alpha)^2(2+\sin\alpha)=$$
$$= \pi r^3/3(1-2\sin\alpha+\sin^2\alpha)(2+\sin\alpha)=$$
$$= \pi r^3/3(2-3\sin\alpha+\sin^3\alpha)=$$
$$= \pi r^3/3(\sin^3\alpha-3\sin\alpha+2);$$

We obtained: $V= \pi r^3/3(\sin^3\alpha-3\sin\alpha+2);$ (4)

As the left sides of equations (2) and (4) are equal, so are the right parts:

$$2\pi r^3/9= (\pi r^3/3)(\sin^3\alpha-3\sin\alpha+2);$$

Bringing the above relation to the form $f(x)=0$, we obtain

$$\sin^3\alpha-3\sin\alpha+4/3=0.$$

We substitute $\boldsymbol{x = \sin\alpha}$ and obtain the equation:

$$\boldsymbol{x^3- 3x + 4/3=0}.$$

To solve this equation, the student can apply either the Bisection method or the Method of chords, or the Newton's method. For example, by developing the unified program for localization and the Method of chords, it is determined that positive roots are only in the segment [0; 1]. Taking into account the accuracy of eps = 0.001, we obtain that the root is 0.482403. Thus, **sin α** = 0.482403 and the angle under which the vessel must be tilted so that exactly one third of the water remains is $\alpha=28^0 48'$.

C) ***Problems for which the process of solving is not known***

In such situations, the student does not know the process of solving the problem. The respective type of problems involves the invention, independent discovery of the algorithm, method or procedures suitable for solving the problem.

For this purpose, the student is oriented to do the following:

- Recalling relevant information and recognizing part of the problem that can be solved by already known algorithms.
- Replacing some conditions of the problem with equivalent conditions in which the solving algorithm is known or represents a combination of known algorithms.
- In case the solution is not seen it is necessary to examine some particular cases of the problem. Particular cases may suggest the general solution of the problem.

- The intermediate results obtained, through various approaches, could suggest the correct algorithm, or at least part of the algorithm, which would subsequently lead to the general solution of the problem examined.

Example 3. *A cone-shaped vessel is given, directly circular with the vertical axis; its location and size are shown in Fig. 2. The vessel is filled with water and then the water is drained (drained) through a small circular hole, located at the bottom of the vessel (which is at tip B). Determine the water drainage time in the vessel.*

Solution. Supposing that the time *t,* for which the water level in the vessel will discharge at height *x,* is an arbitrary function *t (x),* we determine its differential *dt* for the changes of *x* at height *dx*.

Suppose the drop of water in the vessel to a small height *dx* is determined by the increase of time Δt. Then, assuming that, during this small period of time, the water flows out of the vessel at a constant speed equal to $0.6\sqrt{2g(H-x)}$.

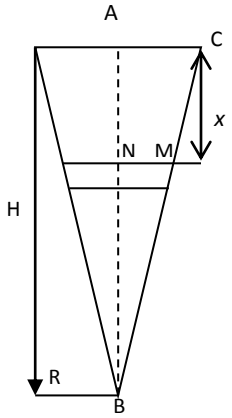


Figure 2. A cone-shaped vessel

We determine the volume of water that flowed during Δt through the hole in the bottom with the area, according to the formula $\Delta v = 0.6\pi r^2\sqrt{2g(H-x)\Delta t}$.

At the same time Δt the volume of water in the vessel will decrease with the height which must be equal to the volume of water that has flowed Δv.

From the equality $\Delta v = \Delta v1$, we obtain

$$\Delta t \approx \frac{y^2 dx}{0.6r^2\sqrt{2g(H-x)}} = dt$$

The time *T* of the total water drainage from the vessel is obtained by integrating *dt* after *x*, from *x = 0 to x = H:*

$$T = \frac{1}{0.6r^2\sqrt{2g}}\int_0^H \frac{y^2 dx}{\sqrt{H-x}}$$

Therefore, the mathematical model of the problem is constructed.

To calculate the integral we will express variable *y* through variable *x.* From the resemblance of the triangles ABC and NBM, we have for the vessel the following formulas:

$$\frac{H}{R}=\frac{H-x}{y}, \quad y=\frac{R}{H}(H-x); \qquad (5)$$

Substituting (5) in *T* we obtain:

$$T=\frac{1}{0.6r^2\sqrt{2g}}\int_0^H\frac{(\frac{R}{H}(H-x))^2dx}{\sqrt{H-x}} \qquad (6)$$

The student then selects one of the known methods (for example: the Trapezoidal rule, Simpson's rule, Milne rule) to compose the program and solve the integral of the formula (6). For example, for the values of *r = 0.5*, *R = 3*, *H = 10*, *g = 9.8*, applying the trapeze method to solving the integral of (6) gives the result:

$$T=\frac{1}{0.6\cdot 0.5^2\sqrt{2\cdot 9{,}8}}\cdot 11{,}3979013=17{,}165237$$

Note: The student must take into account the upper limit of integration. In case of relation (6), for obvious reasons, the upper limit is recommended to be H = 9,999. These aspects show the competences of using numerical methods to solve complex problems.

3. The stages of solving a problem in computer science

Thus, summarizing the above we can point the following steps to solve a informatics problems:

***Stage* 1. *Careful analysis of the conditions of the problem*.** At this stage, students are drawn to examine very carefully the conditions of the problem, including the dates of entry, the results to be obtained, the execution time, etc. Going through the statement of the problem several times is very important. Some aspects of the conditions of the problem can be noticed after careful analysis of all the conditions as a whole.

Stage* 2. *Development of the mathematical model. The mathematical description of the main links between the detected essential components requires the knowledge of the theoretical foundations of the various exact disciplines. Applying the appropriate mathematical apparatus to describe the processes examined in a strict and precise form remains the most difficult problem for students. The mathematics of the examined processes, in this regard, remains one of the most important problems in the process of preparing the students.

***Stage* 3. *Selecting the solution methods*.** After the mathematical model has been developed, it is necessary to decide on the solution methods. For example, if the mathematical model represents a system of linear equations, then one of the many existing methods for numerical solving can be selected. If no suitable method is found, in order to achieve the proposed objectives, it is necessary to modify one of the existing methods, or to develop a new method.

Stage* 4. *Development of algorithms. Before using the computer, the selected method must be displayed in the form of algorithms, using a programming language. It is known that different programming languages are oriented to solving problems of different types. In this case, the most suitable programming language is chosen for writing the respective program. In the context of those examined, this stage was discussed in more detail in section 2.

***Stage* 5. *Verifying the correctness of the proposed algorithms*.** The test data sets should be carefully designed so as to cover, as far as possible, all the variants of the algorithm execution, including exceptional situations, and to verify that each subproblem of the given problem is solved correctly (if possible, each program module will be tested separately). Testing may reveal omissions or errors in the design of the algorithms, but does not guarantee the correctness of the algorithm. For this, the algorithm should be tested on all possible sets of input data, which is practically impossible.

***Stage* 6. *Analysis of the complexity of the algorithm*.** In general, there are several algorithms for solving a given problem. In order to choose the best algorithm, these algorithms must be analyzed in order to determine their efficiency and, as far as possible, their optimality. Students are reminded that the efficiency of an algorithm is evaluated from two points of view:

1) from the point of view of the memory space needed to memorize the values of the variables involved in the algorithm;

2) from the point of view of the execution time. The complexity is analyzed above all when the algorithm is transposed into a programming language.

Conclusions

The interuniversity Olympiads in informatics organized within the Tiraspol State University, during 10 editions, are held for students from year I separately and students from years of studies II-IV separately. One of the central reasons, for which it is organized separately by years of study, relates to the level of student preparation and the subject matter studied in the various specialized disciplines. In this context students are proposed no less than 8 - 10 problems with different levels of complexity, so that the student can select the subjects on their own taste, from different compartments of the computer sciences, according to their own knowledge and skills, which could lead to solving the problem. In this way, the students involved in the development of contests in informatics - organized according to this format- gain confidence in their own strengths and subsequently tend to progress not only in the study of computer science but also in exact sciences [2-8].

References

1. Audrito G., Demo G., Giovannetti E. Olympiads in Informatics. nr.6, 2012. p. 3–20.
2. Bell T., Alexander J., Freeman I., Grimley M. Computer science unplugged: school students doing real computing without computers. In: Journal of Applied Computing and Information. 13(1), 2009. p. 20–29.
3. Mihălache L. Implementation of new didactic technologies in the educational process in computer science in high school. In: Pedagogical Universe Nr. 1, 2013. p. 69-75.
4. Chiriac E., Chiriac L. Development of mathematical models in ecology. The faculty of geography at 60 years. Proceedings of the symposium ”Development of geography in the Republic of Moldova. Chisinau: UST, 1998, p. 129-131.
5. Chiriac L., Mihălache L., Problems related to teaching/learning mathematical modeling. Scientific conference: ”Modernization of pre-university and university education in the context of European integration”, Materials of the scientific conference. Chisinau: UST, November 2009. p 306.
6. Chiriac L., Mihălache L. Methodical approaches regarding the exposure of the modeling process. The scientific conference: ”Modernization of pre-university and university education in the context of European integration”. The materials of the scientific conference. Chisinau: UST, November 2009. p 303.
7. Mihălache L. Methodical considerations regarding the exposure of the modeling process. Volume of the International Conference. Traditions. Values and perspectives in the education sciences, 4th edition, Cluj-Napoca: Science Book House, 2009. p. 267-269. ISSN 2065-006X.
8. Mihălache L. Methodical aspects regarding the elaboration of the tests in the compartment ”Numerical calculation” of the high school computer science course. International Conference Mathematics & Information Technologies: Research and Education (MITRE-2011), dedicated to the 65th anniversary of the Moldova State University. Chisinau: USM, August 22-25, 2011. p. 184-185.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 46-55 E-ISSN 2587-3636

CZU: 37.02.547 DOI: 10.36120/2587-3636.v18i4.46-55

EFFICIENT METHOD OF STUDYING THE INTERMEDIATE REACTIONS IN THE HALOGENATION PROCESS OF 2-METHYL-BUTANE IN THE COURSE OF ORGANIC CHEMISTRY

Eduard COROPCEANU[1], professor, PhD

Ion ARSENE[1,2], associate professor, PhD

Viorica ȘARGAROVSCHI[1,3], univ. lecturer, PhD, teacher of superior grade

Zinaida PURCEL[1], student

[1]Chair of Chemistry, Tiraspol State University, Chișinău, Republic of Moldova

[2]Institute of Chemistry of MECR, Chișinău, Republic of Moldova

[3]PITL "V. Vasilache", Chișinău, Republic of Moldova

Abstract. The contemporary teaching methodology is in a dynamic development due to the multiple socio-economic changes, but mainly due to the rapid evolution of the technologies, which are implemented in the training process with the purpose of solving specific problems. The integration of the methods specific to the field of Chemistry with the information technologies creates new opportunities to make the teaching methodology more efficient in order to ensure the compatibility of the formed competences with the needs of the labour market. The use of quantum calculations allows the understanding of chemical processes in terms of energy efficiency. To delve into the intricacies of the radical substitution mechanism in Organic Chemistry, modern computing software in the field of Chemistry such as Gamess, Gaussian, ChemCraft, ChemBioDraw etc can be used. The use of theoretical calculations offers the possibility to study some complicated phenomena of organic nature, which arguably strengthens the pupils/students' knowledge about the probability of the chemical reaction taking place in different classes of organic compounds.

Keywords: Organic Chemistry, chemical modelling, chlorination, bromination, energy stability, performance in education.

METODĂ EFICIENTĂ DE STUDIU A REACȚIILOR INTERMEDIARE ÎN PROCESUL DE HALOGENARE A 2-METIL-BUTANULUI ÎN CADRUL CURSULUI DE CHIMIE ORGANICĂ

Rezumat. Metodologia didactică contemporană este într-o dezvoltare dinamică grație multiplelor schimbări socio-economice, dar în primul rând datorită evoluției rapide a tehnologiilor, care sunt implementate în procesul de instruire cu scopul soluționării unor probleme specifice. Integrarea metodelor specifice domeniului Chimie cu tehnologiile informaționale creează noi oportunități în eficientizarea metodologiei didactice cu scopul asigurării compatibilității competențelor formate cu necesitățile pieței muncii. Utilizarea calculelor cuantice permite înțelegerea proceselor chimice din punct de vedere a rentabilității energetice. Pentru a pătrunde în subtilitățile mecanismului de substituție radicalică în chimia organică pot fi utilizate softurile moderne de calcul în domeniul Chimiei cum ar fi: Gamess, Gaussian, ChemCraft, ChemBioDraw etc. Utilizarea calculelor teoretice oferă posibilitatea de a studia unele fenomene complicate de natură organică, fapt care consolidează argumentat cunoștințele elevilor/studenților despre probabilitatea desfășurării reacției chimice în cadrul diferitor clase de compuși organici.

Cuvinte-cheie: chimie organică, modelare chimică, clorurare, bromurare, stabilitate energetică, performanță în educație.

Introduction

One of the important goals of the actual educational system (especially in the fields of exact and natural sciences) is the development of motivating training technologies. This process of training the specific competences of the subject is facilitated by the complex combination of the technologies from different fields (informational, industrial etc.). Interdisciplinary studies create additional opportunities for understanding complex phenomena. In the field of Chemistry it is necessary to realize the connection between the assimilated matter and the daily activity to make the pupil to understand the advantages of a conscious and interested training. At the same time, it is important for the educational process to be interspersed with scientific research to ensure a functional theory-application-research-innovation connection [1-4]. Training by research creates favourable premises for the emergence of interdisciplinary educational contexts [5]. For the pupils and the students in the field of Chemistry, first of all, the connections with Physics and Biology are needed to create an integral picture of the material world [6], and in the last decades ICT creates multiple opportunities that allow studies at the border between different disciplines, generating new research directions and useful tools for the teaching methodology. At the same time, the elaboration of the methodology based on combining the digital applications with the technologies specific to the disciplines allows the development of a creative and self-taught training style [7]. The emergence of additional elements in the teaching methodology opens new possibilities for teachers who, manifesting creativity and personal vision, can generate new educational contexts in which the educates can reach more valuable performances and a higher degree of motivation. Adapting information technologies to various disciplines in the field of Education Sciences allows the study of some phenomena that cannot be traced under ordinary conditions. The study of the energetic states allows to determine the stability, as well as the geometrical configuration of the molecular systems [8], and the analysis of some processes of substitution, condensation etc. delivers valuable information to draw conclusions about the occurrence likelihood of some phenomena [9,10]. The use of methods of calculating the energy of molecular systems is necessary to determine the probability of the existence of some molecules in one of the possible geometric configurations. These computational modellings allow the teacher to argue by individual calculations the fairness of theoretical statements for both inorganic and organic molecules [8]. Even more valuable is the level at which the pupils themselves can carry out these studies in order to formulate personal conclusions regarding the possibility of the occurrence of certain phenomena, the molecular architecture of some substances etc.

Applied methods and materials

Calculation of the optimized geometries and imaginary frequencies for reactants, intermediates, products and transition states was performed using the modern software

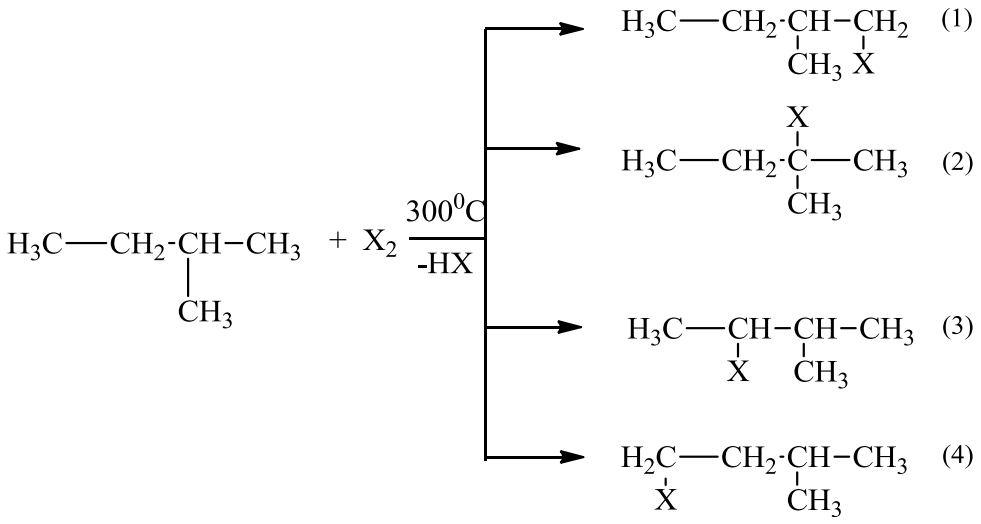
package GAUSSIAN 09 [11], using the standard base set 6-31G [12] for carbon and hydrogen atoms. The modelling results of the studied mechanisms (all possible substitution reactions) were obtained by applying ab initio quantum-chemical calculations. The quantum-chemical calculations performed for these systems allow a deep understanding of some aspects related to the energetic state, the spatial configuration and the molecular structure.

Results and discussions

Professional development based on the methodology focused on information technologies adapted to the specific of the discipline allows the formation of the research competence. As a result, personalities with their own style of training, self-taught, able to solve various problems through unique methods can be developed. The formation of the competence to use specialized software to solve the specific problems of Chemistry even at the pre-university level creates premises for the autonomous development of the pupil's personality in an interdisciplinary educational context.

As many processes could be carried out according to different scenarios depending on the energy of the reactions, external factors that can influence the direction of the flow etc, for the Chemistry teacher it is important to have, besides the theoretical material, practical arguments based on the calculations regarding the energy profitability for carrying out the chemical reaction. As an example we will examine the possible halogenation reactions of an organic compound containing different types of carbon atoms – 2-methyl-butane – to determine on the basis of calculations which of the examined species has a higher probability of formation. The free radicals, being particles with very high reactivity and a very short lifespan, in their stabilization tendency combine two by two or participate in reactions with other substances present in the system. Thus, the radicals are initiators of radical mechanism reactions.

As object of this study is proposed the radical substitution process for the 2-methyl-butane compound (substitution step of the first hydrogen atom):



The halogenation reaction of 2-methyl-butane is a chain and proceeds in a few steps: initiation, propagation and interruption of the reaction. In the first step, under the influence of light or temperature, it is obtained the halogen radical which in the second step reacts with the 2-methyl-butane molecule forming hydrogen bromide or chloride and free hydrocarbon radicals:

1. initiation of the reaction:

$$X\bullet \,|\, \bullet X \xrightarrow{h\nu} X^{\bullet} + X^{\bullet}$$

2. propagation of the reaction:

$$RH + X^{\bullet} \rightarrow R^{\bullet} + HX$$

$$R^{\bullet} + X_2 \rightarrow R\text{-}X + X^{\bullet}$$

3. interruption of the reaction:

$$X^{\bullet} + {}^{\bullet}X \rightarrow X_2$$

$$R^{\bullet} + X^{\bullet} \rightarrow R\text{-}X$$

$$R^{\bullet} + {}^{\bullet}R \rightarrow R\text{-}R$$

In the given case at the first halogenation step 4 different carbon atoms can be substituted and 4 radicals can be obtained. In turn, the free radicals can react with another halogen radical forming the halides: 1-X-3-methyl-butane, 2-X-3-methyl-butane, 2-X-2-methyl-butane, 1- X-2-methyl-butane.

Theoretical study of the species participating in the halogenation reaction.

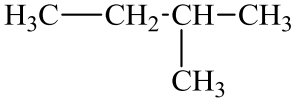
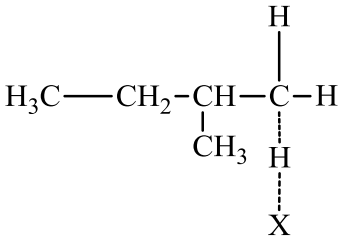
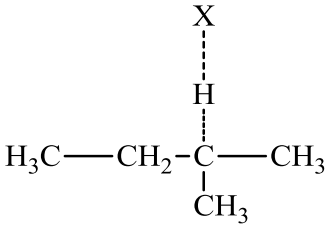
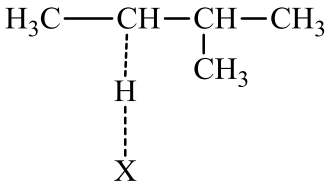
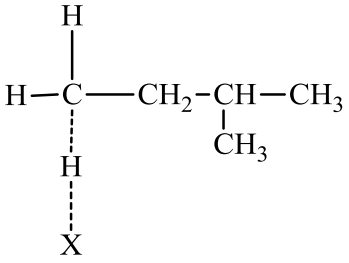
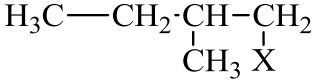
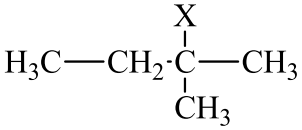
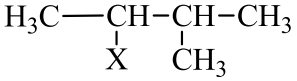
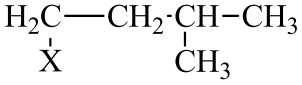
The geometric structure of the reactants (R), the transition states (TS) and the reaction products (P) were studied *ab initio* using the SCF method in the UHF approximation, using base 6-31n (UHF/6-31G) for atomic functions [12]. All calculations were performed using GAUSSIAN 09 [11], which is a modern software package used to investigate the structural properties, those determined by the electronic structure of molecules or complex molecular systems.

As models it has been studied the halogenation reactions of 2-methyl-butane that take place in the radical substitution process, which occur after the radical chain reaction mechanism. The energy status of the investigated chemical systems and the energy profile of the intermediate reactions were studied.

Using some theoretical calculation methods, a series of geometric and energetic parameters related to molecular structure, as well as the energy profile of some chemical processes, can be correlated and predicted. Using this model, calculations were performed for all species involved in this radical substitution reaction.

Initially, the values of total energies of the structures of the reactants and reaction products were calculated (Table 1).

Table 1. Total energy values of the studied species during halogenation reaction

No.	Species name	Geometric structure	E_{tot}	
			Cl	Br
1.	2- methyl-butane		-196.2522	
2.	TS1		-655.6561	-2765.9551
3.	TS 2		-655.6670	-2765.9658
4.	TS 3		-655.6623	-2765.9616
5.	TS 4		-655.6568	-2765.9553
6.	1-X-2- methyl-butane		-655.1346	-2765.9582
7.	2-X-2- methyl-butane		-655.1421	-2765.9672
8.	2-X-3- methyl-butane		-655.1384	-2765.9642
9.	1-X-3- methyl-butane		-655.1351	-2765.9594

For each particle from the mentioned reactions the optimum geometry was determined and the total energies were calculated.

Radical substitution mechanism

In the Chemistry discipline, it can be proposed several models that would allow for complex studies of molecular characteristics and chemical mechanisms. It is proposed the analysis of a model of training-research integrated application, in which the mechanism of substitution in the organic molecules is studied with the help of quantum-chemical calculations based on specialized software, which allow to determine the energetics of the investigated mechanism and the probability of the chemical reactions.

Based on this, we set out to calculate all the species involved in the reaction (reactants, reaction products, intermediates and transition states) and based on these calculations to obtain the energy profile of the radical reaction.

The transition states were localized and verified by vibrational analysis. For the intermediate species, an imaginary frequency was obtained, which demonstrates the presence of these activated states of the investigated systems. The values of the imaginary frequencies are described in the text and in the figures presented below. For all the studied molecules it was considered that the spatial nuclear configuration of the studied molecules corresponds to the C_1 symmetry group.

At the first halogenation step, which is also the initiation reaction that proceeds under the action of light or temperature, the homolytic cleavage of the halogen molecule takes place with the formation of two free radicals:

$$X_2 \overset{h\nu}{\rightarrow} 2\dot{X}, \quad X = Cl, Br$$

From an energy point of view this reaction is thermodynamically convenient (exothermic), because for the chlorine molecule ΔE=7.81 kcal/mol, and for the bromine molecule ΔE=7.72 kcal/mol.

In the case of the 2-methyl-butane chlorination reaction, which proceeds according to the general reaction: R-H + 2Cl$^{\bullet}$ = R-Cl + HCl, four substitution mechanisms were analyzed. Figure 1 shows the energy profiles of the reaction mechanisms of processes 1-4. The highest activation energy for the chlorination process of the studied reactions is owned by mechanism (1) with value 24.60 kcal/mol, while the lowest value being in reaction (2) with value 17.76 kcal/mol, a fact from which we can says that reaction (2) would proceed with a higher reaction rate. Studying the energy profile of mechanisms 1-4 as a whole, it was confirmed that these are exothermic reactions and the most energetically convenient one is reaction (2) with an energy gain equal to -25.79 kcal/mol. It can also be concluded that 2-chloro-2-methyl-butane compound is the most energetically stable, which is also known in the specialized literature.

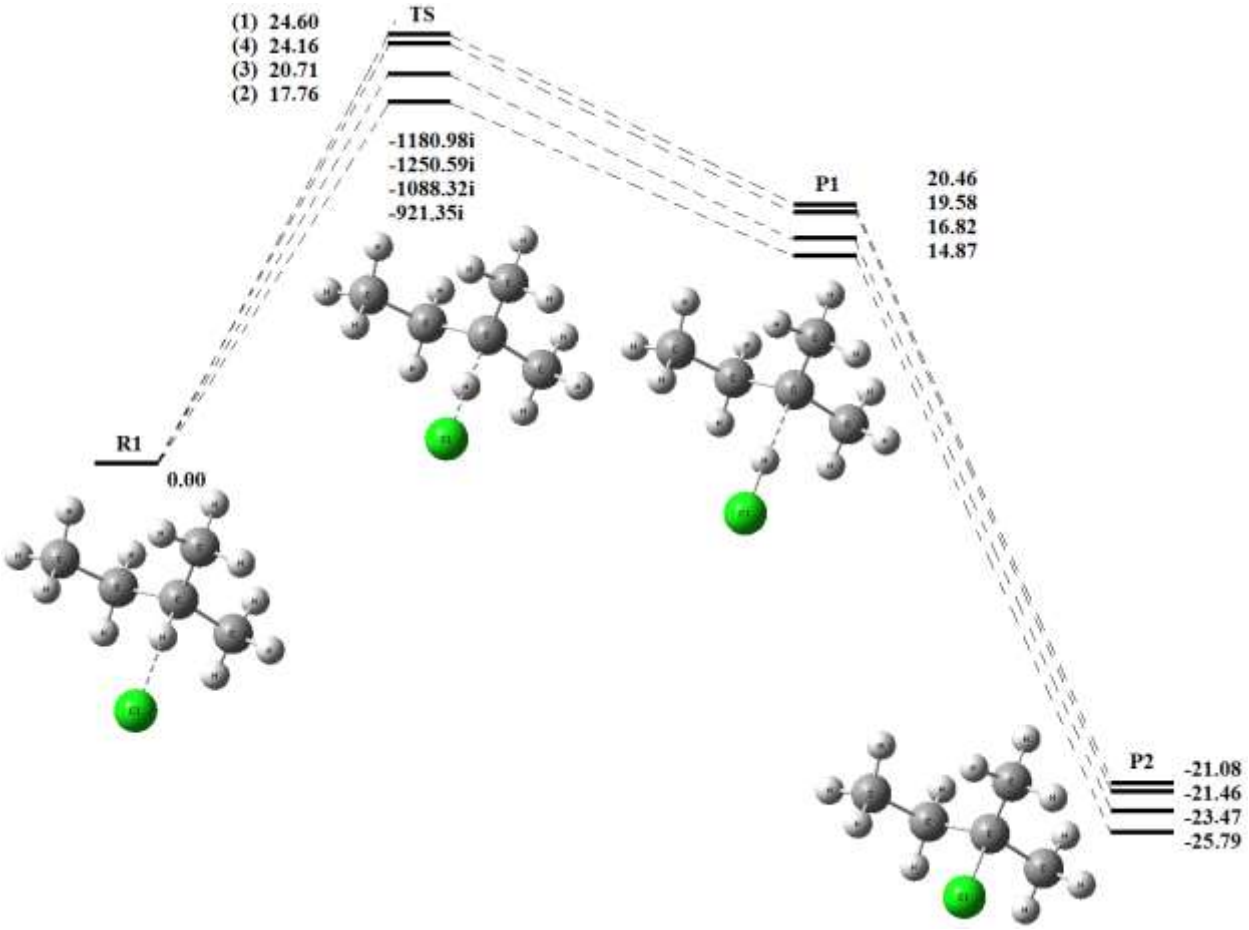


Figure 1. Energy profile of chlorination reactions 1-4 (kcal/mol)

The configuration of the transition states presented for reactions 1-4 in the figure above is characterized by the breakdown of the hydrogen atom from the respective carbon atom of the studied hydrocarbon and has a single imaginary frequency with value -1180.98i cm^{-1}, -921,35i cm^{-1}, -1088.32i cm^{-1}, -1250.59i cm^{-1}, corresponding to mechanisms 1-4 which show the movement of the hydrogen atom between the chlorine and carbon atoms. The intermolecular transfer of hydrogen in the studied mechanism takes place according to the reaction mechanism described above.

The own vector of the transition state, associated with the unique imaginary frequency, represents an adequate structure of the transition state with the migration of the hydrogen atom from the carbon atom to the chlorine one. Also, in this context we can say that the analyzed organic isomer is more energetically stable in form 2-chloro-2-methyl-butane.

In the case of bromination, which proceeds according to the general reaction: R- H +2Br$^{\bullet}$=R-Br+HBr, also based on the obtained energies, the graph of the energy profile of the reaction was elaborated according to Figure 2.

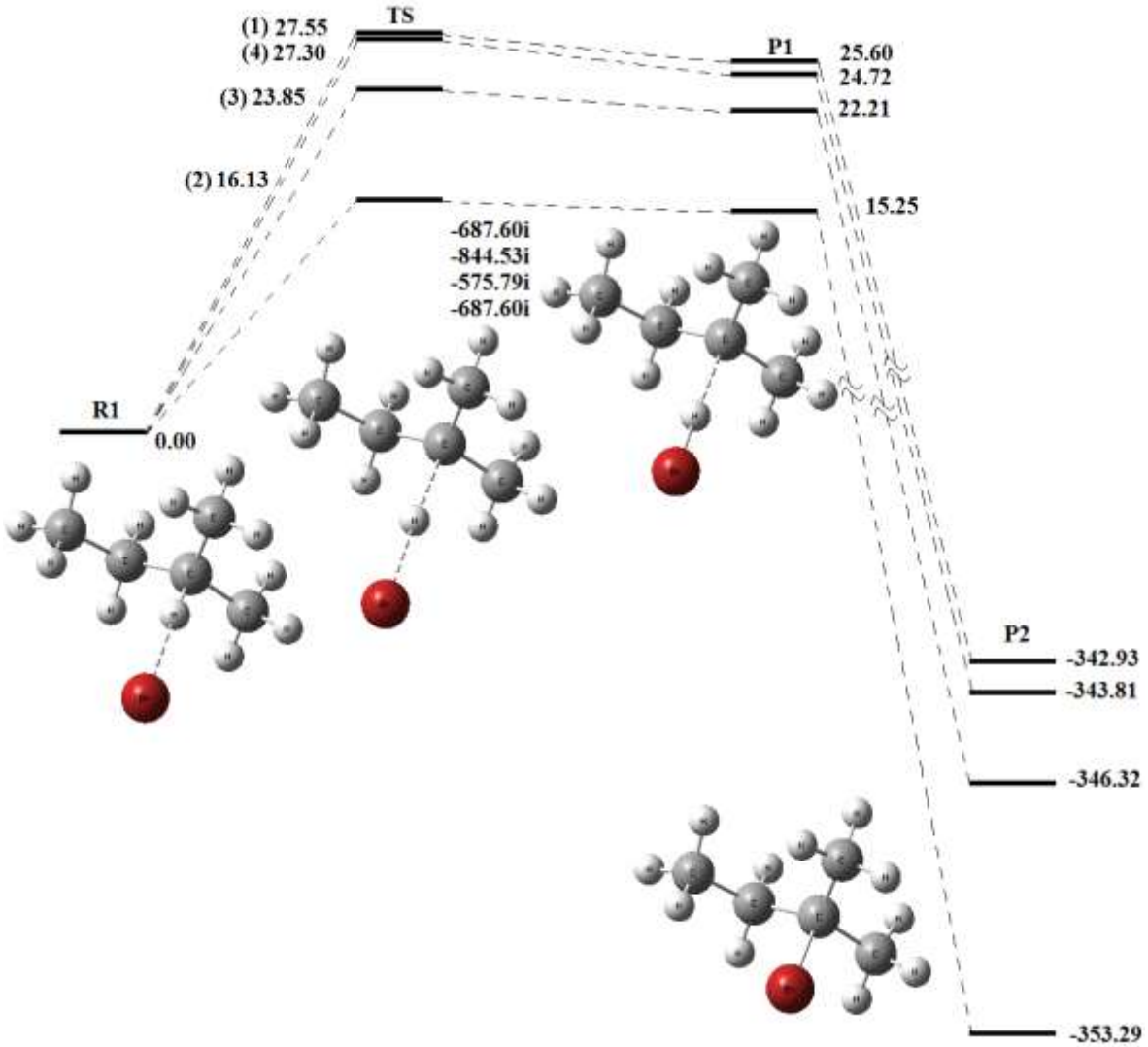


Figure 2. Energy profile of bromination reactions 1-4 (kcal/mol)

At first the first Br• radical forms with the saturated hydrocarbon a stable R-H-Br• complex (R1, Figure 2). There were located the transition states for this step which are characterized by a single imaginary frequency -687.60i cm^{-1}, -379.97i cm^{-1}, -575.79i cm^{-1}, -844.53i cm^{-1} corresponding to reactions 1-4 (TS complex in Figure 2). The transition vector indicates that the molecular motion of this frequency is dominated by the transfer of a hydrogen atom H from C to Br. It can be seen that the geometry of the transition state differs significantly from that of the reactants. The result of the first step is the formation of the free radical R• and the HBr molecule.

In the next step, that is considered to be the step of interrupting the bromination reaction, the combination of two hydrocarbon and bromine radicals takes place. It is interesting that this reaction takes place without any activation energy by obtaining stable and neutral halogenderivat.

In order to acquire the new applications that allow to perform quantum-chemical calculations in the study programmes Chemistry, Chemistry and Biology at cycle I, students are trained in the courses Information Technology in Chemistry and

Computational Modelling in Chemistry etc., and at cycle II in Computational Chemistry and Quantum Chemistry. As a result of the evolution of the competence of using information technologies in the field, the students begin to develop various graphic representations in Chemistry and related disciplines, PPT presentations, to solve tasks within the individual work, to process materials for conferences and articles in scientific journals. Some of the students are included in student research projects (eg: Quantum-Chemical Study of the Energy Stability of Some Molecules and the Energy Profile of the Mechanisms of Chemical Reactions). All students use these applications in the elaboration of their bachelor and master degree theses, and up to 20% of the graduates elaborate theses on scientific topics in the field of quantum-chemical calculations. These acquisitions are very useful in the professional activity.

Teachers in pre-university education can use these applications on different topics: the spatial structure of inorganic and organic molecules; determination of the geometrical parameters of the molecules (bond length, bond angle, etc.); probability of the existence of a molecule in a certain isomeric configuration; energy stability of some species and molecular systems; strength of the chemical bond; electrolytic dissociation etc. In university education these applications can be used for the topics: the determination of the activation energy and the transition state; study of the mechanisms of some chemical reactions; caloric effects of thermodynamic processes; charges on the atom; HOMO LUMO etc. During the continuous training courses and additional professional qualification the teachers are trained how to use these tools to carry out interdisciplinary activities in Chemistry.

These things allow us to state that the use of calculation methods based on modern information technologies ensures the creation of an interdisciplinary educational context, which allows the development of strong skills and the readiness for solving different problems of specific character in the field of Chemistry. These practical examples allow the motivation for a conscious and deep training, the development of some individual aspects of the specialist and the self-taught capacity of development, marking positively the trajectory of formation of the professional competence.

Conclusions

In the context of rapid evolution of information technologies it is necessary to develop digital competence and acquire specific applications in specific fields, which increase pupils' level of motivation for the educational process and broaden the spectrum of available tools for solving problems with different aspect and degree of difficulty. The use of theoretical calculations to determine the energy of some chemical processes allows to determine the probability of their occurrence. These computerized calculations allow pupils and students to individually identify the energy profile of some processes in Organic Chemistry. The proposed methodology aims to provide the school and university teachers

with real mechanisms for the substantiated demonstration of the occurrence direction of some reactions in Organic Chemistry. The method is recommended for use at the university level, while for the pre-university level it may be optional for the students who show interest or are preparing for various competitions in the field.

It has been established that students/students who use a large set of information technologies in the process of studying Chemistry achieve high performance in the following stages of study and employment in the field of work.

Bibliography

1. Bocoş M. Teoria şi practica cercetării pedagogice. Cluj-Napoca: Editura Casa Cărţii de Ştiinţă, 2003.
2. Enăchescu C. Tratat de teoria cercetării științifice. Iași: Polirom, 2007.
3. Ionescu M., Bocoş M. Cercetarea pedagogică şi inovaţia în învăţământ. In: Pedagogie. Suporturi pentru formarea profesorilor. Cluj-Napoca: Presa Universitară Clujeană, 2001.
4. Coropceanu E. Dezvoltarea sistemului de competențe cercetare-inovare-antreprenoriat în context interdisciplinar. In: Congresul Ştiinţific Internaţional Polono-Moldo-Român: Educaţie – Politică – Societate. Chișinău-Iași, 1-4 aprilie 2019. n. 1. p. 19-25.
5. Coropceanu E., Nedbaliuc R., Nedbaliuc B. Motivarea pentru instruire: Biologie şi chimie. Chişinău: UST. 2011.
6. Heisenberg W. Imaginea naturii în fizica contemporană. Bucuerşti: All. 2001.
7. Coropceanu E., Rija A., Arsene I., Putină M. Dezvoltarea abilităților de autoformare la chimie în baza unor tehnologii informaționale. Studia universitatis moldaviae. Seria Științe ale educației. 2014. nr. 9(79). p. 92-98.
8. Codreanu S., Arsene I., Coropceanu E. Utilizarea unor modalități moderne de calcule cuanto-chimice a stării energiei sistemelor moleculare în cursul de chimie. Acta et Commentationes. Științe ale Educației. 2017. nr. 1. p. 147-156.
9. Codreanu S., Arsene I., Coropceanu E. Theoretical study of some phenomena and processes in the course of organic chemistry. Annales Universitatis Paedagogicae Cracoviensis. Studia ad Didacticam Biologiae Pertinentia. 2018. 8. p. 151-159.
10. Coropceanu E., Arsene I., Șargarovschi V., Purcel Z. Studiul instabilității unor izomeri ai alcoolilor nesaturați și a reacțiilor intermediare în procesul transformării tautomerice în cadrul cursului de chimie organică. Acta et Commentationes. Științe ale Educației. 2019. nr. 2. p. 32-42.
11. Frisch M.J., Trucks G.W., Schlegel H.B. et al. Gaussian 09, Revision B.01. Gaussian. Inc.: Wallingford. CT. 2009.
12. Hehre W.J., Radom L., Schleyer P.V.R., Pople J.A. Ab Initio Molecular Orbital Theory. Wiley: New York, 1986.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 56-61 E-ISSN 2587-3636

CZU: 372.8004 DOI: 10.36120/2587-3636.v18i4.56-61

COMPUTER SCIENCE PERFORMANCE - RESOURCES, PARADIGMS, IMPACT

Emanuela CERCHEZ*

”Emil Racoviţă” National College, Iaşi, România

*Author of textbooks and collections of problems in computer science; cofounder of the **.campion** educational archive; 8 years Team leader of the national team of Romania at IOI (2001-2008); coordinator of the national lot of Romania in computer science (2004-2008); teacher didactic grade 1.

Summary. The article identifies the ways in which the Olympiads and competitions influence the quality of computer science education. Two terms are defined: *Olympic* and *performance*. The differences and analogies between "competition" and "class" problems, "competition" evaluation versus "class" evaluation are indicated. Also, the auxiliary tools of the training program for the programming competitions are analyzed, such as: online performance training platforms (.camp, infoarena etc.). Two paradigms of performance training are analyzed: the segregated model, the integrated model. The article also deals with software creativity contests in which students present a complex application, developed over time. Such contests do not exclude algorithms, because any program involves the implementation of an algorithm.

Key words: Olympic students, performance, evaluation, competition, solving competition problems, paradigms of performance training.

INFORMATICA DE PERFORMANȚĂ RESURSE, PARADIGME, IMPACT

Rezumat. În articol sunt identificate modurile prin care olimpiadele şi concursurile influenţează calitatea învăţământului la informatică. Se definesc doi termeni: *olimpic* şi *performanţă*. Sunt indicate diferențele și analogiile dintre problemele ”de concurs” și probleme ”de clasă”, evaluare ”de concurs” versus evaluare ”de clasă”. De asemenea, sunt analizate instrumentele auxiliare ale programului de pregătire pentru concursurile de programare cum ar fi: platformele online de pregătire de performanţă (.campion, infoarena etc.). Se analizează două paradigme ale pregătirii de performanţă: modelul segregat, modelul integrat. Sunt tratate în articol și concursurile de creativitate software în care elevii prezintă o aplicaţie complexă, dezvoltată în timp. Astfel de concursuri nu exclud algoritmica, deoarece orice program presupune implementarea unui algoritm.

Cuvinte cheie: studenți olimpici, performanță, evaluare, competiție, rezolvarea problemelor de competitivitate, paradigme ale învățământului de performanță.

Introduction

Many times I have heard, in various contexts, the following statement: ”*The results of the students at the Olympic educational competitions are irrelevant, they are not a measure of the quality of an educational system. Only* 0*% of the students are Olympic*”. In what follows I will try to identify ways in which the Olympic competitions and ordinary competitions influence the quality of Computer Science education.

First of all, I will define two terms, which will appear during this process: *Olympic* and *performance:*

- Olympic is a pupil who participates in the Olympic competitions and school competitions, according to his choice, preparing to standards corresponding to the competition. In other words, I do not define the *Olympic* by the results obtained in competitions, but by his educational career and his attitude. He chooses to be competitive (not the teachers / parents), he is motivated to study a certain field to performance standards.
- Through Computer Science performance I understand the ability of a pupil to build algorithmic solutions to new problems and to implement them efficiently using appropriate technologies. Again, it should be noted that I do not measure performance by the number of diplomas obtained in competitions, but by creativity, innovation and (here's what differentiates Computer Science from other fields!) the ability to transpose the solution into a functional and efficient programme.

”Competition” issues versus ”Class” issues

The first aspect that I want to highlight is that the Olympic and school competitions are the prerequisites for the formation of high performance computer scientists. Not just because it offers the competitive framework (to be an Olympian you must participate!), but because it offers relevant educational resources for a performance training. The problems created by the scientific commissions of the competitions represent the best tool for the formation of Olympians. These differ from the problems currently used in the classroom for introducing new concepts. First of all, the formulation is different, a competition problem will always be formulated in a context (real or imaginary). The pupil must analyze the context of the problem, identify the relevant elements and develop the algorithmic model for the given problem. Extremely important skills for a computer scientist! In the course of a career, a computer scientist will never meet a client who will say: "I have a vector with *n* elements, sort it!". The client will come with a story, he has cows, each cow has its own personality, which he will describe in detail, etc. From a complex story, the computer scientist must extract the relevant data for the client's application and then algorithmically model the reality. Solving competition problems forms such skills.

Problems formulated in the following way are frequently used in the classroom:

1. Give a vector with *n* elements, order it in ascending order.
2. Give a vector with *n* elements, order it descending.
3. Give a vector with *n* elements, sort the first *k* elements in ascending order and the last *n-k* elements in descending order.
4. etc.

It can be seen that the problem is "out of context" and the work tasks are repetitive. Competition problems, besides the context, have another feature: they are not repetitive. Each problem is a new problem (of course, on different levels of difficulty and complexity). A pupil working on competition problems will not apply mechanically,

repetitively an assimilated algorithm, but will create new, innovative algorithmic solutions.

Furthermore, competition issues contain restrictions on the volume of data and restrictions on available memory and runtime. Again, it is very important for a computer scientist, because the pupil must analyze the complexity of the algorithms he builds, both in terms of runtime and memory space and decide which algorithm is more efficient.

By solving problems such as those commonly formulated in the classroom, the pupil demonstrates that he has acquired a certain algorithm and can apply it. Specifically, we can achieve objectives from the first 3 levels of Bloom's taxonomy (knowledge, understanding, application). Solving competition problems, the pupil will form competences that correspond to all the 6 taxonomic levels [1].

”Competition” evaluation versus ”Class” evaluation

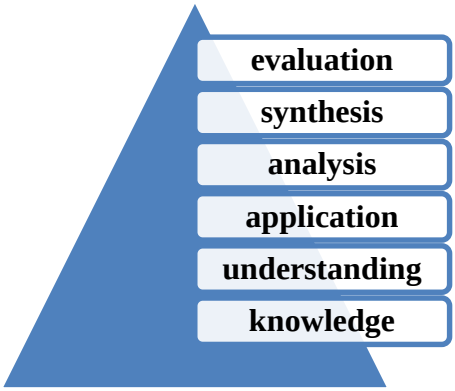


Figure 1. Bloom's taxonomy

The assessment that we will call "class assessment" is consistent with that of the pupils facing the baccalaureate exam. The competences to which this type of assessment aims rarely exceed the first 3 taxonomic levels. The pupil evaluates expressions, applies an algorithm step by step on a specified data set, identifies data sets on which the algorithm provides a specified result, implements a given algorithm, completes or writes small code sequences that target elementary data processing / data structures. data or write a function / programme for such processing. The evaluation scale is formulated so that each constituent of the code is scored. Even if the pupil's programme does not solve even the given problem, only by disparate elements of the programme, scored according to the scale, the pupil can obtain 6-7 points out of 10.

I did not intend here to make an analysis of this way of evaluation, but only to put it in the mirror with the evaluation at the Computer Science competitions. In competitions, the pupil not only builds an algorithm, but also implements it in a programming language. The pupil's programme is run on a variable number of test data sets, and the results provided by the programme are verified with an automatic evaluation system, more precisely by a programme written by the author of the problem for this purpose.

Thus, the first condition of the pupil's programme to receive points is that the programme should be functional, the number of points received being proportional to the efficiency of the programme. The evaluation in this case mainly concerns competences from the top of Bloom's taxonomy. The pupil builds different algorithmic solutions, analyzes them from the point of view of complexity, chooses the optimal solution, and then implements this solution. In order that the programme may work, the algorithmic

solution must be logically correct, produce correct results on at least some types of test data, and be properly implemented. Let's remember an old, but completely true, saying: "a programme that almost works is like a plane that is almost flying." This is the starting point in the competitive evaluation and in this way we practically build a working style: the pupil knows that he does not receive points for "code breaks", that a program must be completed in its entirety, that it must be checked on tests with a varied and graded structure, that it must be optimized in accordance with the restrictions of the problem. This style of programming, which is learned step by step, problem by problem, will become a philosophy of life

Regardless of the system, the evaluation is the mechanism by which we regulate the good functioning of the system. The performance standards that we use for the evaluation will be crucial for the quality of education, because both pupils and teachers build their educational approach in relation to these standards.

Online performance training platforms

Over time, various platforms have been developed for preparing pupils' performance in Computer Science, which mainly contain problems with automatic evaluation of pupils solutions, but also other resources, such as specialized articles, educational software, etc. Such platforms exist in all the countries with a tradition in Computer Science, but I will mention the 3 most important ones existing in Romania:

1. The **.campion** educational archive (campion.edu.ro/arhiva) - initiated by Prof. Emanuela Cerchez and Prof. Marinel Şerban, was launched as an auxiliary tool of the .campion performance training programme (2002-2012); the archive contains over 1500 problems with online evaluation, articles and educational software;
2. **Infoarena (**infoarena.ro) - project started by a group of enthusiastic students (Cristian Strat, Silviu Gănceanu, Mircea Paşoi and Leonard Crestez), who promote excellence in programming by organizing high level competitions and writing educational articles;
3. **Pbinfo** (pbinfo.ro) - a platform created by Prof. Silviu Candale, which contains numerous problems with online assessment, structured on the topics of the school syllabus, with difficulty level adapted to the class hours, but also competition problems.

The resources created by the scientific commissions of the school competitions and the olympic competitions represent a significant part of the educational content of these platforms.

Paradigms of performance training

I think we can conclude that educational resources exist. How do we use them? I will put in antithesis two paradigms of performance training.

1. The s*egregated* model: we identify the pupils capable of performance and carry out their training separately from the rest of the class.

Advantages: the way of preparation being customized for a very small number of pupils, good results can be obtained in short time in competitions.

Disadvantages: The pupils involved are isolated from the rest of the students in the class, a fact which generates a disadvantage both at the class level (the level of preparation of the class is not positively influenced, sometimes even on the contrary) and for the selected pupils (even if the level of their preparation grows at a more alert rate, this is done to the detriment of the development of their social, communication and collaboration skills).

2. The *integrated* model: the teacher does not select the pupils capable of performance, but works with the whole class pursuing educational objectives of high taxonomic level.

For this purpose, the use of competition problems and of the automatic evaluation of problem solutions is strictly necessary. Each problem solved, each problem proposed to be solved must be selected so as to bring a cognitive benefit to the pupils, to challenge them in discovering a solution. The automatic assessment will provide the pupil with an immediate feedback on his solution. He will be able to use this feed-back and will resume solving the problem whenever necessary, until he reaches the maximum score. The initial dramas generated by multiple evaluations of 0 points will turn into frenetic joy when, after identifying the errors and correcting them, the pupil will get 100 points. And every 100 points earned will still be a motivation to continue, to progress.

The pupils capable of performance are not selected by the teacher, but will naturally self-select. The evolution of each student will be natural, according to their own potential, the pupils will not be subjected to doping. Most of the pupils in the class will choose to participate in the olympic competition (because it will consider the competition stimulating and motivating). Of course, from one competition stage to another, the number of participants will be reduced, so that at the national stage the number of participants will naturally be small. But behind each national olympic pupil, there are, in a pyramid, numerous pupils with whom he "ran" together. And, surprisingly, even if they participated in the same competitions, with more or less success, no antagonistic feelings appear, in the classroom there is created emulation, collaboration, constructive communication between people with common passions

Advantages: In the long run, most pupils in the class will progress, achieving, at different levels, high-level educational goals. In this context, the results of the pupilss at the competitions are a relevant indicator of the quality of the education in the school.

Disadvantages: If at the beginning the teacher can work with all the pupils of the class, during their evolution it will be different and the teacher will have to individualize the learning process, using tasks from the area of the next development of each pupil / group of pupils. This becomes all the more difficult as the number of pupils in the class is higher. Even so, the progress is slower because the teacher has to make sure that all the pupils achieve the teaching goals (of course, at different levels of performance).

Software creativity contests

Although until now I have referred strictly to algorithmic competitions, I do not

want to exclude the contests in which the pupils present a complex application developed over a long period of time. In these competitions, the emphasis is placed mainly on the technologies used, on the software engineering, on the functionality of the product related to its stated purpose and not least the utility and originality of the application. Obviously, the algorithm is not excluded, any programme involves the implementation of an algorithm. These applications are usually developed in a team and develop communication and management skills of teamwork (adapting to different roles and responsibilities, productive collaboration with others, empathetic behavior, respecting other points of view) that few algorithmic competitions target. The pupils study new technologies, suitable for the implementation of the application, thus becoming independent in the learning process, able to adapt to the new things (very important ability in such a dynamic field). Starting from the purpose of the application, the pupils have to identify the problems that they have to solve, to find solutions, to plan the project stages, to constantly evaluate the project execution stage The complexity of the application requires the development of the project management, time management skills, self-assessment, as well as skills for public presentation of the final product.

Identifying, formulating and solving problems; responsibility and ability to adapt; communication skills; creativity and intellectual curiosity; critical thinking and systemic thinking; collaboration and interpersonal skills; demonstrating teamwork and leadership skills; self-training; social responsibility: all the essential competences for a person who will live in the 21st century are formed during the development of such an application. The participation in such competitions is therefore essential for becoming successful in the field of Computer Science [2].

Conclusion

This article reflects the didactic experience accumulated in 30 years of activity in the school, 18 years of activity at the Iaşi Center of Excellence and 23 years of participation in scientific commissions in competitions and school olympic competitions. In all these years, we have used the olympic and the school competitions as a tool for achieving the performance in Computer Science. Each year, other olympic pupils demonstrate that the method works, through their results in school, but especially in after-school life. As Aristotle said, excellence is a custom.

References

1. Eșanu A., Haut C. Obiective de învățare pentru secolul XXI. București: CEAE, 2016. ISBN: 978-973-0-21173-3. On line: http://ceae.ro/wp-content/uploads/2016/04/Raport_Competente.pdf
2. Parteneriat pentru competenţele secolului XXI: www.21stcenturyskills.org

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 62-72 E-ISSN 2587-3636

CZU: 004.02 DOI: 10.36120/2587-3636.v18i4.62-72

BITWISE OPERATORS IN C LANGUAGE
THE EXPLANATION, IN DETAIL, OF THEIR USE. LITTLE TRICKS

Marius NICOLI, ”Frații Buzești” National College, Craiova, România
*Coordinator of the National Junior Lottery; Team Leader of the national team at several international competitions; coordinator of the Scientific Commission at the Balkan Youth Olympiad (2018); teacher didactic grade 1.

Summary. The programming languages provide a number of operators, primarily for the usual mathematical operations. Usually they have associated instructions that the microprocessor can execute directly. The bitwise operators also correspond to instructions that the microprocessor has in its set, even if they do not necessarily correspond to widely known operations. Thus, using them to perform certain tasks makes the programs very fast, especially if they are used in their critical sequences. In this material we will present the syntax and the effect of these operators in the C / C ++ language and we will present relevant situations where certain problems can be solved elegantly and by code that is executed quickly, with the help of bitwise operators.

Keywords. Programming languages, bitwise operators, the solution of the problem, instructions.

OPERATORI PE BIȚI ÎN LIMBAJUL C.
EXPLICAREA ÎN DETALIU A MODULUI DE UTILIZARE. MICI TRUCURI

Rezumat. Limbajele de programare oferă o serie de operatori, în principal pentru operațiile matematice obișnuite. De obicei, au instrucțiuni asociate pe care microprocesorul le poate executa direct. Operatorii pe biți corespund, de asemenea, instrucțiunilor pe care microprocesorul le are în set, chiar dacă nu corespund neapărat cu operații cunoscute. Astfel, utilizarea lor pentru a îndeplini anumite sarcini face ca programele să fie foarte rapide. În acest articol este prezentată sintaxa și efectul acestor operatori în limbajul C / C ++ și elucidate situații relevante în care anumite probleme pot fi rezolvate în mod elegant și printr-un cod care este executat rapid, cu ajutorul operatorilor pe biți.

Cuvinte cheie. limbaje de programare, operatori pe biți, soluția problemei, instrucțiuni.

Introduction

They are operators that apply to data of whole types. In order to understand the effect of their application, we must master the internal representation of integers in the reserved memory area. To be simpler to write the examples, we will consider integer variables on 2 bytes (16 bits), which we will present being valid for the other types of whole data. The first rule is that positive values are stored in their reserved bits according to the writing of the value in base 2. Exemple:

```
short a;
a = 43;
```

The variable a will occupy 16 bits and following the assignment, their configuration will be the following:

Bit position	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit value	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1

According to the writing rule in base 10 we have $2^5 + 2^3 + 2^1 + 2^0 = 43$. A short variable type we know it may range between -2^{15} and $2^{15} - 1$ (there are a total of 2^{16} values, that is, for each configuration of 0 and 1 of length 16, one value in this range). Therefore, the maximum positive value is $2^{15} - 1$. But this number is equal to the value of the expression: $2^{14} + 2^{13} + \ldots + 2^1 + 2^0$. This is a known thing in mathematics, but we will see that there are more interpretations in informatics that justify this result. We deduce that for the maximum value, all bits, except the first one, have the value 1. So, the internal representation of 32767 is:

0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

We deduce automatically: the other positive values have as internal representations configurations that can have 0 values instead of the existing 1 values. The conclusion is as follows: all configurations that have the first bit 0 correspond to positive values. If our variable were of the unsigned short type, the value 1 for the first bit (the front bit, bit 15 or, more often said, the most significant bit) would have represented positive values between 2^{15} and $2^{16} - 1$. That's because at the positive values of the short 2^{15} is added (because it is the 1 the most significant bit).

For the `short` type (and the other integer types that are not written `unsigned`), the configurations that have the most significant bit equal to `1` are internal representations of negative numbers.

We will present below the rule after which the module is deduced when the configuration stores a negative value. If we thought hastily, we could say that the presence of 1 in front means a negative number and the conversion into the base 10 of the configuration with the other bits is the absolute value. But it is not quite so. Here is a simple justification, for example:

The number 1 stored in a short variable has the representation:

```
0000000000000001
```

Number -1, as in the above reasoning would be:

```
1000000000000001
```

The addition of the two values should be 0. But it is obvious that this is not the case.

```
0000000000000001 +
1000000000000001 =
1000000000000010
```

Here is the correct rule of finding the absolute value at the negative numbers (called the rule of the complement against 2): all the bits are complemented (the values 1 are replaced by 0 and the 0 are replaced by 1), then they are added to the value 1 (based on 2).

Here is an example:

Internal representation	1111111100010100
After complementary	0000000011101011
After adding 1	0000000011101100 (bits of 1 at the end become 0 and the first 0 becomes 1, the remaining values of the bits are kept).

Now, writing in base 10 the obtained number (0000000011101100) we have $2^7 + 2^6 + 2^5 + 2^3 + 2^2 = 236$.

So the initial sequence, that is 1111111100010100, represents the internal representation of the number -236.

Here's a sample:

-236 +	1111111100010100 +
236	0000000011101100
0	10000000000000000

It is observed that the last 16 bits reach all 0, and the front bit, 1 (given by transport), is outside the allocated memory area, so the result is 0.

Remarks:

- the value -1 will be represented by all bits 1 (if we add the value 1 the transport always propagates and the value 0 is obtained);
- the value with all bits 1 is at the same time the maximum value of the corresponding *unsigned* type;
- a configuration consisting only of bits 1 represents a power shape number of 2 - 1;
- the odd numbers are those that have the least significant bit equal to 1 (the only odd power of 2 that appears in sum when transforming from base 2 to base 10).

We can say that the following *rule for transforming a number from base* 10 *to base* 2 is valid (apart from the general one where we repeatedly divide to 2 and take the remaining values in reverse order): For the given number, we look for the highest power of 2 less than or equal to the given number, then subtract it from the number and resume for the remaining value. We repeat this until the value reached is 0. This approach is correct because we will not determine twice the same power of 2 (the sum of two powers of 2 is also a power of 2 and would mean that at a step we would not have found correctly the power of 2 less than or equal to the current value). Also, the fact that the powers of 2 that appear are distinct is consistent with the fact that we have only the digit 0 and the digit 1 in the base 2.

Exemple: We would like to write in base 2 the number 41. We look for the biggest power of 2 less than or equal to 41 and we find 32. We continue with 9 (41 - 32). We find 8 and we continue with 1 (9-8). Now we have 1 and so we reach 1-1 = 0, meaning that we can stop. So $41 = 32 + 8 + 1 = 2^5 + 2^3 + 2^0$. So the writing in base 2 of the number 41 is: 101001.

Bitwise operators are divided into two categories: *logical bitwise operators* and *shift operators*.

Logical bitwise operators

We emphasize what we said before, they apply to whole data.

They are:

~	*not* logical on bits, it is a unary operator
&	*and* logical on bits, binary
|	*or* logical on bits, binary
^	*exclusiv*e (*xor*) logical on bits, binary

We notice that operators `and/or` are written differently than logical ones that contain the double sign character. The operator `~` is unary, it is applied to an integer operand and the result is the one obtained by complementing the bits of the operand. Obviously, it has no effect on the operand, which can be any kind of expression.

Examples:

Code sequence	What is displayed	Explanation

`short a;`
`a = -1;`
`cout<<(~a);`
	`0`	The internal representation of `-1` is formed only of bits `1`. After complementing them, only bits `0` are obtained which represents the internal encoding of the value `0`.

`short a;`
`a = -236;`
`cout<<(~a);`
	`235`	The internal writing of `-236` is:
		`1111111100010100`
		Its negation:
		`0000000011101011` (`235`)

The `&` operator is binary, it is applied to two integer values and the result is what we get by applying "`and`" bit by bit between all pairs of bits in the same position. At "`and`" the result is `1` only if both "operands" are `1`.

Examples:

Code sequence	What is displayed	Explanation

`short a, b, c;`
`a = 101;`
`b = 87;`
`c = (a&b);`
`cout<<c;`
	`69`	`a: 0000000001100101`
		`b: 0000000001010111`
		`c: 0000000001000101`

`short a, b, c;`
`a = 501;`
`b = -236;`
`c = (a&b);`
`cout<<c;`
	`276`	`a: 0000000111110101`
		`b: 1111111100010100`
		`c: 0000000100010100`

If we imagine the writings in base 2 as being sets represented by a characteristic vector, we can say that after applying the operator & we obtain the intersection of the two sets.

The | operator is binary, it is applied to two integer values and the result is what we get by applying "or" bit by bit between all pairs of bits in the same position. In "or" the result is 1 only if there is 1 in at least one operand.
Examples:

Code sequence	What is displayed	Explanation

short a, b, c;
a = 101;
b = 87;
c = (a|b);
cout<<c;
	119	a: 0000000001100101
		b: 0000000001010111
		c: 0000000001110111

short a, b, c;
a = 501;
b = -236;
c = (a|b);
cout<<c;
	-11	a: 0000000111110101
		b: 1111111100010100
		c: 1111111111110101

If we imagine the writings in base 2 as being sets represented by a characteristic vector, we can say that following the application of the operator | the reunion of the two sets is obtained.

The operator ^ is binary, it is applied to two integer values and the result is what we obtain by applying "exclusive or" bit by bit between all pairs of bits in the same position. At "exclusive or" the result is *1* only if exactly *one* operand is 1.
Examples:

Code sequence	What is displayed	Explanation

short a, b, c;
a = 101;
b = 87;
c = (a^b);
cout<<c;
	50	a: 0000000001100101
		b: 0000000001010111
		c: 0000000000110010

short a, b, c;
a = 501;
b = -236;
c = (a^b);
cout<<c;
	-287	a: 0000000111110101
		b: 1111111100010100
		c: 1111111011100001

If we imagine the writings in base 2 as some sets represented by a characteristic vector, we can say that following the application of the operator ^ the symmetric difference of the two sets is obtained.

The shift operators

There are two shift operators, the left shift operator and the right shift operator. The operator << (moving left) is binary, both operands being integers. The result is obtained by translating to the left the bits of the left operand with as many positions as the value of the right operand. The values of the bits that were in front and are out of the memory area will be lost and the right side will be filled with 0.

Examples:

Code sequence	What is displayed	Explanation

short a, b;
a = 101;
b = (a << 3);
cout<<b;
	808	a: 0000000001100101
		b: 0000001100101000

short a, b;
a = -236;
b = (a << 2);
cout<<c;
	-944	a: 1111111100010100
		b: 1111110001010000

Here's a very important observation: the operation a << b is equivalent to $a * 2^b$. If we look at the first example, it is easy to prove what we said earlier. Moving to the left with 3 positions all the exponentials of the powers of 2 that made up the representation reach with 3 larger ones, that is, if at the new representation we give common factor 2^3 we get in parenthesis even the initial value.

$101 = 2^6 + 2^5 + 2^2 + 2^0$.

$808 = 101 << 3 = 2^9 + 2^8 + 2^5 + 2^3 = 2^3 * (2^6 + 2^5 + 2^2 + 2^0) = 2^3 * 101$.

The >> operator (moving to the right) is binary, both operands being integers. The result is obtained by translating to the right the bits of the left operand with as many positions as the value of the right operand. The values of the bits that were in the back and are out of the memory area will be lost and the left side will be filled with the sign bit. So, in the negative numbers it will be filled with 1 (on all the necessary positions), and in the positive numbers, analogous, with 0. Thus, the sign of the result will be the same as the sign of the left operand. With a similar reasoning to the one shown to the left shift operator, we deduce that the shift to the right is equivalent to the division to a power of 2. More precisely, the expression a >> b is equivalent to $a / 2^b$. Here is just the quotient of the division (bits of 1 can be lost on the right when the division is not complete).

Examples:

Code sequence	What is displayed	Explanation

`short a, b;`
`a = 101;`
`b = (a >> 1);`
`cout<<b;`
	`50`	`a: 0000000001100101`
		`b: 0000000000110010`

`short a, b;`
`a = -236;`
`b = (a >> 2);`
`cout<<c;`
	`-59`	`a: 1111111100010100`
		`b: 1111111111000101`

Bitwise operators (all except ~) can be combined with the assignment operator to obtain new shortcut operators. Thus, you can write form expressions:

variable op = expression

where op can be: `&`, `|`, `^`, <<, >>.

Exemple: `n = (n >> 1)` may be written as `n>>=1`

Pay attention to the use of bitwise operators in expressions with other operators as they have very low application priority and so it is recommended to use as many brackets as possible for grouping.

Using bitwise operators in expressions leads to increased running speed because for each of them the processor has its own instructions.

Here are some things we can do elegantly using bit operators.

`1<<k`	This expression is equivalent to 2^k. By using the operator we obtain this result directly compared to the classical calculation that requires repetition.
`1LL<<k`	Thus it must be written if we want to obtain powers of `2` greater than `32`. In the above expression constant `1` was of type int (so it is implicit in C / C ++) so we must put the suffix to force the representation of `1` on `64` bits.
`n>>1`	Equivalent to `n / 2`
`n&1`	This expression is equivalent to `n%2`. This means we can test the parity of a number this way. Here is the justification: the number `1` has only bits `0`, except the least significant bit which is `1`. Putting 1 as an operand next to `&` all bits of the result, minus the last one, will be `0`. The last bit is obtained as `&` between bit `1` (from number `1`) and the last bit of the other operand. So you get `1` only if that bit is `1`, too.
`n^n`	`0`
`n^n^n`	`n` (`xor` applied by an even number of times on the same value leads to `0`, and `xor` applied by an odd number of times on the same value leads to that value).
`n^0`	`n`

`(n >> k) & 1`	This is the expression we write to get the value of bit `k` of `n`. In practice, by shifting we ”take” that bit to the last position, and according to the previous example, it remains to apply `&` with the number `1`.

`n = n + (1<<k)`
or
`n += (1<<k)`
	This instruction adds to `2k` the value `n`. If the bit at position `k` of `n` is `0`, this expression has the effect of setting that value to `1` in `n` of that bit. Note that if the bit in position `k` is already `1`, it becomes `0` and more significant bits will be affected as transport to the assembly operation in base `2` appears.

`n = (n | (1<<k))`
or
`n |= (1<<k)`
	This is the expression that sets to `1` the bit value at position `k` in `n` regardless of its previous value, the rest of the bits remaining unchanged. If we imagine the binary representation of `n` as a characteristic vector associated with a set, by this operation we add (if it does not already exist), to the set the element `k`.
	Here, more detailed, we prove what happens:
	`k = 4`
	`n: XXXXXXXXXXXXXXXX`
	`1: 0000000000000001`
	`1<<k 0000000000010000`
	At ”`or`” with bit `1`, at that position the obtained bit will be `1`, and the others remain unchanged since `0` is a neutral element in this operation.

`n = (n & ~(1<<k))`
or
`n &= ~(1<<k)`
	These are ways to set to `0` a certain bit of a variable, regardless of its previous value and without affecting the values of the other bits. If we knew for sure that the bit `k` is `1`, to make it `0` we could use the expression `n - = (1 << k)`
	Here, more detailed,it is what happens:
	`k = 4`
	`n: XXXXXXXXXXXXXXXX`
	`1: 0000000000000001`
	`1<<k 0000000000010000`
	`~(1<<k) 1111111111101111`
	After the operation `&` with bit `0`, `0` is obtained, and otherwise no changes occur as `1` is a neutral element in ”`and`”.

`n & -n`	The value of this expression is always a power of `2`. It represents the least significant bit of `n`. If we analyze the complement rule for `2` in order to find the absolute value for a negative number, we notice that that is the only bit that remains `1` at both `n` and `- n` (the remaining positions have a maximum operand `1`).

`for (;n;n-=(n&-n))`
`{`
`  cnt++;`
`}`
	This code sequence counts in cnt how many bits of `1` it has `n` written in base `2`. It is observed that at every step it is eliminated from its least significant bit. It is interesting that the number of repetitive steps is equal to this value. In the following example we will also present the raw variant that checks the value of each bit separately.

	We emphasize that the current code sequence is also useful in implementing the data structure called binary indexed trees.

`for (i=15;i>=0;i--)`
`  if ((n>>i)&1)`
`  cnt++;`
	This is the sequence equivalent to the previous one but the number of steps is always equal to the total number of bits of the internal representation of `n`.

Exercises and solved problems

Problem	Solution
1. For a given variable of type `int`, write an expression that gives the value obtained with its last `k` bits (and the value `k` is given).	The operator `&` is applied between the value of `n` and the one with `1` on the last `k` positions and in the rest `0`. We have shown above that such values are powers of `2` of which is subtracted `1`. We obtain this: `n&((1LL<<k)-1).`
2. For a given variable of type `int`, write an expression that gives the value obtained with its bits from positions `6,7,8`.	We reduce this problem to the previous one by moving the three bits so that they reach the last three positions. Then we apply the above code for `k = 3`. `(n>>6)&7.` We chose to use directly `7(1112)` being equivalent to `(1LL<<k)-1)` for `k = 3.`
3. Write an expression that sets to `1` the last `2` bits of the value of an `int` variable, regardless of their previous value. The rest of the bits must keep their value.	`n |= 3` I made "`or`" with the number that has `1` on the last two positions, in the rest `0`. This is exactly `3`.
4. Write an expression that sets to `0` the last `2` bits of the value of an `int` - type variable, regardless of their previous value. The rest of the bits must keep their value.	`n &= (-1 << 2)` Here we use "`and`" the other operand having a value that has `0` on the last two positions and subtract `1`. We obtained it using the fact that `-1` contains only bits of `1` and the shift to the right introduces `0` at the end.
5. Given a variable of type `short`, write a sequence of instructions for the interchange of its two bytes (the sequence of the most insignificant `8` bits must arrive, in the same order on the most significant `8` bits, and vice versa).	`n = ((n&255)<<8)+((n>>8)&255)` Number `255` represents the value with the last `8` bits `1`. The first term obtains the value with the last `8` bits and by moving it `8` positions to the left, it adds practically `8` to `0` at the end, preparing it to add the value with the first `8` bits (obtained in the the second term by an "`and`" with `255` after going to the last 8 positions, by moving with `8` positions to the right).

6. Write a sequence of assignments that interchanges the values of two variables `a` and `b`, using the `xor` bitwise operator and without additional variables.	`a=a^b;`
	`b=a^b;`
	`a=a^b;`

Problem: We give n and n non-negative values and we ask to say for each how many bits of 1 it has in the internal representation. The given values are 32 bits and the number of given values is up to 10^6.

Solution:

The first solution is to apply the raw algorithm that tests each of the 32 bits of each number in turn. Thus, the number of steps taken is of the order n * 32 (32 = the number of bits on which the values are represented).

Another solution is to use the optimized alternative to obtain the number of bits 1 of a value (the one by which we always eliminate the least significant bit of 1 and count, that is n - = (n & -n)). However, if all given numbers have large number of bits 1, the execution time tends towards the one from the previous version.

There is also a third, much faster solution, which we will present below. It is based on the precalculation, for certain numbers, of the number of bits and the use of this information. Calculating how many bits of 1 has each value on 32 bits is not practical, because they are of the order of 4 billion values and both the time to realize this and the memory needed to store all the values would be too large. We will precalculate the number of bits 1 for each possible number on 16 bits and then see how we use this information.

On 16 bits there are 65536 possible values, that is the memory used is not big and we will see in the count that neither the time used. If we note B [i] = the number of bits 1 of the writing in the base 2 of i, we have:

B[0] = 0;

B[1] = 1;

B[i] = B[i/2] if i is even;

B[i] = 1+B[i/2] if i is odd.

This is because the value i / 2 contains exactly the same bits as i, except the last bit of i (i / 2 is equivalent to i >> 1 and it is possible to lose a bit of 1, in which case the number i would be odd and then the value 1 will be added to the result).

Therefore, the precalculation is as follows:

```
b[0] = 0;
b[1] = 1;
for (i=2;i<=(1<<16)-1;i++)
    b[i] = b[i/2] + i%2; /// or b[i] = b[i>>1] + (i&1);
```

If all given numbers are on 16 bits, we can use precalculation and solve the problem as follows:

```
cin>>n;
for (i=1;i<=n;i++) {
    cin>>a;
    cout<<b[a];
}
```

But we can do a little trick and solve the problem also for 32-bit numbers:

```
cin>>n;
for (i=1;i<=n;i++) {
    cin>>a;
    cout<<b[a & 0xFFFF] + b[(a>>16) & 0xFFFF];
}
```

What is the meaning ?

The value `0xFFFF` represents a `16`-bit number, all of `1`, that is (`216-1`), I could also write `(1 << 16) -1`, as we have used so far in the article. This is because `F` (`15`) is written as `1111`. Therefore, writing `a & 0xFFFF` gives us the value that is obtained by taking into account only the last `16` bits of `a`, so accessing `B` at this position we obtain as many as `1` among the last `16` bits of `a`. It remains to add the number of bits with the value of `1` of the first `16` bits of `a`. We notice that if we move to the right with `16` positions, the bits that interest us remain the last `16` so we can use on the new values formula from the other `16` - bit group.

We can say that we now answer directly for each given number, using an expression, without additional iterations.

Conclusions

A number of common operations (such as those with powers of `2`) can be performed very quickly using bitwise operators, often reducing the use of a repetitive code sequence when evaluating a single expression. The usual operations with sets can also be done elegantly by using bitwise operators, obtaining besides good execution time a reduction in the memory used, too. For example, we can represent a set that can have values between `0` and `31` through a single int type date, this having the characteristic vector meaning for the set. Thus, the intersection operation of two sets can be performed by applying the bitwise operator `&` to the variables that encode the set and the meeting operation by applying the operator `|`.

References

1. King K.N. C Programming: A Modern Approach, 2th Edition. W. W. Norton & Company, New-York, 2008. 832 p.
2. Kelley A., Pohl I. A Book on C: Programming in C, 4th edition. Addison-Wesley Professional, USA, 1998. 752 p.
3. Kochan S.G. Programming in C, 4th Edition. Addison-Wesley Professional, USA, 2014. 600 p.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 73-85 E-ISSN 2587-3636

CZU: 372.8004 DOI: 10.36120/2587-3636.v18i4.73-85

DIDACTIC ASPECTS REGARDING TO CREATING TEST SETS FOR COMPETITION PROBLEMS

Angela GLOBA[*], associate professor, PhD
Tiraspol State University (located in Chisinau)
Sergiu CORLAT[**], university lecturer
Technical University of Moldova

*Deputy leader of the national team of Moldova at BOI, EJOI, JBOI, IATI (since 2015); member of the Scientific Council at the Balkan Computer Science Olympiad (2017); member of the Republican Olympic Council of Informatics.

**Team leader of the national team of Moldova at IOI (2014-2019) and other international competitions; coordinator of the national batch of Moldova in computer science (2014-2019); member of the Scientific Council at the Balkan Computer Science Olympiad (2017).

Abstract. A correct evaluation and a qualitative separation of the participants is a major objective for any programming contest. The article examines this aspect from the perspective of the correct elaboration of the tests for the competition problems. Examples of tests analyzed in the article show the added value brought in the elaboration and solving of the competition problems. The article is focused on the elucidation of the methodological aspects regarding the elaboration of tests for the competition problems in the categories of Combinatorics and computational Geometry.

Keywords: competition problem, test set, evaluation, programming contest, problem solving, algorithm, algorithm efficiency, computational geometry, Combinatorics, programming techniques.

ASPECTE DIDACTICE REFERITOR LA CREAREA SETURILOR DE TESTE PENTRU PROBLEME DE CONCURS

Rezumat. O evaluare corectă și o separare calitativă a participanților este un obiectiv major pentru orice concurs de programare. Articolul examinează acest aspect din perspectiva elaborării corecte a testelor pentru problemele de concurs. Exemplele de teste analizate în articol arată valoarea adăugată adusă în elaborarea și rezolvarea problemelor competitive. Articolul este axat pe elucidarea aspectelor metodologice privind elaborarea de teste pentru problemele de concurs în domeniile Combinatorică și geometrie computațională.

Cuvinte cheie: problemă de concurs, set de teste, evaluare, concurs de programare, rezolvare de probleme, algoritm, eficiență algoritm, geometrie computațională, combinatorică, tehnici de programare.

1. Introduction

At the current stage, the Informatics Olympiads and other Programming Contests are quite popular and diverse. They have a long history and a long life even after their development. The competition problems, however, represent the image of any contest, and the success of the competition depends on how successfully (correctly) they will be developed. Moreover, after the contest, the problems gain a new life: they spread rapidly among the future participants in the programming contests, are used in the subsequent preparations for competitions by teachers and mentors, and on their basis new problems are created for the Olympiads [1]. The elaboration of competition problems for any stage

(institutional, local, National, International) is a rather difficult task for authors, which is why it requires complex methodical approaches, which take into account: the theoretical and practical training of the participants; psychic aspect and stress situations; the complexity of the competition, etc.

A competition problem includes a minimum set of resources (fig. 1) [2].

The methodology for evaluating a competition problem is largely determined by its type and the form established for presenting its solution (the output data), and the test sets of the problem have a substantial importance.

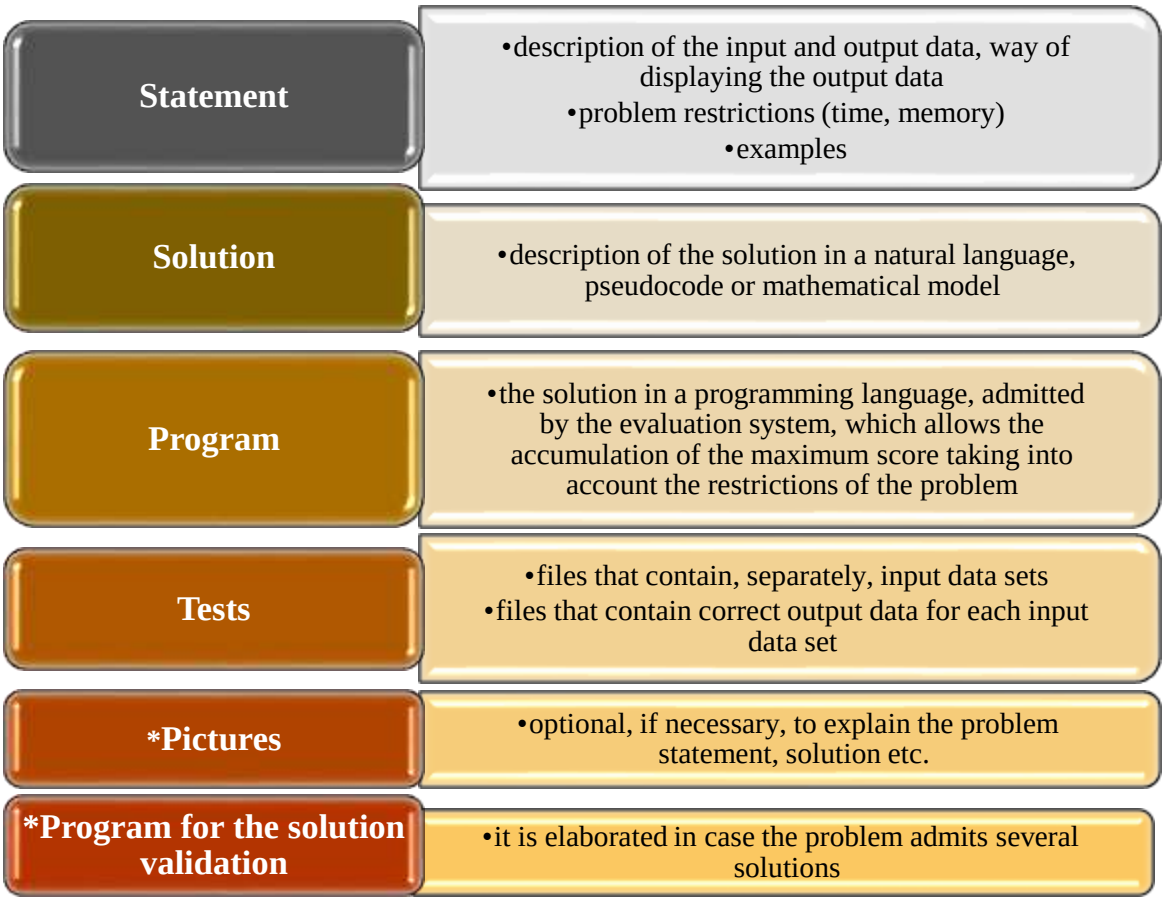


Figure 1. The minimum set of resources for a competition problem (* - optional)

2. Types of tests for a competition problem

In the process of developing the evaluation methodology, initially, it is necessary to establish the maximum score for a complete solution of the problem, then this score is distributed between the partial solutions of the problem or between the solutions for the subtasks highlighted in the statement of the problem. To determine the maximum number of points for a problem, usually two approaches are used. The first approach is based on the preliminary assessment of the complexity/difficulty of the problems selected for the competition. Thus, problems are evaluated with different scores, depending on the degree of difficulty. The second approach is that, each problem is evaluated in the same way, for example with 100 points, regardless of the complexity (degree of difficulty) estimated. The

second approach is used more often in all competitions: municipal, republican as well as in the international Informatics Olympiads and contests. This approach has as a solid foundation the following reasoning. It is difficult to answer the question *Which problem is the most complicated for the participants?* except when the competitors' level of training is known from the beginning. As the results of the participants in the contest are determined by several factors which are difficult to consider at the beginning of the competition, the introduction of the complexity coefficients of the problem loses its meaning. There are cases when, according to the jury, a simple problem can be quite complicated for all the participants and vice versa. There are cases when the first approach is applied, i.e. the problems are evaluated depending on the degree of difficulty, and a large part of the participants, uncertain of themselves, begin to solve problems evaluated with a minimum number of points, while another part of the participants, more confident in their own forces, do the opposite. As a result, both the first and the other spend a lot of time solving the first chosen problem and fail to solve other problems, not because they are complicated, but because they did not have enough time. There are situations when strong participants fail to solve the simplest problem, but such cases already relate to the psychological aspect of the participants, which has a role no less important than the level of preparation of the competitors for the competition.

The distribution of the maximum number of points (for example, 100) for a problem between its partial solutions is generally based on the set of tests developed. If the solution of the proposed problem is obtained after running the program proposed by the participants, then the test set is elaborated in such a way as to allow the jury to correctly evaluate different algorithms proposed by the participants in order to obtain the maximum score according to their correctness and effectiveness.

The complete test set for a competition problem should include the following test groups:

1) tests of minimum size (trivial tests);
2) tests for particular cases, which allow to identify the specific characteristics of the implemented algorithms;
3) tests for the accuracy of calculations with real numbers, if the input data cause numerical instability of the algorithms;
4) tests that highlight the specific characteristics of the programming language or environment (for example, the difficult accomplishment of the operations with large numbers in the Pascal language, the inefficient implementation of the input-output flows and the sequential containers in C ++);
5) random tests of different dimensions (here we refer to the dimensions of the input data: from simple tests to complicated tests);
6) tests that check the presence of heuristic in algorithms (which does not analyze all the available information, a very fast procedure for solving a problem);

7) maximum size tests (allow to evaluate the effectiveness of the proposed algorithms at maximum dimensions of the input data specified in the problem restrictions).

The distribution of the maximum number of points for a problem is made initially between all the test groups, and then the score for each group is distributed between the tests in each group. This distribution is made in a table where each test and group of tests is assigned a certain number of points. For example, for the *Rectangles* problem (IOI 2019) it is proposed table 1.

Table 1. Distribution of the points number for each subtask (Rectangles problem, IOI 2019)

Subtasks	Points	Restrictions
1 11 tests	8	$n, m \leq 30$
2 8 tests	7	$n, m \leq 80$
3 8 tests	12	$n, m \leq 200$
4 16 tests	22	$n, m \leq 700$
5 11 tests	10	$n \leq 3$
6 8 tests	13	$0 \leq a[i, j] \leq 1$(for all $0 \leq i \leq n-1, 0 \leq j \leq m-1$)
7 18 tests	28	No additional constraints
Total	**100** points	

When distributing the score for a competition problem between all test groups, the following principle will be taken into account: the correct solution for all restrictions in the problem statement will accumulate the maximum score, while the correct solution for some subtasks, but inefficient for all restrictions, should accumulate between 30% and 70% of the maximum score offered for the given problem.

Since each test in the group is used to test the properties of the proposed algorithm for solving the formulated problem, the score within the group is distributed taking into account the importance of these properties for solving the problem as a whole. If the contestant gets correct answers to the tests from a group or to certain tests within that group, he is credited with the number of points set for this group or for the tests covered, otherwise points are not awarded.

Analyzing the information in table 1 it can be concluded that for subtasks 1, 2, 5 the accumulation of a point is possible only in case of passing/covering at least two tests from the group. For subtasks 3, 4, 6, 7 the distribution of the score per test is more complicated, but not less than one point per test.

If the competition problem is divided into several separate subtasks, then the evaluation of the proposed solution for a subtask can be carried out for the whole test group, i.e. the maximum score is offered only if the entire test group has been passed successfully,

otherwise no points are awarded. The evaluation can also be done for each separate test, but in this case, a prior distribution of the score is required for each test from the group. The first approach is simpler in terms of implementation, but more "painful" for the contestants. The second approach is more "friendly" with the participants in the competition, but requires a careful analysis of each separate test with the identification of the test subsets for accumulating a point or the test subsets for accumulating the maximum score per group, etc.

The total score, obtained by the contestant, for solving a problem represents the sum of the score accumulated in the tests from all the test groups elaborated for the proposed problem. Thus, the final score accumulated by the contestant in all the problems proposed within the contest is formed as the sum of the total scores received by him for each problem.

3. Methodological benchmarks regarding the creation of the test set for a competition problem

The problems proposed for solving to the participants in the Informatics competitions fall into the following categories: combinatorial problems; problems with computational graphics; problems focused on data structures; simulation problems etc.

2.1. The test set for combinatorial problems

To solve combinatorial problems (processing the elements of a set, generating the set/subset, searching for a subset or element of the set with certain properties, etc.) the contestants must implement a programming technique: the most elementary - Brute force, Backtracking, Greedy, Divide and Impera and, the most difficult - Dynamic Programming [3].

Obviously, when generating tests for such problems, the author will take into account the programming technique adopted. For example, the **Books** problem (Republican Informatics Olympiad, 2015), which, in fact, is a classic problem:

Description: The great tycoon BitMan has in one of its stores n boxes, numbered from 1 to n, full of books. There are no boxes with the same number of books. The i box contains x_i books ($x_i \neq x_j$, $1 \leq i, j \leq n$). BitMan expects to go to the orphanages and distribute books equally, taking only k boxes.

Task. Make up a program that would determine the maximum number of books that BitMan could donate to each orphanage.

Input. The standard input contains, on the first line, a natural number n - the number of boxes in the stores. The second line contains the natural numbers $x_1, x_2, \ldots, x_n$, - the number of books from boxes 1, 2, ..., n, separated by a space.

Output. The standard output will contain a natural number - the amount of books each orphanage will receive.

Constraints: $1 \le n \le 255$; $1 \le x_1, x_2, \ldots, x_n \le 1000$. The execution time will not exceed 1 second. The program will use up to 32 megabytes of operational memory.

Examples.

book.in	book.out	Explanation
5 **12 9 14 17 11**	**8**	BitMan can take boxes 1, 4 and 5 or boxes 2, 3 and 4, each orphanage receiving 8 books. Any other set of boxes will not allow BitMan to share books equally, distributing more than 8 books to each orphanage.

The task of the problem does not include the display of the boxes number that BitMan has to take, as it admits several solutions (for example, (1,4,5), (2,3,4)), but the maximum number of books that each orphanage will receive is unique. Obviously, all the solutions of the problem can be displayed, but for this it will be necessary to develop a validation program of the problem solution.

The solution of the problem is reduced to the processing of the elements of an A set of *n* elements, more precisely, the identification of a subset of the A set for which the sum of the elements of the respective subset is divided by *n.* The formulated problem always admits a solution.

Tests production. The tests set was produced according to programming technique adopted.

Programming technique	Description	Points
Brute force	It is known that the number of subsets of a set with *n* elements equals to 2^n. The brute force is reduced to: generating all the subsets of the given set; keeping subgroups generated in a vector; calculating the sum of the elements of each subset with checking the condition of divisibility of the sum calculated with *n;* choosing the maximum amount divisible by *n.* The temporal complexity of this algorithm is $(O(2^n))$. $1 \le n \le 10$; $1 \le x_1, x_2, \ldots, x_n \le 1000$	10
Backtracking	This technique is faster than Brute force, but it also admits a time complexity equal to $(O(2^n))$. In this case, only those subsets having the sum of the elements divisible by *n* are generated. In this case, it is necessary to save in a variable, at the generation stage of the *i* subset, the maximum sum divisible by *n*. $1 \le n \le 20$; $1 \le x_1, x_2, \ldots, x_n \le 1000$	30
Dynamic Programming	The implementation of the Dynamic Programming technique admits a temporal complexity equal to $(O(n^2))$. $1 \le n \le 255$; $1 \le x_1, x_2, \ldots, x_n \le 1000$	60

It is recommended that the tests, which allow to obtain a score by implementing the Brute force technique, admit a score of not more than 10 points out of 100 (depending on the level of the programming competition: sectoral, republican, international, etc.), with a purpose, more, to stimulate the participants in the contest than to evaluate them.

The use of the Backtracking technique to obtain the solution of the problem speaks of elementary knowledge of the programming techniques, by the participants in the contest, a fact which deserves to be stimulated and appreciated by the jury. It is also recommended

that the tests that support the application of the Backtracking technique for writing the solution cover a score of no more than 20-30 points out of the total score of 100 points.

A solution, which would cover all the tests, is provided by the Dynamic Programming technique - a difficult technique from several points of view. The contestant will have to find that principle of optimality, which would give him the solution of the problem in a useful computation/calculation time, but, he will need knowledge in the field of Mathematics, such as: theorem of division with remainder, Dirichlet's principle etc., and the correctness of the algorithm can be demonstrated using the mathematical induction. Greater attention will be given to the data structures used to store partial solutions.

2.*2. The test set for computational Geometry problems*

The description of the test creation methodology for computational Geometry problems was made based on an example - the Tale problem (Balkan Olympiad in Informatics, 2017, [4]).

Description: Every year, in the middle of summer, a short-stature king, called Statu Palmă Barbă Cot, gives a feast in his castle at which all the knights from the Balkan countries are invited. Făt Frumos is one of the guests, and, like a true knight, he rides to the feast on his fairy horse, named Flămânzilă. Horses are forbidden inside the castle, therefore Făt Frumos intends to tie up Flămânzilă to one of the trees nearby, outside the castle. Flămânzilă is very quiet as long as it has something to eat (its favourite grass grows everywhere), but, after it grazes all the reachable grass, it becomes nervous and begins to blow fire, like a dragon. At such a point, Făt Frumos has to leave the feast in order to calm his quadruped companion. In order to prevent Flămânzilă from causing a fire, Făt Frumos needs to know the area of the figure within which it can move. The figure is shaped by:

- the tree to which the horse is tied up (a point of integer coordinates *Xc*, *Yc*);
- the length of the rope *L* (a positive integer);
- the wall of the castle that the horse cannot jump over (the wall forms a convex polygon with *N* edges).

Task: Write a program to calculate the area of the figure within which Flămânzilă can move.

Input. The standard input contains, on the first line, three integers X, *Y*, and *L*, separated by a space - the coordinates of the tree to which the horse will be tied up and the length of the rope. The second line contains the positive integer *N* - the number of vertexes in the polygon. *N* lines follow, each of them containing two integers *Xi* , *Yi* (*i* = 1, ..., *N*), separated by a space - the vertexes of the polygon, given clockwise.

Output. The standard output will contain a real number with five decimals - the area of the figure within which Flămânzilă can move.

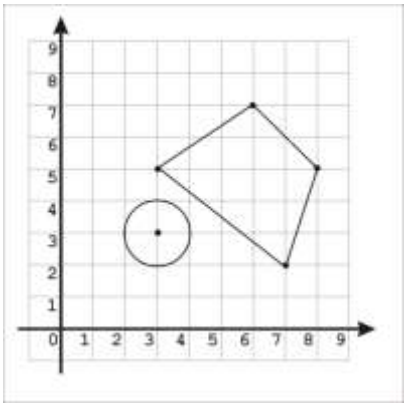
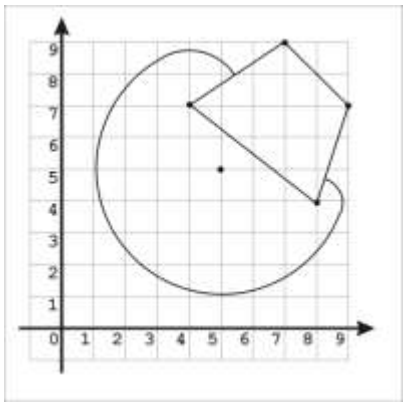
Constraints: $-10000 \leq X, Y \leq 10000$; $3 \leq N \leq 1000$; $1 \leq L \leq 10^9$**.**

Subtasks:

Subtask	Description	Points
1	The length of the rope does not exceed the distance from the tree to the wall	10
2	The length of the rope does not exceed the half-perimeter of the wall	60
3	The length of the rope allows reach all the points on the polygon edges, making a combined cover in two directions: clockwise and anticlockwise. However, each of the invisible vertices of the polygon (a vertex is invisible if the segment joining it with the tree intersects the polygon in an interior point) can be reached from no more than one direction.	30

Note: Results will be evaluated with a precision = 1.0.

Examples:

tale.in	tale.out	
3 3 1 **4** **3 5** **6 7** **8 5** **7 2**	**3.14159**	
5 5 4 **4** **4 7** **7 9** **9 7** **8 4**	**36.71737**	

Tests production. The tests set was produced according to subtasks description [5, 6].

First subtask was the simplest one: because the circle does not intersect the polygon, calculations has to be done using standard formula for circle area. (see example 1) The test production was also relatively simple, but a visual verification tool, programmed by task authors was used to exclude some overlaps between the circle and the polygon.

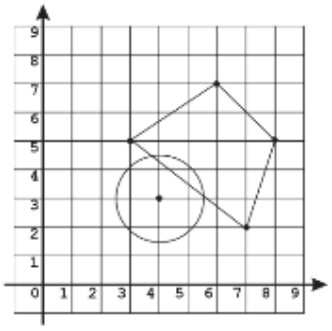
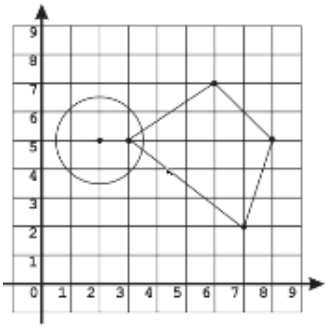
The second subtask is the biggest one. There are more situations to solve inside this subtask. The first case – circle is not transformed, it just intersects a side or couple of sides, no touch extreme viewpoints of the polygon. (see images a - c below) These case tests are easy-solved using simple calculations for circle sector area. Tests for this case were produced in two modes: "by hand", in developed visual tool (for small test cases: $N < 20$)

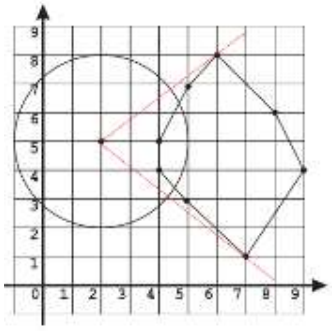
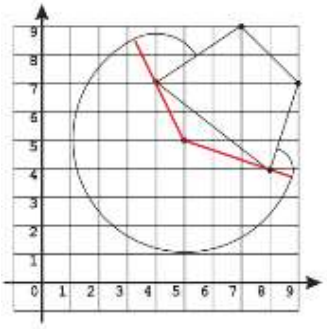
and randomly, for medium and large sets of input data. The process of random test production was divided into three steps:

- generate a convex polygon (the problem solved by generating a set of points in *R*2, then finding a convex hull of this set);
- place randomly a "tree" point outside the polygon;
- finding extreme "viewpoints" in the polygon. Extreme "viewpoints" are the polygon vertexes, which, together with "tree" point forms an angle, that contains the whole polygon inside;
- set randomly rope length to be less then distance to nearest "viewpoint" and longer then the distance to polygon.

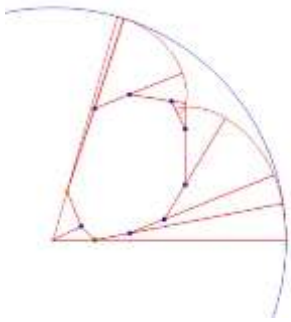
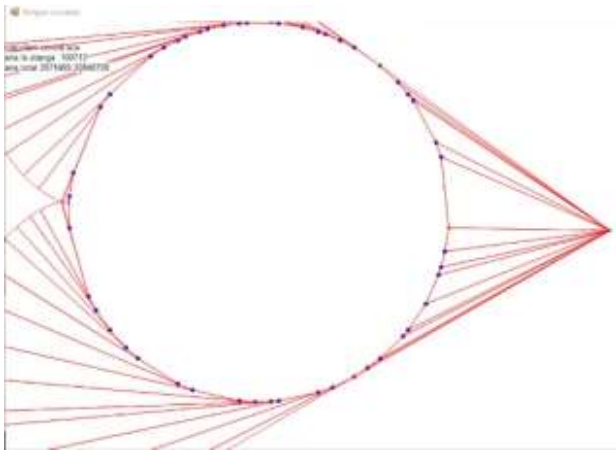
The second case takes in consideration circle deformation after touching a vertex (extreme viewpoint). The series of deformations continue with each next side, touched by the rope (image d). Some "pie slices" will be added to the calculated area.

There were two types of generated tests: produced "by hand" in a visual environment and random generated. The production follows the procedure, like described above. The only difference was that rope length restrictions were longer that the distance from "tree" point to nearest extreme "viewpoint" and less than a half of polygon perimeter.

 a)	Subtask 2, case 1a intersection with one side
 b)	Subtask 2, case 1b intersection with two sides. No "full" sides inside the circle

 c)	Subtask 2, case 1c intersection with two sides. some “full” sides inside the circle, but extreme viewpoints are not touched.
 d)	Subtask 2, case 2 intersection with two sides. some “full” sides inside the circle, one or two extreme viewpoints are touched.

***Third subtask*.** The most complicated – rope overlap is possible. The overlap is allowed but will not exceed singular side zone. Like in previous cases small size tests were produced directly in visual environment, and large – using random parts, in a mode similar to previous. An additional step was the singularity check for the side that can be reached from both directions: by left and right polygon crossing. After production all tests were additionally tested by visual checker and solver. (see images below)

	
Test case visualization result: one common side, test ok.	Test case visualization and solving: one common side, test ok. The calculated area and some intermediary info – upper left corner.

4. The impact of the tests, created by the participants during the contest, on the solution of the problem

From experience, the success at the Informatics Contest is ensured not only by the participants' knowledge, but also by the strategy of approaching the competition problems.

It often happens that a contestant will find a good solution to a problem, which will not ensure the maximum score, giving up the search for the most efficient solution, because the time used is not directly proportional to the accumulated score. This time can be used to solve other problems. In this case, this is a correctly chosen competition strategy.

There is also a substantial impact on the success achieved by the contestant, both internal and external psychological factors: the temperament; the individual experience; the lack of sleep before the competition; the psychological climate established between the team members, including with the deputy and team leader; the psychological climate during the competition; the complexity/level of the competition, etc.

Usually, the first participation in an Informatics Olympiad provides the contestant with a maximum average score, which is due to the specificity of Informatics Olympiad. The experience accumulated by the participants matters a lot, as, historically, it is shown that the true results begin to appear starting with the second participation, sometimes even later.

To solve a competition problem, the contestant must go through several stages (fig. 2) [7].

Figure 2. Stages of solving competition problems

If the compilation of the program was successful, then one can go to the test stage of the solution on his own tests, as, the contestant does not have access, that is, he cannot see the tests elaborated by the jury (the input files). It is recommended to go through the following steps:

- ✓ checking the solution on the tests from the example of the problem statement (for example, here errors can be identified regarding incorrect reading of the input data, display of the output data, etc.); the size of the input file can essentially change the difficulty of the problem (this is the computation time, which usually does not exceed a few seconds);
- ✓ checking the solution of the problem on a medium difficulty test, but which can be easily solved manually (if the test has been passed, the contestant can pass to other types of tests, minimum 4 tests and maximum 7-8);
- ✓ checking the solution on a set of small tests (For example, the program processes squares of side $0 \leq a \leq 10000$. Errors may occur for $a = 0$);
- ✓ checking the solution on a set of maximum size tests (If the program processes squares of side $0 \leq a \leq 10000$, *what happens if a = 10000*?); here you can create tests of a particular structure, so that it is easy to calculate the manual result (if it is complicated to manually create such a file, you can write a program for generating it);
- ✓ checking the solution on tests containing particular cases of the proposed problem (it may happen that these cases were not fully included in the program developed by the contestant);
- ✓ writing a program for generating random tests (this step allows you to check if the proposed solution does not block when allocating large areas of memory or if it does not exceed the time limit).

When testing the solution, it is better to consider the following recommendations:

1. If for some tests the program fails, it is necessary to run the program again, step by step; evaluate the results (variable values) after each step, thus you can precisely locate the subprogram, then the line that produced the error; run the program again on this test.
2. In case of program troubleshooting step by step, other errors may be highlighted; correct the errors and run the program on all previously tested tests.
3. Keep a copy of the unmodified program to restore the original form of the program, if the new version is not a better one;
4. If the problem statement does not specify the data, for example, the integer type, it is advisable to use the long long int type; increased attention should be paid to the accuracy of real numbers;
5. Pay attention to the calculation of the computation time.

Conclusions

In order to separate participants by their programming skills level the tests set will be generated in different sizes: small tests to check basic problem solution and, may be, some particular situations, easy to be identified; small tests for special cases, depending of problem domain; medium size tests for general solution based on standard techniques and/or data structures; medium size tests for combinations of special cases; large tests to test optimality of the algorithm and used data structures, for arbitrary generated data; extremal size tests for "sharpen" evaluation of best solutions.

The solutions of Computational Geometry problems usually are real numbers. The evaluation of floating-point results can be done in several modes: results are accepted by special checker program which calculates the difference between official solution and contestant's solution or the task asks for approximation to a given range or even to an integer value.

The correctness of the test sets elaborated for the evaluation of the competition problems solution is essential in the organization and qualitative development of an Informatics contest. The validation of the created tests is quite difficult for several reasons, but necessary to avoid some confusion that may arise during the evaluation phase of the solutions. The test set created for a problem in the Computational Geometry category can be verified using an application such as GeoGebra, Mathematica, Maple etc. Obviously, checking the correctness of the algorithm created is essential.

Bibliography

1. Chiriac L., Globa A., Bobeică N. Procedee metodice privind pregătirea şi desfăşurarea olimpiadelor de informatică. Mahematics & Information Technologies: Research and Education (MITRE 2011) dedicated to the 65th anniversary of the Moldova State University. Chişinău, August 22-25, 2011. p.168-169.
2. Pregatire pentru concursuri informatice: https://www.slideshare.net/scorlat/pregatire-pentru-concursuri-informatice
3. Globa A. Abordări metodice privind implementarea unor tehnici de programare prin prisma complexităţii algoritmilor. În: Acta et Commentationes. Ştiinţe ale Educaţiei. Revistă ştiinţifică. 2014, nr.1(4). Chişinău: Universitatea de Stat din Tiraspol, 2014. p.41-49. ISSN 1857-0623.
4. BOI 2017: http://boi2017.utm.md/wp-content/uploads/2017/01/TaleEng.pdf
5. Corlat S., Gremalschi A. Metodologia rezolvării problemelor de geometrie computaţională. Universitatea de Stat din Tiraspol, 2015.
6. Skiena S., Revilla M. Programming challenges. The Programming contest Training Manual. Springer, 2002.
7. Оршанский С. А. О решении олимпиадных задач по программированию формата ACM ICPC. В: Журнал „Мир ПК", Nr. 9, 2005.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 86-90 E-ISSN 2587-3636

CZU: 372.851 DOI: 10.36120/2587-3636.v18i4.86-90

THE METHODOLOGY OF USING THE LIFTING EXPONENT LEMMA AND HANSEL'S LEMMA IN CONTEST PROBLEMS

Marcel TELEUCA*, associate professor, PhD
Tiraspol State University, Lyceum "Orizont"
Mihai SPINEI, student
Lyceum "Orizont"
*Team leader, Deputy leader of the national team of Moldova at IMO (2013, 2011, 2010, 2005) and Balkan Mathematical Olympiad.

Summary. In this article we will analyze how we can use Lifting the exponent lemma (LTE) and Hansel's Lemma to solve Diophantine equations in mathematical contests. Firstly, we will go through the statements of the lemmas and some simple examples, and then we will show how they can be used in some recent problems.

Keywords: Number Theory, LTE, Hansel's Lemma, exponential Diophantine equations, polynomials, modular congruence.

METODOLOGIA UTILIZĂRII LEMEI LTE ȘI A LEMEI LUI HANSEL ÎN PROBLEME DE CONCURS LA MATEMATICĂ

Rezumat. În acest articol vom analiza modul în care putem folosi lema LTE și lema lui Hansel pentru a rezolva ecuațiile diofante la concursurile matematice. În primul rând, vom parcurge enunțurile lemelor și câteva exemple simple și apoi vom arăta cum pot fi utilizate în unele probleme recente.

Cuvinte cheie: teoria numerelor, LTE, lema lui Hansel, ecuații diofantine exponențiale, polinoame, congruență modulară.

Introduction

Solving the problems of competition in mathematics requires from the participants an involvement not only at the level of knowledge, but also elements of creativity. The solution of such problems is far beyond the curricular contents of the school. We propose in this article to highlight some didactic aspects on solving the contest problems in mathematics.

Initially, we will introduce some definitions:

a) Definition of v_p function

We define $v_p(n)$ to be the greatest power of a prime p that divides n.

(Example: $v_3(3) = 1,\ v_2(16) = 4,\ v_3(63) = 2$)

b) Proprieties of v_p function

The following will hold for any natural numbers k, a, b and for any prime p : $v_p(p^k) = k$

$v_p(ab) = v_p(a) + v_p(b)$,

If $p \nmid k$then $v_p(k) = 0$,

$v_p(n!) = \sum_{i=1}^{\infty} \lfloor \frac{n}{p^i} \rfloor$. (Legendre's Formula)

c) Proprieties of some recurrent expressions in the article

If for a natural number n and a prime p we have $n \vdots p$ then $(a^n - b^n) \vdots (a^p - b^p)$

And $(a^n - b^n) \vdots (a - b)$

Theorem Statements

Hansel's Lemmas

***Lemma* 1**: Let$x$ and y be (not necessarily positive) integers and let n be a positive integer. Given an arbitrary prime p such that:$\gcd(n,p) = 1$,$p \mid x - y$ and neither x nor y is divisible by p. We have $v_p(x^n - y^n) = v_p(x - y)$.

***Proof**:* We will use the fact that $x^n - y^n = (x - y)(x^{n-1} + x^{n-2}y + \cdots + y^{n-2}x + y^{n-1})$. So $v_p(x^n - y^n) = v_p(x - y) + v_p(x^{n-1} + x^{n-2}y + \cdots + y^{n-2}x + y^{n-1})$.

Also from $p \mid x - y$ we have that $x \equiv y(mod\ p)$.

Therefore $x^{n-1} + x^{n-2}y + \cdots + y^{n-2}x + y^{n-1} \equiv x^{n-1} + x^{n-1} + \cdots + x^{n-1} + x^{n-1} \equiv nx^{n-1} \not\equiv 0\ (mod\ p)$, in other words $p \nmid x^{n-1} + x^{n-2}y + \cdots + y^{n-2}x + y^{n-1}$, so $v_p(x^{n-1} + x^{n-2}y + \cdots + y^{n-2}x + y^{n-1}) = 0$.

Hence $v_p(x^n - y^n) = v_p(x - y)$■

***Lemma* 2:** Let x and y be (not necessarily positive) integers and let n be a odd positive integer. Given an arbitrary prime p such that: $\gcd(n,p) = 1$,$p \mid x + y$ and neither x nor y is divisible by p. We have $v_p(x^n + y^n) = v_p(x + y)$.

***Proof**:* We will use the fact that $x^n + y^n = (x + y)(x^{n-1} - x^{n-2}y + \cdots - y^{n-2}x + y^{n-1})$. So $v_p(x^n + y^n) = v_p(x + y) + v_p(x^{n-1} - x^{n-2}y + \cdots - y^{n-2}x + y^{n-1})$.

Also from $p \mid x + y$ we have that $x \equiv -y(mod\ p)$.

Therefore

$x^{n-1} - x^{n-2}y + \cdots - y^{n-2}x + y^{n-1} \equiv x^{n-1} - x^{n-2}(-x) + \cdots - x(-x)^{n-2} + x^{n-1} \equiv x^{n-1} + x^{n-1} + \cdots + x^{n-1} + x^{n-1} \equiv nx^{n-1} \not\equiv 0\ (mod\ p)$, in other words $p \nmid x^{n-1} - x^{n-2}y + \cdots - y^{n-2}x + y^{n-1}$, so $v_p(x^{n-1} - x^{n-2}y + \cdots - y^{n-2}x + y^{n-1}) = 0$.

Hence $v_p(x^n + y^n) = v_p(x + y)$■

Lifting the exponent lemma (LTE)

***The First Lemma*:** Let x and y be (not necessarily positive) integers and let n be a positive integer and p be an odd prime such that $p \mid x - y$ and none of x and y are divisible by p. We have $v_p(x^n - y^n) = v_p(x - y) + v_p(n)$.

***The Second Lemma*:** Let x, y be two integers, n be an odd positive integer, and p be an odd prime such that $p \mid x + y$ and none of x and y are divisible by p.

We have $v_p(x^n + y^n) = v_p(x + y) + v_p(n)$.

(***Note***: We can see that the second form of the LTE lemma can be deduced form the first by setting $y := -y$ and setting n to be odd.)

Proof Sketch: The proof of the LTE lemma is similar to the given proof of Hansel's Lemma. Taking care of the case where $\gcd(n,p) > 1$ can be handled by using induction on the number of prime factors of n

We can use ***Lifting The Exponent Lemma*** and ***Hansel's Lemma*** in lots of problems involving exponential equations, especially when we have some prime numbers. The conditions required seem very particular, but with enough experience of problem solving in number theory these lemmas become one of the most important tools. This lemmas shows how to find the greatest power of a prime p in exponential expressions.

The proofs of these lemmas use nothing but simple mathematical proprieties and methods.

Some proofs were left unnoted because understanding the theorems usage and its meaning is more important than remembering its detailed and somewhat long proof.

Example Problems

***Problem* 1. (*Art of Problem Solving*).** Let x, y, p, n, k be natural numbers such that $x^n + y^n = p^k$.

Prove that if $n > 1$ is odd, and p is and odd prime, the n is a power of p.

Solution: Let $g = \gcd(x, y)$. Clearly g is a power of , so dividing both sides by g^n we get the same equation. So we may assume $\gcd(x, y) = 1$, which will give us $x + y$ is divisible by p.

Assume $n = p^t r$ for some natural numbers t, r. If we assume that > 1, by LTE we get $v_p(x^{p^t r} + y^{p^t r}) = v_p(x + y) + v_p(p^t r) = v_p(x + y) + v_p(p^t) = v_p\left(x^{p^t} + y^{p^t}\right) = k$.

So $p^k m = x^{p^t} + y^{p^t} \leq x^{p^t r} + y^{p^t r} = p^k \leq p^k m$. So $m = 1$ and $= 1$, hence n is a power of p ■

***Problem* 2. (*UNESCO contest*, 1995).** Let a, n be natural numbers and p an odd prime, such that $a^p \equiv 1 (mod\ p^n)$. Prove that $a \equiv 1 (mod\ p^{n-1})$

Solution: The statement is equivalent to $a^p - 1$ is divisible by p^n. Clearly $\gcd(a, p) = 1$and $a - 1 \vdots p$.

$$v_p(a^p - 1^p) = v_p(a - 1) + v_p(p) \geq n.$$

So $v_p(a - 1) \geq n - 1 \Leftrightarrow a - 1 \vdots p^{n-1}$■

***Problem* 3. (*Bulgaria* 1997)** Assume that $3^n - 2^n = p^k$ for some natural numbers n, k and a prime p. Prove that n is a prime.

***Solution*:** Suppose for contradiction that $n = qr$.

We get $3^q - 2^q | (3^q)^r - (2^q)^r, so\ p | 3^q - 2^q$.

Applying LTE we get $v_p(3^{qr} - 2^{qr}) = v_p(3^q - 2^q) + v_p(r)$. Hence$p \mid r \Rightarrow 3^p - 2^p | 3^n - 2^n \Rightarrow 3^p - 2^p \equiv 0\ (mod\ p) \Rightarrow 3 - 2 \equiv 0 (mod\ p) (By\ Fermat's\ Theorem)$ which is a contradiction.

***Problem* 4. (*TST Romania* 2009)** Let $a, n \geq 3$ be two positive integers such that there exists a natural number k satisfying

$n \mid (a - 1)^k$. Prove that $n \mid a^{n-1} + a^{n-2} + \cdots + a + 1$.

***Solution*:** $a^{n-1} + a^{n-2} + \cdots + a + 1 = \frac{a^n - 1}{a - 1}$.

Let p be a prime that divides n. $v_p(a^n-1) = v_p(a-1) + v_p(n) \Rightarrow v_p\left(\frac{a^n-1}{a-1}\right) = v_p(n) \Rightarrow n \mid \frac{a^n-1}{a-1}$.

***Problem* 5. (*P4 IMO* 2019):** Find all pairs of positive integers (n,k) such that:

$$k! = (2^n-1)(2^n-2)(2^n-4)\cdots(2^n-2^{n-1})$$

(***Hint***: The main idea is to try to bound the ranges of k and n to reduce the problem to just solving some finite cases.)

***Solution*:** Comparing the v_2 on both sides we get:

$$v_2(LHS) = \sum_{i=1}^{\infty} \left\lfloor \frac{k}{2^i} \right\rfloor \leq k$$

$$v_2(RHS) = 1 + 2 + \cdots + n - 1 = \frac{n(n-1)}{2}$$

Hence $k \geq \frac{n(n-1)}{2}$.

(***Note***. that $v_3(2^{2i}-1) = 1 + v_3(i) (By\ the\ LTE\ Lemma\ 1)$and $v_3(2^{2i-1}-1) = 0$.)

Comparing the $v_3$on both sides we get:

$$v_3(LHS) = v_3(k!) \geq \frac{k}{3}$$

$$v_3(RHS) = \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} 1 + v_3(i) = \left\lfloor \frac{n}{2} \right\rfloor + v_3\left(\left\lfloor \frac{n}{2} \right\rfloor !\right) \leq \frac{n}{2} + \frac{n}{4}$$

$$\frac{k}{3} \leq v_3(k!) = v_3\left((2^n-1)(2^n-2)(2^n-4)\cdots(2^n-2^{n-1})\right) \leq \frac{n}{2} + \frac{n}{4}$$

And finally $\frac{n(n-1)}{2} \leq k \leq \frac{9n}{4}$, meaning$n \leq 4$. Now we can check that the only solutions are $(k,n) = (1,1), (3,2)$. ■

***Problem* 6. (*Shortlist IMO* 2014)** Find all triplets of natural numbers (n,m,p) such that both $x + y^{p-1}\ and\ x^{p-1} + y$ are powers of p, where p is a prime.

Solution: If $= 2$, then any pair $(x, 2^k - x)$ is a solution. If $p \geq 3$ we have that:

If p divides one of $x$or y, then it clearly divides the other. Assume that $v_p(x) = a$ and $v_p(y) = b$ and $a, b > 0$. WLOG assume that $b \leq a$. $v_p(x^{p-1} + y) = b$, but clearly $x^{p-1} + y > p^b$, which is a contradiction.

If p does not divide any of x and y, then we have $x^{p-1} - 1 \vdots p\ and\ y^{p-1} - 1 \vdots p$.

Clearly $x \neq y$, WLOG assume x > y.

Let $p^b = x + y^{p-1}\ and\ p^a = x^{p-1} + y, with\ a > b$. We have that:

$p^b = x + y^{p-1} | x^{p-1} + y \Rightarrow p^b | y^{p-2}(x^{p-1} + y) - x - y^{p-1} = x(x^{p-2}y^{p-2} - 1) \Rightarrow$

$p^b | x^{p-2}y^{p-2} - 1$. But $x \equiv -y^{p-1} (mod\ p^b) \Rightarrow (xy)^{p-2} - 1 \equiv -y^{p(p-2)} - 1\ (mod\ p^b)$

$\Rightarrow$

$p^b | y^{(p-2)p} + 1$. Since $y^{p-2} + 1 \equiv 0\ (mod\ p) \Rightarrow v_p(y^{p(p-2)} + 1) = v_p(y^{p-2} + 1) + 1 \Rightarrow$

$x + y^{p-1} = p^b | p(y^{p-2} + 1) \Rightarrow p(y^{p-2} + 1) \geq p^b = x + y^{p-1} > y + y^{p-1}$

$= y(y^{p-2} + 1)$

$\Rightarrow p > y \Rightarrow y = p - 1$.

$x + y^{p-1} | p(y^{p-2} + 1) = (y + 1)(y^{p-2} + 1) = y^{p-1} + (y^{p-2} + y + 1) < 2y^{p-1} + 2x \Rightarrow x + y^{p-1} = p(y^{p-2} + 1) \Rightarrow p^{b-1} = (p - 1)^{p-2} + 1$.

For $p = 3$ we get $y = 2\ and\ x = 5$. If $p > 3$, we have that $(p - 1)^{p-2} + 1 > p$ and $(p - 1)^{p-2} + 1 \equiv p(p - 2) - 1 + 1 \equiv -2p(mod\ p^2) \Rightarrow (p - 1)^{p-2} + 1$ can not be a power of p.

So finally we have the solutions: $(x, y, p) = (x, 2^k - x, 2), (2,5,3), (5,2,3)$.

***Problem* 7. (*Iran* 2008)** Let a be a natural number such that $4(a^n + 1)$ is a perfect cube for any natural number n. Prove that $a = 1$.

***Solution*:** Set $n = 2k$ for some natural number k. Take some prime divisor of $a^2 + 1, p$.

$3 \mid v_p(4(a^n + 1)) = v_p(a^2 + 1) + v_p(k)$, but clearly we can change the value of k, such that 3 does not divide $v_p(4(a^n + 1))$.

***Problem* 8. (*China TST* 2009)** Let a and b be two natural numbers greater than 1, and b is odd, and n is a natural number such that $b^n | a^n - 1$. Prove that $a^b > \frac{3^n}{n}$.

***Solution*:** Let p be the smallest prime divisor of b ($p \geq 3, since\ b\ is\ odd$).

Then $p^n | b^n | a^n - 1 \Rightarrow n \leq v_p(b^n) \leq v_p(a^n - 1)$.

We also have that $v_p(a^n - 1) \leq v_p((a^{p-1})^n - 1) = v_p(a^n - 1) + v_p(n) = v_p(n(a^n - 1)) \Rightarrow n \leq v_p(a^n - 1) \leq v_p(n(a^n - 1)) \Rightarrow p^n \leq n(a^n - 1) \Rightarrow a^b > a^{p-1} - 1 \geq \frac{p^n}{n} \geq \frac{3^n}{n}$.

Conclusions

Even though number theory is not covered as a topic in school, students that are preparing for national and international contests need to have advanced knowledge in some topics as LTE and Hansel's Theorem. This article is a guideline to the study of these theorems on a variety of contest problems.

References

1. Amir Hossein Parvardi Lifting The Exponent Lemma, 2011. http://services.artofproblemsolving.com/download.php?id=YXR0YWNobWVudHMvYy82LzdjNTI1OGIyMmNjYmZkZGY4MDhhY2ViZTc3MGE1NDRmMzFhMTEzLnBkZg==&rn=TGlmdGluZyBUaGUgRXhwb25lbnQgTGVtbWEgLSBBbWlyIEhvc3NlaW4gUGFydmFyZGkgLSBWZXJzaW9uIDMucGRm
2. Problem Sources: https://artofproblemsolving.com/community
3. Billal M., Parvardi A.H. Topics in Number Theory: An Olympiad Oriented Approach, v. 1.2, 2018.
4. Teleucă M., Pop V., Croitoru D. Probleme de teoria elementară a numerelor pentru concursurile şcolare. Ed. Mega, 2016.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 91-95 E-ISSN 2587-3636

CZU: 372.8574 DOI: 10.36120/2587-3636.v18i4.91-95

METHODOLOGICAL BENCHMARKS FOR PREPARING THE OLYMPIC TEAM FOR ECOLOGY

Gheorghe GÎNJU*, PhD in biology
headmaster of ”M. Marinciuc” High school
Stela GÎNJU, associate professor, PhD in biology
”I. Creangă” Teacher Training University

*Team leader of the national team of Moldova at the International Ecology Olympics - INESPO (Netherlands) and INEPO (Turkey) from 2007 to present; member of the Republican Olympic Council on Ecology.

Summary. The current period is characterized by the presence of various ecological problems. A solution to the recovery of this situation is the ecological education of the young generation, which can be achieved through various ways. An efficient method is that of ecological projects. The article presents the advantages of this method, examples of good practices from the activity of ”M. Marinciuc” high school, which have in its list about 22 medals obtained at the Ecological Olympics using the method of ecological projects.

Keywords: ecological education; ecological scientific projects; research competence; Olympic lot in ecology.

ASPECTE METODOLOGICE DE PREGĂTIRE A ECHIPEI OLIMPICE LA ECOLOGIE

Rezumat. Perioada actuală se caracterizează prin prezența diferitelor probleme ecologice. O soluție pentru redresarea acestei situații este educația ecologică a tinerei generații, care poate fi realizată prin diferite modalități. O metodă eficientă este cea a proiectelor ecologice. Articolul prezintă avantajele acestei metode, exemple de bune practici din activitatea Liceului „M. Marinciuc”, care are în palmares circa 22 de medalii obținute la Olimpiadele Internaționale de Ecologie prin metoda proiectelor ecologice.

Cuvinte cheie: educație ecologică; proiecte științifice ecologice; competență de cercetare; lot olimpic în ecologie.

”What we shall do after we have learned; we learn only by doing”
(*Aristotle*)

Introduction

Currently, in the face of the frequent global ecological problems related to the increasing anthropopression on the natural ecosystems, the ecological education of the population must become a priority for all the states of the world, and the school plays a decisive role in the formation and development of responsible personalities for their actions towards the environment. The European Union's environmental policy is based on the belief that strict environmental norms can provide an incentive for innovation and new opportunities for the development of a human society.

In the Republic of Moldova, ecological education is carried out in several ways:

- Compulsory school subjects (science, biology, chemistry, geography, physics, education for society, personal development, etc.);
- Various optional courses required by students (ecological education, environmental protection, etc.);

- Various school circles (young naturalist, florists, etc.).

Ecological education is not only a form of education, a tool for solving environmental problems or managing natural resources, but it is also a process of essential dimension in recognizing the needs of the environment and defining the concepts of the environment aimed to improve the quality of life [4].

A successful technique with a major impact on the ecological education of the students, especially for the gifted children, represented the scientific researches in the field of ecology are finalized by scientific works with which the students can participate in a series of competitions:

- municipal / district - the scientific conference "Work. Talent.Courtesy." - the ecology department; the municipal / district Olympics for ecology;
- republican - the republican Olympic Games on ecology; the national competition of Science and Engineering "Mold SEF";
- international - the international ecology Olympics "INESPO" (Netherlands) and "INSPO" (Turkey); the international ISEF Science and Engineering competition.

"Learning by doing" - is becoming more and more one of the basic principles of contemporary education and the training of children in scientific research skills is the highest product that the educational system can offer. "Mihai Marinciuc" high school is recognized at national level (certified by the diplomas of the Ministry of Culture and Research Education and the Ministry of Agriculture and the Ministry of the Environment) as "the most creative high school". The scientific works of our students have been appreciated with 22 medals at various International competitions. The results of the Olympic lot of the high school in ecology can be presented in the table 1.

Table 1. Results of the"M. Marinciuc" high school Olympic Ecology Group

Contest	The place obtained		
	1st place	2nd place	3rd place
The International Ecology Olympiad	2 golden medals	8 silver medal	12 bronze medals
The Republican Ecology Olympiad	24 diplomas	16 diplomas	6 diplomas
The Municipal Ecology Olympiad	12 diplomas	8 diplomas	6 diplomas

The formation of scientific research competence in high school is achieved through:

1. Motivating the students to obtain performances and to develop complex skills;
2. Motivating teachers to accumulate credits, to support and to be appreciated by the management staff;
3. Distributing the optional hours and the school circles according to the students' interests;
4. The example given by the administration;

5. Providing the institution with a material base necessary for organizing the scientific conferences of the students;
6. Organizing scientific conferences of students at local level;
7. Collaborations with scientific institutions in the country;
8. Participating in the organization of scientific conferences at national level: Mold-SEF, the Republican Ecology Olympiad.

We consider that one of the most effective methods in motivating students to research is the project method, which we use frequently. The project method was initiated by J. Dewey, supported and popularized by W. Kilpatrick, since the beginning it was based on the principle that "life is an action, not a work in order and that the school, being part of life, must adopt its characteristics [1, 2, 3].

The project method encourages the investigative spirit, enhances self-confidence and improves the attitude towards learning, assumes an increased responsibility towards one's own study, possesses opportunities for training complex competences: higher-level thinking skills, problem solving skills, collaboration and communication skills.

The results presented above do not come by themselves, after a rigorous preparation of a regular work for several years, so within the High School we already have many years in a row an Ecology Olympic Lot and the following stages of the preparation of the Olympic team are:

Stage I - preparatory: selection of gifted children; establishing the research theme; purpose development; setting the objectives of the scientific work.

Stage II- organization: documentation of specialized literature; choice of material and working methods; conducting own investigations; analysis of the obtained results; elaboration of conclusions and recommendations.

Stage III - presentation: poster creation; developing the presentation in Power Point or other tools; trainings with students and presentation at various competitions.

Always, as a basis for the preparation of the internal Olympic team for ecology, we have been guided by the requirements of the International Ecology Olympiad, hereinafter INESPO, which was first organized nationally on May 16, 2009, and since 2010, the Olympics has expanded internationally with over 45 different countries participating each year.

The Cosmicus Foundation of the Netherlands (the founders of this international competition) wants to raise awareness of the young generation about its contribution to the environment by introducing simple and practical scientific projects. INESPO will be the stage in which the next generation will exercise their knowledge and love for science, contributing to a solution for different environmental problems.

All the registered scientific projects go through a preliminary round of elimination due to the large number of registrations, and in the elaboration and presentation of the scientific project the judging criteria presented must be taken into account [5], in Table 2.

Table 2. Judging criteria at INESPO

Criteria	Criteria Details	Point
1. Quality of poster	*Well-organized the research and project log.*	10
2. Quality of presentation	*Clear explanation of the project, research method, results and conclusion.*	10
3. Creativity/Originality	*Originality of the problem, unique approach to solving a societal problem.*	10
4. Feasibility	*Practical applications of the project; economic and technological consistence, does the project contribute to the society in some way.*	10
5. Literature review	*Have scientific literature and references been used during research?*	10
6. Scientific Thinking	*Do students have a clear statement of the hypothesis? Are the goal and the identification of all relevant variables clearly defined?*	10
7. Scientific Methods/ Data management	*The students must prove that they are well aware of the contents of the substance, followed by appropriate methods for experiments and investigations.* *Proper recording and playback of data in tables and graphs, proper analysis of the data.*	10
8. Conclusion	*Sketches of logical conclusion, the consistency of the statement with acquired results, recommendation for further investigation.*	10
9. Research Skills and Effort	*Level of skills and efforts of (every) researcher to carry out the project of work, knowledge of techniques and equipment used to collect data.*	10
10. Insight / Understanding of Project	*Insight of (every) student into each step during the implementation of the project.*	10

We urge all the animators of the Olympic groups to take into account the requirements of INESPO, because since 2017 in the Republic of Moldova they have been taken as a basis to organize the Republican Olympics for Ecology. Unfortunately, we

would like to mention, that at the republican stage, not all the districts of the country are included and we want a more active participation of the children from the country in this contest.

Conclusion

Regardless of the way in which ecological education is carried out, it is important to execute it at the level of humanity, because it is a "process meant to attract categories of people who are aware and concerned about environmental issues and complementary problems, people who have the knowledge, the attitude, the ability, motivation and ability to work individually and collectively to find solutions to current problems but also to prevent the emergence of others".

Bibliography

1. Cerghit I. Metode de învățământ. Iași: editura Polirom, 2006. p. 123.
2. Constantinescu I., Stolearu A. Creativitate, cunoaştere, potenţial creativ. Timişoara, 1991. p 253.
3. Bîrcă M., Aghenie N., Prunici V. Metoda proiectelor de cercetare ştiinţifică a elevilor la chimie. http://dspace.usm.md:8080/xmlui/handle/123456789/230
4. Gînju S. Educația ecologică în diverse țări ale lumii. Studiu comparativ. În: Probleme ale științelor socioumanistice și Modernizării Învățământului, materialele Conferinței științifice anuale a profesorilor și cercetătorilor UPS "Ion Creangă": Seria XIX, volumul II, coord. șt. Racu I. Chișinău: S.N. Tipogr. UPS "Ion Creangă", 2017. p. 28-33.
5. http://www.inespo.org.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 96-103 E-ISSN 2587-3636

CZU: 004.02 DOI: 10.36120/2587-3636.v18i4.96-103

A NON-STANDARD METHOD OF SOLVING COMPUTATIONAL GEOMETRY PROBLEMS

Andrei BRAICOV, associate professor, PhD

Tiraspol State University

Summary. Computational geometry problems are often encountered in programming contests. In this article we examine some problems that, traditionally, are solved using vectors. Another approach to finding solutions is proposed, less commonly found in the specialized literature.

Keywords: competitive problems, computational geometry, programming, original solutions.

O METODĂ NON-STANDARD DE SOLUȚIONARE A PROBLEMELOR DE GEOMETRIE COMPUTAȚIONALĂ

Rezumat. Problemele de geometrie computațională se întâlnesc deseori în concursurile de programare. În acest articol sunt examinate câteva probleme care, tradițional, se rezolvă utilizând vectorii. Se propune o altă abordare de căutare a soluțiilor, mai puțin întâlnită în literatura de specialitate.

Cuvinte cheie: probleme competitive, geometrie computațională, programare, soluții originale.

Introduction

Computational geometry problems are often encountered in programming contests at different levels. A series of notions and formulas from analytical geometry are used to solve them.

In the specialty literature there are several standard problems with their solutions: the problem of convex windings; the problem of triangularizations; proximities and belongings etc. [1, 2].

It is obvious that knowledge of analytical geometry (Cartesian coordinates, polar coordinates, the equation of the straight line, coordinate transformations, geometric transformations, vectors, relative positions, intersections, etc.) is required to solve them.

Problems related to the positions of the point relative to another geometric figure are solved starting from the notion of the position of the point relative to a vector, which, for some students, is a difficult subject to understand.

In the following, we will try to describe how to solve some problems of computational geometry related to the positions of the point against another geometric figure without using the notion of vector.

Problem 1. Point and polygon. The natural number n, $n < 100$, the Cartesian coordinates of point M and the Cartesian coordinates of n points A_1, A_2, … A_n, representing the vertices of the convex polygon A_1A_2 … A_n, are given. Determine an algorithm that verifies whether or not the point M belongs to the mentioned polygon.

Solution:

For the point M to belong to the polygon $A_1, A_2, \ldots A_n$ it must:

- coincide with one of the points of the polygon or;
- belong to one of the segments $[A_i, A_{i+1}]$, where $i = 1, 2, \ldots n$ and $A_{n+1} = A_1$.

The point M belongs to the segment $[A_i, A_{i+1}]$ if $d(A_i, M) + d(M, A_{i+1}) = d(A_i, A_{i+1})$. The second case also includes the first.

The complexity of the algorithm is $O(n)$.

Problem 2. Convex polygon. The natural number n, $n < 100$, and the Cartesian coordinates of n points $A_1, A_2, \ldots A_n$, representing the vertices of the polygon $A_1A_2 \ldots A_n$, are given. Write an algorithm that checks whether or not the mentioned polygon is a convex polygon.

Solution:

It is known that the polygon $A_1A_2 \ldots A_n$ is convex only if all its vertices (except for the points A_i and A_{i+1}) belong to the same semiplane determined by any of the line $A_i A_{i+1}$, where $i = 1, 2, \ldots n - 1$.

It can be shown that the requirement in the definition can be "simplified" as follows:

Definition 1. Polygon $A_1A_2 \ldots A_n$ is a convex polygon only if for any two vertices A_j, A_{j+1}, points A_{j-1} and A_{j+2} will belong to the same semiplane determined by the segment $A_j A_{j+1}$, where $j = 1, 2, \ldots, n$, and $A_0 = A_n$, $A_{n+1} = A_1$, $A_{n+2} = A_2$.

In solving the problem we will use the notion of *deviation of a point from a straight line*, a notion less commonly found in the specialized literature.

Definition 2. Let d be a line with the equation $ax + by + c = 0$ and the point $M(m_1, m_2)$. The number $am_1 + bm_2 + c$ is called *the deviation of the point M from the line d*, which we note $Ab(M, d)$.

The following Lemma can be easily demonstrated.

Lemma. Points M and N belong to the same semiplane determined by line d if and only if

$$Ab(M, d) \text{ and } Ab(N, d) \text{ have the same sign. (1)}$$

The relationship (1) can be written as follows:

$$Ab(M, d) \times Ab(N, d) > 0. \text{ (2)}$$

Therefore, taking into account definition 2 and relation (2), we conclude that for the mentioned polygon to be convex the condition must be met:

$$Ab(A_{j-1}, A_j A_{j+1}) \times Ab(A_{j+2}, A_j A_{j+1}) > 0, \text{ (3)}$$

for any $j = 1, 2, \ldots, n$, where $A_0 = A_n$, $A_{n+1} = A_1$ and $A_{n+2} = A_2$.

If we define the type

```
struct t_point {
    double x, y;
} TPoint;
```

then the function for determining the deviation of the point *X* from the line side *YZ* can be defined as follows:

```
double ab(TPoint X, TPoint Y, TPoint Z)
{
 return (Y.y - Z.y) * X.x + (Z.x - Y.x) * X.y + Y.x * Z.y - Y.y * Z.x;
}
```

Thus, the complexity of the algorithm was reduced to O(*n*).

Problem 3. The interior of the polygon [3]. The natural number *n*, $n < 100$, the Cartesian coordinates of point *M* and the Cartesian coordinates of *n* points $A_1, A_2, \ldots An$, representing the vertices of the convex polygon $A_1A_2 \ldots A_n$, are given. Write an algorithm that determines whether or not the point *M* belongs to the interior of mentioned polygon.

Solution:

We will create the solution of the problem using the same notion as in the case of problem 2, i.e. the deviation of a point from a straight line.

In this case must be verified the relationship:

$$Ab(M, A_i A_{i+1}) \times Ab(A_{i+2}, A_i A_{i+1}) > 0,$$

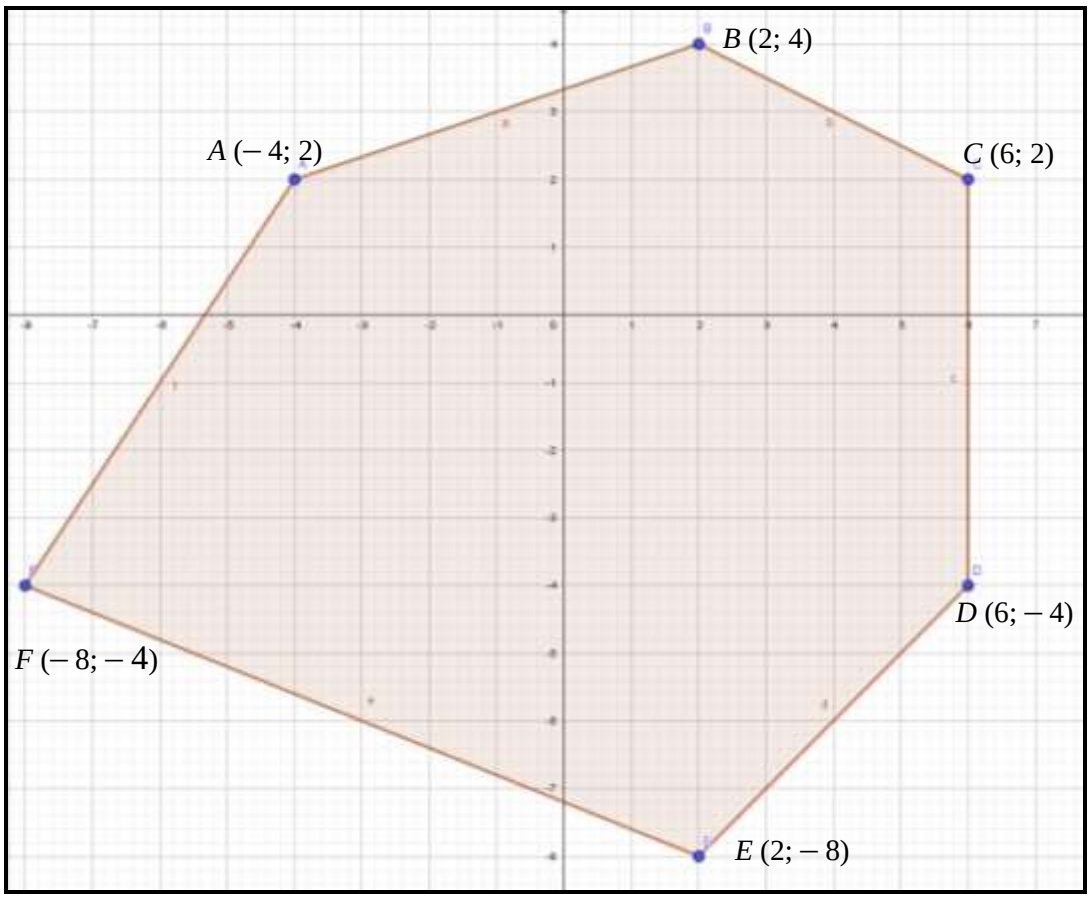
for any $i = 1, 2, \ldots, n$, where $A_{n+1} = A_1$ and $A_{n+2} = A_2$.

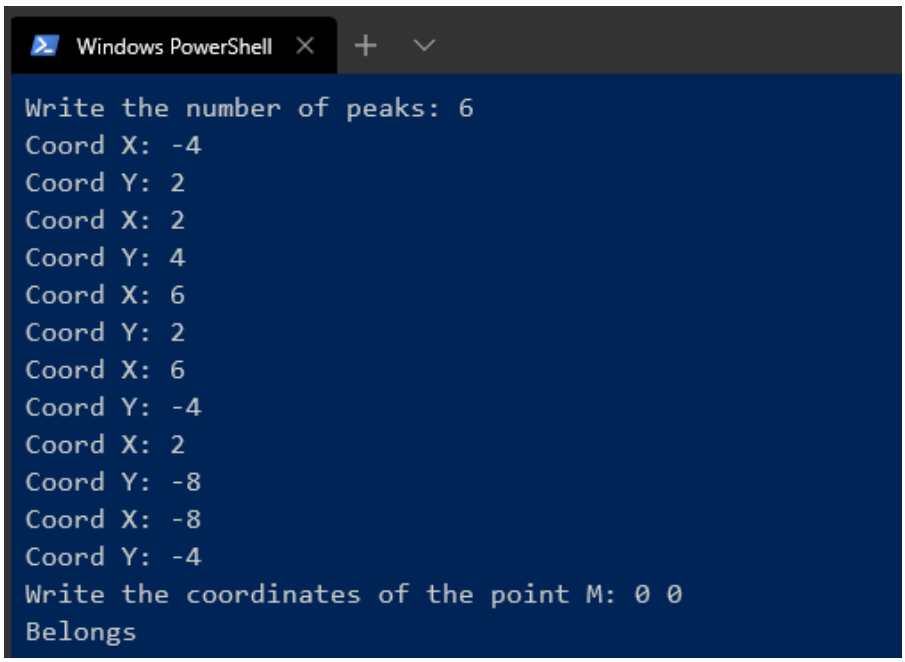
In this case the complexity of the algorithm is also O(*n*).

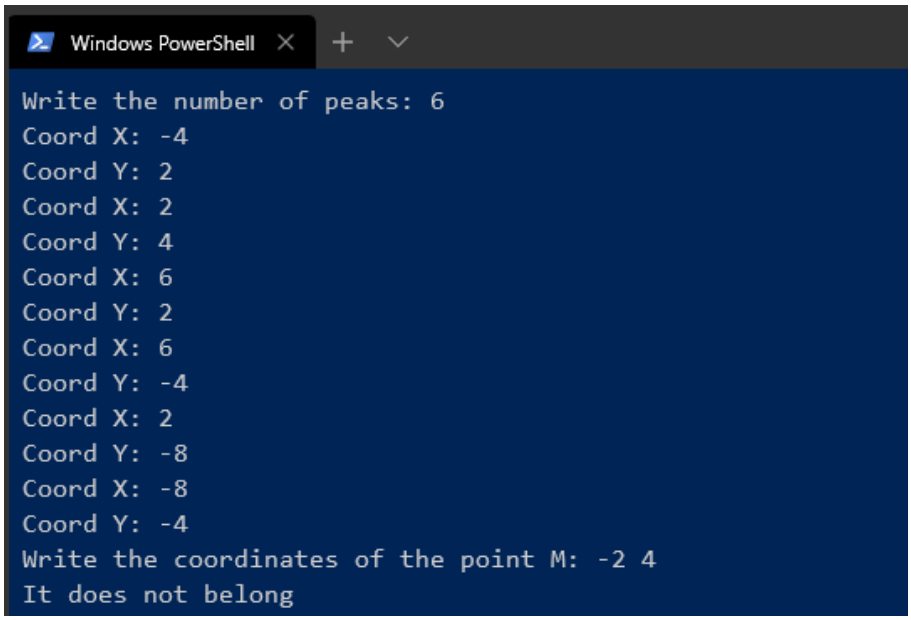
```
#include <iostream>
using namespace std;
typedef struct t_point {
    double x, y;
} TPoint;
//Deviation of a point X from a straight line YZ
double ab(TPoint X, TPoint Y, TPoint Z)
{
 return (Y.y - Z.y) * X.x + (Z.x - Y.x) * X.y + Y.x * Z.y - Y.y * Z.x;
}
int main()
{
    int n;
    TPoint M;
    TPoint *A;
    bool f;
    cout << "Write the number of peaks: ";
    cin >> n;

    A = new TPoint[n + 2];
    for (int i = 0; i < n; ++i) {
        cout << "Coord X: ";
        cin  >> A[i].x;
        cout << "Coord Y: ";
```

```
        cin  >> A[i].y;
    }
    cout << "Write the coordinates of the point M: ";
    cin  >> M.x >> M.y;
    f = true;
    // We add the first 2 vertices to the vector
    A[n].x = A[0].x;
    A[n].y = A[0].y;
    A[n + 1].x = A[1].x;
    A[n + 1].y = A[1].y;
    for (int i = 0; i < n; ++i) {
        if (ab(M, A[i], A[i + 1]) * ab(A[i+2], A[i], A[i+1]) <= 0) {
            f = false;
        }
    }
    if (f) {
        cout << "Belongs";
    } else {
        cout << "It does not belong";
    }
    return 0;
}
```







Problem 4. The outside of the polygon. The natural n, $n < 100$, the Cartesian coordinates of point M and the Cartesian coordinates of n points A_1, A_2, … A_n, representing the vertices of the convex polygon $A_1A_2 \ldots A_n$, are given. Write an algorithm that determines whether or not the point M belongs to the outside of the mentioned polygon.

Solution:

The search for the solution of this problem includes the methods applied to solve problems 1 and 3.

Thus, we will verify that at least one of the two relationships is respected:

1) there is at least one i, where $i = 1, 2, \ldots, n$, and $A_{n+1} = A_1$, $A_{n+2} = A_2$, such that:

$$Ab(M, A_i A_{i+1}) \times Ab(A_{i+2}, A_i A_{i+1}) < 0;$$

2) there is at least one i, where $i = 1, 2, \ldots, n$, $A_{n+1} = A_1$ and $A_{n+2} = A_2$, such that: $Ab(M, A_i A_{i+1}) \times Ab(A_{i+2}, A_i A_{i+1}) = 0$ and $d(A_i, M) + d(M, A_{i+1}) > d(A_i, A_{i+1})$.

In this case the complexity of the algorithm is also $O(n)$.

Problem 5. The area of the polygon. We give the natural number n, $n < 100$, the Cartesian coordinates of n points A_1, A_2, … A_n, representing the vertices of a simple polygon (the edges of the polygon do not intersect). Write an algorithm that calculates the area of the polygon $A_1A_2 \ldots A_n$.

Solution:

a) If $A_1A_2 \ldots A_n$ is a convex polygon (this condition can be verified, see the solution of Problem 2), then the problem will be solved by triangulating the polygon ([1, page 29]), then summing the areas of all the obtained triangles.

b) The problem is more difficult if the polygon $A_1A_2 \ldots A_n$ is concave.

In this case it results that there is at least one i, where $i = 1, 2, \ldots, n$, $A_0 = A_n$, $A_{n+1} = A_1$ and $A_{n+2} = A_2$, that $Ab(A_{i-1}, A_i A_{i+1}) \times Ab(A_{i+2}, A_i A_{i+1}) < 0$. (Figure 1).

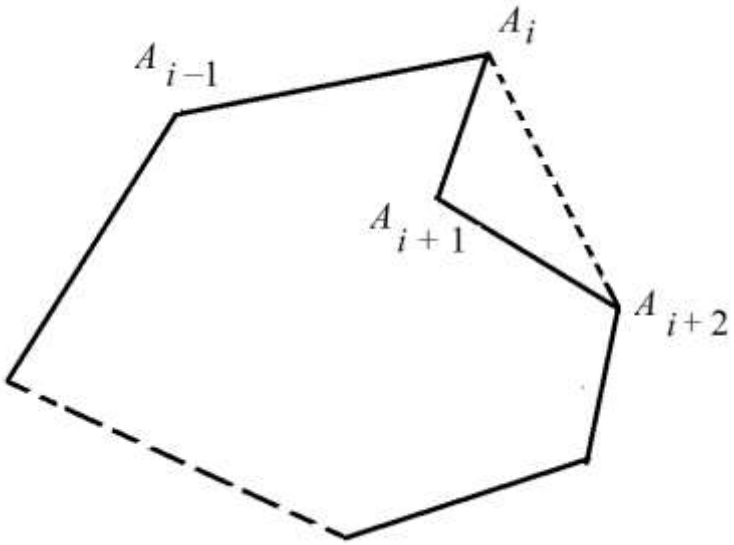


Figure 1. The case of the concave polygon

Figure 1 suggests the following algorithm:

1. "Exclude" the vertex A_{i+1}. The obtained polygon will have the area greater than the polygon $A_1A_2 \ldots A_n$, the difference being equal to the area of the triangle $A_i A_{i+1} A_{i+2}$. We store the area of this triangle in a vector S.

2. We use a "queue" in which we store all the points of the given polygon except the "excluded" vertices. Let $B_1B_2 \ldots B_m$, where $m \leq n$, be the obtained polygon (extracted from "queue").

3. The area of the polygon $A_1A_2 \ldots A_n$ is equal to the area of the polygon $B_1B_2 \ldots B_m$ minus the sum of the elements of the vector S. The polygon $B_1B_2 \ldots B_m$ is convex, so we reduced the problem solving to the case a).

This algorithm is not valid for any polygon!

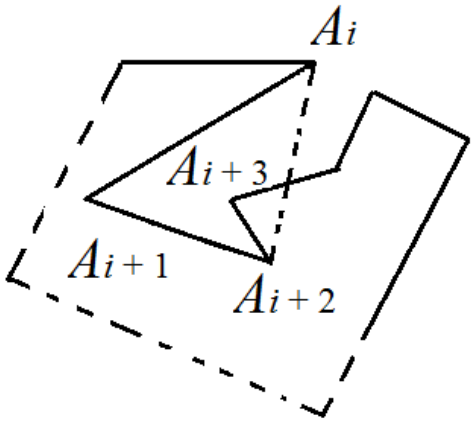


Figure 2. The case 2 of the concave polygon

From figure 2 we notice that if we "exclude" the peak A_{i+1}, then the obtained polygon is not simple, so the described algorithm is not valid.

It can be shown (see, for example, [4, page 258]) that the most "convenient" algorithm for calculating the area of polygon $A_1A_2 \ldots A_n$ is the calculation of the expression value:

$$|0{,}5 \sum_{i=1}^{n}(x_i(y_{i+1} - y_{i-1}))|,$$

where $A_{n+1} = A_1$, $A_0 = A_n$, and x_i and y_i – the coordinates of the point A_i.

Problem 6. The intersection of two segments. The Cartesian coordinates of 4 points P_1, P_2, P_3, P_4 are given. Determine an algorithm that checks if the segments $[P_1P_2]$ and $[P_3P_4]$ intersect.

Solution:

In [4, page 254] the solution of this problem is presented in two stages:

1. *Rapid rejection test*. If the rectangle with the diagonal $[P_1P_2]$ does not intersect with the rectangle with the diagonal $[P_3P_4]$, then neither the segments $[P_1P_2]$ and $[P_3P_4]$ intersect.

2. *Checking of intersection*. If the rejection test is "not passed", then for segments $[P_1P_2]$ and $[P_3P_4]$ to intersect, each of the two segments must intersect the line containing the other segment.

Another way to solve this problem:

1. *Rapid rejection test*. If the lines P_1P_2 and P_3P_4 do not intersect, then neither the segments $[P_1P_2]$ and $[P_3P_4]$ intersect.

2. Let the rejection test be "not passed" and $M = P_1P_2 \cap P_3P_4$. The segments $[P_1P_2]$ and $[P_3P_4]$ intersect if and only if point M belongs to the segment $[P_1P_2]$, that is, only if the relation $(M, P_1) + d(M, P_2) = d(P_1 P_2)$ occurs.

Conclusions

Computational geometry highlights the applicative role of mathematics. In the context of competitive informatics, it strengthens the motivation and the need for a deep study of analytical geometry (in plan and in space).

Obviously, the skills of solving the problems of computational geometry increase the chances of success in the contests and the programming Olympics.

The didactic approaches to study this field must be folded on the degree of understanding of the new concepts by the students. If the use of vectors has difficulties, at the first stage problems should be analyzed for which solutions can be found without vector calculations. Subsequently, one can return to vectors, which represent an efficient tool for solving problems with high degree of difficulty.

Bibliography

1. Corlat S. Metodologia rezolvării problemelor de geometrie computaţională. Chișinău: Tipografia UST, 2013.
2. Ласло М. Вычислительная геометрия и компьютерная графика на C++. Москва: Бином, 1997.
3. Braicov A. Informatică. Turbo Pascal. Culegere de probleme. Editura *Prut Internațional*, 2007. 232 p. ISBN 9975 – 69-788-7.
4. Cerchez E., Șerban M. Programarea în limbajul C/C++ pentru liceu (volumul III). Iași: Editura Polirom, 2006. 296 p.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 104-108 E-ISSN 2587-3636

CZU: 004.02 DOI: 10.36120/2587-3636.v18i4.104-108

THE TRICK FROM ALIENS IN COMPETITIVE PROGRAMMING

Șerban CERCELESCU

International Computer High-School, Bucharest, Romania

Abstract. The scope of this article is presenting a very useful DP optimization technique, introduced in the competitive programming community with the problem Aliens at IOI 2016. The technique is used to reduce dimensions in particular DP configurations, by exploiting the convex nature of some cost functions. We will introduce the technique by starting with a simpler DP problem, show the optimization from $O(N^2)$ to $O(NlogVAL)$ then reveal the full solution of the original problem. Apparently, the official name of this optimization technique is "parameter search" and the Chinese community calls it "wqs binary search".

Keywords: dynamic programming, parameter search, competitive programming.

TRUCUL PROBLEMEI ALIENS ÎN PROGRAMAREA COMPETITIVĂ

Rezumat. Scopul acestui articol reprezintă ilustrarea unei tehnici de optimizare a DP foarte utilă, introdusă în comunitatea de programare competitivă odată cu problema Aliens la Olimpiada Internațională de Informatică din 2016. Tehnica este utilizată pentru a reduce dimensiunile în anumite configurații DP, prin exploatarea caracterului convex al unor funcții de cost. Vom introduce această tehnică începând cu o problemă DP mai simplă, vom arăta optimizarea de la $O(N^2)$ la $O(NlogVAL)$, apoi vom dezvălui soluția completă a problemei originale. Aparent, denumirea oficială a acestei tehnici de optimizare este „căutare de parametri", iar comunitatea chineză o numește „căutare binară wqs".

Cuvinte cheie: programare dinamică, căutare de parametri, programare competitivă.

A problem example: You are given an array *v* of integers (possibly negative) of length *N* ($\leq 10^5$) and a number *K* ($\leq N$). Select at most *K* disjoint sub-arrays of the initial sequence such that the sum of the elements included in the sub-arrays is maximized. The standard approach to such a problem would be a DP of the form:

$dp[n][k]$= ["*the solution for an array with the first n elements of the given array and k sub-arrays to be taken*"] where

$$dp[n][k] = max\{dp[n-1][k], max_{i=k}^{n-1}\{dp[i-1][k-1] + \sum v_k\}\}$$

Implementing this recurrence directly would be $O(N^4)$, supposing that *K* is comparable in size to *N*. It is left as an exercise to the reader to find a way of optimize this recurrence to $O(N^2)$. The trick behind the "aliens optimization" is that we can add a cost (penalty) which we will denote by λ for each taken sub-array. If λ=0, then the solution would be taking a sub-array for each positive element, but by increasing the value of λ, the optimum solution shifts to taking fewer sub-arrays. Now we just have to find a λ that allows us to take as many sub-arrays as possible, but still fewer that *K*. To do a small recap, λ is the cost we assign to adding a new sub-array, and increasing λ will decrease the number of sub-arrays in an optimal solution or keep it the same, but never increase it. That suggests that we could just binary search the smallest value of λ that yields an optimal solution with less than *K* elements.

$dp_\lambda[n]$=[“*The solution for the prefix of length n of our initial array v, where adding a sub-array comes with cost λ*”]

$$dp_\lambda[n] = max\{dp_\lambda[n-1], max_{i=1}^{n-1}\{\sum v_k + dp_\lambda[i-1] - \lambda\}\}$$

Besides just the dp, we will store another auxiliary array:

$cnt_\lambda[i]$ = [“how many sub-arrays does $dp_\lambda[n]$employ in its solution”]

These recurrences are easily implementable in linear time using partial sums and maxima. The pseudocode behind all of it would go something like this:

```
minbound = 0, maxbound = 1e18
while maxbound - minbound > ε:
   λ = (maxbound + minbound) / 2
   #compute dp and aux for λ
   if cnt[n] <= k:
      minbound = λ
   else:
      maxbound = λ
#compute dp and cnt for the final λ (= final minbound)
return dp[n] + cnt[n] * λ #note that if there are less than k positive values, then cnt[n] < k
```

Proof and Formal Requirements: In the case of our initial problem, the fact that increasing λ never increases the number of sub-arrays taken was probably a very intuitive fact, but we'd like to find an actual proof that this works and find a general criterion for using the peak setting optimization in reducing DP dimensions. This criterion is in a way concavity (or convexity). Let's denote by *ans*[*k*] the answer for the problem, but using exactly *k* sub-arrays. The key observation in proving that our solving method is correct is that the *ans*[*k*] sequence is concave, that is $ans[k] - ans[k-1] \leq ans[k-1] - ans[k-2]$. A more natural way of thinking about this and the actual way most people “feel” the concavity/convexity is by interpreting it as *if I have k sub-arrays and add another one, it will help me more than if I had k* + 1 *sub-arrays and added another one*.

Now let's see how this concavity helps us prove the correctness of our solution. Suppose λ=0. Our solution will just find the global maximum of our concave sequence, be it *ans*[*k*]. Notice that no matter the value of λ, the fact that our sequence is concave won't change. Let's shift our attention for a bit from concave sequences to concave functions. $f(x) = \lambda x - x^2$ is a fine example. By changing λ, we can move the peak of the function wherever we want and the function will remain concave.

Now let's go back to our more “discrete” sequence. We have an algorithm that finds *p* and *ans*[*p*] such that *ans*[*p*] is the maximum of the sequence, but we don't want the maximum of the sequence, we want *ans*[*k*] for some given *k*. So... we can force *k* to be the maximum of the sequence, by adding a linear function to our sequence (*ans*[*k*] → *ans*[*k*] + λ*k*), just as we changed the peak of our continuous function, we can forcefully change the peak of our sequence, which is exactly what we did in our solution.

The algorithm will yield that the maximum of the sequence is at *k* with the value *ans*[*k*] + λ*k* and we just need to subtract λ*k* to obtain our desired value: *ans*[*k*]. As for the

general criterion, you might have already guessed it: if $(ans[k])_{1 \le k \le n}$ is the sequence of answers for given *k*s, the sequence must be convex or concave, that is:

$$\forall i \in (1 \ldots n), ans[i] - ans[i-1] \le ans[i+1] - ans[i]$$

or

$$\forall i \in (1 \ldots n), ans[i] - ans[i-1] \ge ans[i+1] - ans[i]$$

Reconstruction Issues: Let's get back to our initial problem (there are *N* integers, you have to choose *K* sub-arrays etc...) and let's change the statement, instead of selecting at most *K*, you have to select exactly *K* sub-arrays. The difference is quite subtle, and the actual result is different iff there are less than *K* non-negative integers in the sequence. In this case, we just have to replace return $dp[n] - \lambda * aux[n]$ with return $dp[n] - \lambda * k$. This may seem weird and quite unintuitive as for why it works. Let's look at a few proprieties of our algorithm. First of all, it may not be the case that for each *k* we have a corresponding set of lambdas, that is: if for a given *p*, the maximum λ for which taking *p* objects is optimal, then the solution for $\lambda \leftarrow \lambda + \varepsilon$ where ε is an arbitrarily small value, may use more than *p*+1 objects, i.e. there may be no choice of λ for which the optimal solution employs a fixed number of elements. This may seem as a small game-killer for our technique, but let's look at the cause of this issue. Looking back at the Proof paragraphs, we are given the condition:

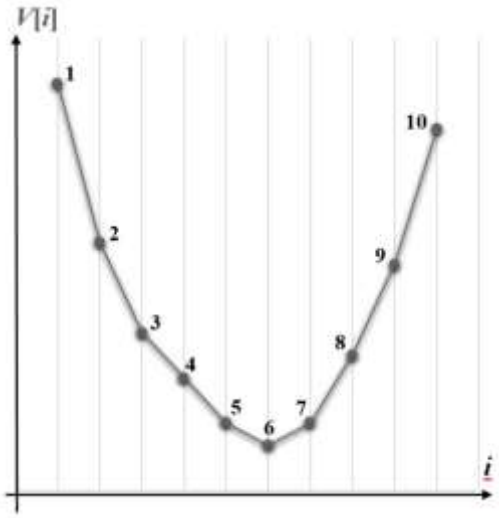
$$\forall i \in (1 \ldots n), ans[i] - ans[i-1] \le ans[i+1] - ans[i]$$

In case of equality, we may have the following situation: $ans[i+1] = ans[i] + t$, $ans[i+2] = ans[i] + 2t$etc. If the λ we choose equals *t*, then all of these solutions will seem equivalently "good". In fact, if a sub-array of solutions $ans[a], ans[a+1], \ldots, ans[b]$ for an arithmetic progression, there is no choice of λ that finds any other optimal solution other than using *a* or *b* objects. However, the fact that our solution fails only on possible arithmetic progressions from our sequence (i.e. if the sequence is not *strictly convex*) is the very thing that will help us solve this issue. Suppose we find the smallest lambda that makes the solution employs $\le k$ objects (let's say it uses *a* objects). This means the answer using exactly *p* objects is $dp[p] - \lambda p$, but this basically implies that between *p* and *k* (the fixed number of objects we want to use) there is an arithmetic progression (i.e. $ans[k] = ans[p] + t*(k-p)$). So if the answer for *p* would be $dp[n] - p\lambda$, then the answer for *k* must be $dp[n] - p\lambda - (k-p)\lambda$ (which equals $dp[n] - \lambda k$). This is quite weird as by finding a solution for *p*, we also find the answer for *k*, even if $(aux[n] = p) \ne k$. The downside of this workaround, is that even if we can find the value of the answer, a general method of reconstructing the solution (finding out what sub-arrays should we select) may not always exist.

Integral Lambda Search: You might have noticed that we are binary searching a floating point λ, not an integral valued one. The reason is that if the prerequisites of applying the optimization are satisfied, then we have proved that a λ exists, not that it

would have an integral value. The thing is, in most DP problems, the optimization works just as well with integers. It's just not that obvious to prove why.

Let's consider a convex sequence of N elements, call it $V_{1...N}$. Now let's consider a set of points $P=\{(i, p_i) \mid i \in [1...N]\}$, once drawn, together with the line segments between consecutive points (which will bear great importance in the following steps), you will see a convex/concave lower/upper hull.



A key observation now is that when looking at our convex sequence geometrically, the "peak" of the sequence will be the unique point that has segments with different signs of the slope to its left and to its right (with the exception the edge cases where the optimum is the first or last element of the sequence). In our drawn example, that peak is the 6th point, with a segment with negative slope value on its left and positive on its right. Another useful observation is that if we have two lines$y = a_1x + b_1$and $y = a_2x + b_2$, adding a constant value λ to both their slopes doesn't change the x coordinate of their intersection abscise.

In our context, this means that if we add a constant value to all the slopes of the segments, the intersection points will remain the same, so the peak x coordinate will still be integral. Now all that is left to do is "say" this: if we want to force a position t to be the global optimum of this sequence and the slope of the segment to the left of the point is l and the slope of the segment right of the point is r and $\{l,r\} \in Z$, then there exists at least one value $\lambda \in Z$ such that $l + \lambda$ and $r + \lambda$ have different signs. Translating this directly into terms of our convex sequence of answers, where our "slopes" are just the differences between two adjacent answers (i.e. $slope[k] = ans[k + 1] - ans[k]$), if the values of ans are integral, then obviously the differences (slopes) will also be integral, so if the answers to our problem are integral, then we can always binary search λ as an integral value.

Aliens – IOI 2016: I find it quite amusing that when the IOI introduces some totally new technique for 99% of its contestants (like the convex hull trick with the problem *batch scheduling* at IOI 2002), the technique is usually merely a sub-problem of a task that is

quite difficult on its on own, even without the fact that the contestant is required to rediscover some then obscure technique.

So is the case with Aliens (IOI 2016), even if you know the optimization, it's still quite a tricky convex hull trick problem.

The solution goes something like this:

First of all, notice that if a point lies below the main diagonal, we can just replace it with its transpose (i.e. $(x, y) \rightarrow (y, x)$), as any photo that captures (x, y) will also capture (y, x). After this transformation, notice the fact that we might be left off with a lot of useless points, that if removed will not change our answer. That is because if we have two points $p_2 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ such that $x_1 \leq x_2$ and $y_1 \geq y_2$, any photo that captures p_2 will also capture $p1$, but not all photos capturing p_2 will capture p_2 and given that we must capture all points (so p_2 must be captured), we might as well remove $p1$ because it would have been captured by the photo containing $p2$ anyway. We can remove these useless points using a stack, and we'll be left with a sequence of points (x_1, y_1), (x_2, y_2),…, (x_b, y_b) that if sorted in increasing order by the x coordinate, the sequence will also be sorted in decreasing order by the y coordinate. From now on, we will consider the sequence of points in this order.

Let's define $dp[n][k]$ = {["*minimum area of a square that covers all points from t + 1 to n*"] +

$dp_\lambda[t]$ - ["*the area of the intersection between the square and the ones used in* $dp_\lambda[t]$"] + λ}, which is $dp_\lambda[n] = min_{t=1}^{n-1}\{(x_n - y_n + 1)^2 + dp_\lambda[t] - max(x_t - y_{t+1}, 0)^2 + \lambda\}$, which can be computed in linear time using the convex hull optimization.

Conclusions

"The aliens trick" is a very powerful optimization technique that most probably will soon be widespread in the competitive programming world, exploiting the convex nature of the solution space of some problems. I see this optimization, as a vital technique in the toolkit of all future and current competitive programmers.

Bibliography

1. Meenakshi K. R. Dynamic Programming for Coding Interviews: A Bottom-Up approach to problem solving. 1st Edition. Kindle Edition, 2017. 144 p.
2. Lew A., Mauch H. Dynamic Programming. A Computational Tool. Springer, 2007. 377 p.
3. https://ioinformatics.org/files/ioi2016problem6.pdf
4. https://www.hackerearth.com/practice/algorithms/dynamic-programming/introduction-to-dynamic-programming-1/tutorial/
5. https://www.tutorialspoint.com/data_structures_algorithms/dynamic_programming.htm

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 109-113 E-ISSN 2587-3636

CZU: 378.147:57/59 DOI: 10.36120/2587-3636.v18i4.109-113

TEACHING ASPECTS ON THE EXAMINATION OF STUDENTS PERFORMANCES IN SOME BIOLOGICAL DISCIPLINES

Eugenia CHIRIAC, associate professor, PhD
Tiraspol State University

Abstract. This work examine some teaching methods on students performance assessment from biologists specialties: biology- chemistry, biology and ecology**.**

Keywords: performance, biology, education.

ASPECTE DIDACTICE PRIVIND EXAMINAREA PERFORMANȚELOR STUDENȚILOR LA DISCIPLINE BIOLOGICE

Rezumat. În acest articol se examinează câteva metode didactice de evaluare a performanței studenților la specialitățile biologice: biologie-chimie, biologie și ecologie.

Cuvinte cheie: performanță, biologie, educație.

1. The student and information dominance

Currently, it is trying to place education in a new socio-cultural context, and the student from the higher education school is oriented towards choosing, from several alternatives, those that lead to the autonomy gain in personal action. A first element that facilitates this in terms of applicability is the concept of performance and its pedagogical values. Thus begin to outline even to achieve some of the essentials goals, such as: transmission of knowledge and forms of knowledge that allow the student formation of realistic and rational vision of the world; facilitating the achievement of each student's unique potential through skills training; preparation of new specialists for the needs manifested in economic, social and cultural. Results of education aimed at personality that has formed conscience affirm social interest through variety behaviors and attitudes. Therefore, education stands out as a value, which responds to a complex of existential problems, the performance has an important role.

Performance study placed in the context of the new realities of modern pedagogical information generated dominance, and students within an educational space defined by the organization and connecting information in a performative perspective, in an educational environment near the complex reality than they are formed. Although we are currently talking about the centrality of the subject in the formative action, capital conceptual education is not sufficient evidence for the independence and freedom of students to translate into behaviors, skills and performance [1-6].

2. Performance and learning

Achieving academic performance by students is possible but it involves a significant pedagogical effort for the elaboration of the appropriate methodological landmarks and the efficient use of the possibilities available to them.

Based on the above, the following served as purpose observing the students' performance in the following biological disciplines: Plant anatomy and morphology; *Plant systematics.*

In this connection we have developed the following objectives:

- examining the performance categories as a reference point at which the student's purchases are articulated in the educational activity;
- establishing the functional differences between competence (as a possible behavior) and performance (as a real, observable operative behavior);
- researching the performance from the perspective of the result that the vast majority of students have to reach in the biological disciplines in the study.

3. Results and discussions

The research was carried out within the Faculty of Biology and Chemistry. The experiment was conducted in several groups of students with different specialties: Biology and Chemistry -10 students; Biology -10 students; Ecology -10 students. In the experimental part it was taken into account: analysis of situations, comparison and generalization of association, collaborative learning activities, the Zig-Zag method, Brainstorming method [1-3].

The control experiment was carried out in the groups of year I. Of the existing forms, the lesson was considered the basic form of the education process, and the teacher - coordinator of the didactic action. They made observations on student activity and meetings Botanical circle.

The two student samples the experimental sample and the control sample - were created so that the differences of success are not explained by differences of pedagogical conditions, personality traits, instrumental difficulties. The participating students were part of three specialties of university education, and the experiment was carried out at the level of some biological disciplines, starting from the Plan and Curriculum of studies, describing the levels of development of students' communicative competences, necessary and possible to be achieved in the training process. The first experimental stage aimed to determine the level of achievement of the performing students and was carried out in two samples.

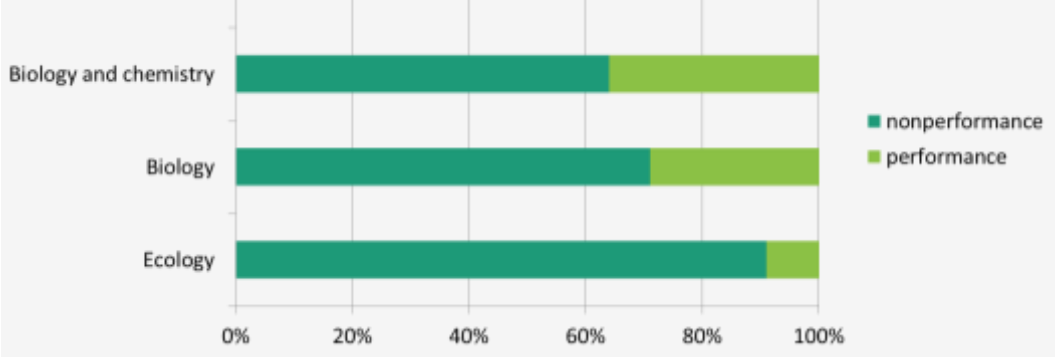


Figure 1. Student performance on analysis and description skills

Sample I provided for the examination of the students' performances regarding the competences of analysis and description of the morpho-structural features of the lower plants, respectively, of the vegetative and generative organs within the higher plants (fig. 1).

This has been carried out tests and questionnaires among students above specialties. Students performing in the category were included those who recorded scores between 9:10 in the evaluation process. The rest of the students were included in the non-performance category. The results show a worrying trend: the number of students who record performances in the monospecialty, decreases by about 7% -17% compared to students from the specialty biology-chemistry where the performance is about 36%. The highest percentage of non-performance was registered by the student-ecologists. This is explained by the following: students from traditional double specialties are better prepared from the perspective of interdisciplinarity in the field of real sciences [4]; students from the monospecialty have more limited visions regarding the understanding of some biological phenomena. For these reasons, the faculty members in the faculty are looking for high school graduates with a good background in the biological sciences determined to teach in education. Unfortunately, at present, there is a decrease in the number of high school graduates who want to become teachers in the pre-university system and for these reasons the quality of training in higher education is dropping.

Sample II was based on the evaluation of the students' performances regarding the competences: informative exploration, updating information, comparison and generalization, association, of synthesis in the following sequence: identifying the differences between different plant structures in the upper and lower plants [5,6]; their identification at the level of the photonic microscope, elaboration of models inspired by nature (fig. 2).

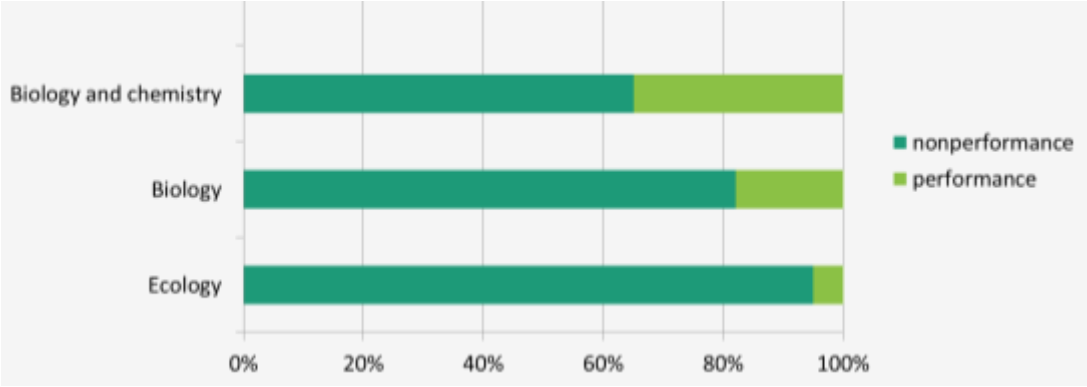


Figure 2. Student performance on informational exploitation skills, updating information, rationalization, comparison and generalization, of association and synthesis

In the examined situation, likewise, the students from the specialty biology-chemistry registered the best results. About 35% of the students tested were included

in the performing students category, compared to 18% in the biology specialty and 5% in the ecology specialty. It is interesting to note that the study programs in the analyzed specialties are basically the same. The same teachers are involved in the teaching-learning process but, unfortunately, the students from the mono-specialties registered weaker results. The explanations mentioned above remain valid in this case.

The second experimental stage included a single sample and was based on two performance pedagogical formulas:

- The first formula (20 minutes) included the following: writing a scientific essay based on the references given by the teacher, using previous skills (it has been observed if the student respects the reference elements and if he / she correctly senses the essence).
- The second formula involves working with information. The student analyzes the content of the essay from another point of view, elaborating in 25 minutes, the following: interpreting the content of the essay from the perspective of the inter / transdisciplinarity of the real sciences, arguing with concepts and notions from other fields such as: mathematics, chemistry, physics. It was examined how many arguments the student makes and their persuasive character (fig. 3).

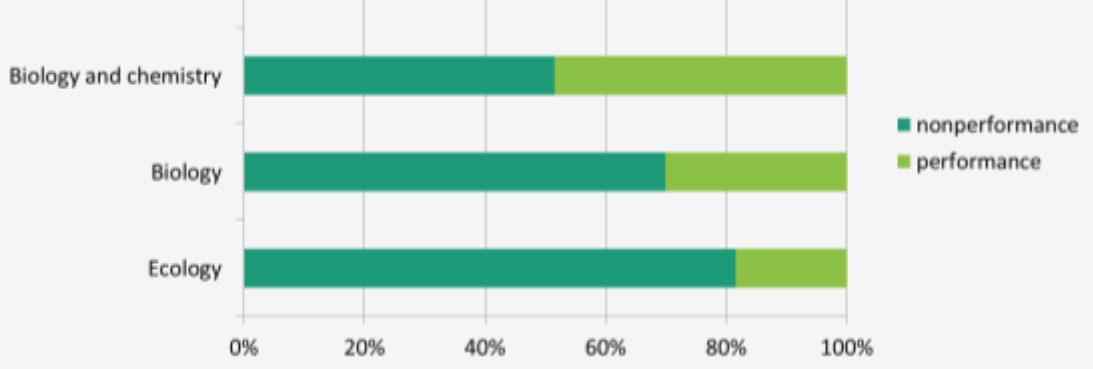


Figure 3. Student performance on the achievement of the experimental stage II

The results obtained in the second phase based on pedagogical formulas show a clear improvement of the students' performances in all the studied specialties. Thus, in the specialties of biology and chemistry, biology and ecology The performances are respectively registered 48.5%, 30, 1% and 18,5%. These results are due, first of all, to the pedagogical benchmarks established by the teacher, which guide students to the topics examined in depth at the course and laboratory hours. Undoubtedly, intelligent and imaginative students are the determining factor, what follows in the near future to make a significant contribution both in education as well as the socio-economic development of the country.

4. Conclusions and recommendations

- Biology students from the biology-chemistry specialty have a higher level of preparation compared to students from monospecialties. his, in our opinion, is

due to fundamental theoretical and practical training, students in terms of inter / transdisciplinary and wider vision on understanding biological phenomena

- The teaching-learning process should focus not only, exclusively, on the curriculum but also on the development of students' skills and abilities.. In this context, clearly, the assessment must also take into account not only the academic progress and performance, but also the growth and development of the students' innate skills.
- The teaching career is not attractive for young talented, dynamic and entrepreneurs who see perspectives in education in Moldova. Not reinforce existing educational system in terms of quality because young talented, and well known for objective reasons, refuse to work in education.

Bibliography

1. Chiriac E., Țiganaș A. Procesul de instruire la facultatea de biologie și chimie prin prisma asigurării calității. Acta et commentationes. Ştiinţe ale Educaţiei. Nr. 1 (8), Chişinău, UST, 2016. p. 21-33.
2. Chiriac E. Abordări didactice în procesul de studiere a științelor biologice prin prisma interdisciplinarității biologie-matematică. CAIM 2017. Book of Abstracts. The 25 Conference on Applied and Industrial Mathematics Iași, România, 2017. p. 116.
3. Chiriac E. Research work of Biology students in the context of inter - disciplinarity. III Miedzynarodowej Konferencji Naukowej – Uniwersalizm Pracy Ludzkiej Praca Jako Wyznacznik Egzystencji Czlowieka, Krakow, 2018. p. 32-36.
4. Chiriac E., Chiriac L. Aspecte de interdisciplinaritate în predarea biologiei. Conferința Republicană a Cadrelor Didactice, Chişinău, UST, 2018. p. 7 -10.
5. Chiriac E. Bionica – o nouă paradigmă în educația interdisciplinară. CAIM-2018, Proceedings of the 26 th Conference on Applied and Industrial Mathematics, UTM, Chișinău, 2018. p. 32 -38.
6. Chiriac E., Nedbaliuc B., Grigorcea S. Simetria florii – concept matematic fundamental în studierea naturii. CAIM-2019, Proceedings of the 27 th Conference on Applied and Industrial Mathematics, Valahia University, Targoviste, România, 2019. p. 56-61.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 114-119 E-ISSN 2587-3636

CZU: 37.016.046:91 DOI: 10.36120/2587-3636.v18i4.114-119

CONSIDERATION ABOUT QUALITY IN EDUCATION THROUGH GEOGRAPHY - THE PREMISE FOR A PERFORMANCE GEOGRAPHICAL EDUCATION

Elena SOCHIRCĂ, associated professor, PhD

The Chair of Human, Regional Geography and Tourism, Tiraspol State University

Vitalie MAMOT, lecturer

The Chair of Human, Regional Geography and Tourism, Tiraspol State University,

Abstract. The three pillars of the educational reforms of the last decades in Europe and in the world, on which is based the national systems of management and quality assurance, including the Republic of Moldova one, are quality, equity and efficiency. This alignment is necessary not only to ensure a real and functional integration, from an educational point of view, of the Republic of Moldova into the European Union, but also, for the initiatives in this field to be consonant, from the theoretical and methodological points of view, with what is happening now in the world. The research endeavor in education quality does not suggest a sistematically coordinated agenda. That is why integrated information from cognitive science into the conceptualization of giftedness and educational program design targeted at gifted children are more tnan welcomed.

Key-words: education, geographic education, student results, assessment, curriculum.

CONSIDERAȚII PRIVIND CALITATEA ÎN EDUCAȚIA PRIN GEOGRAFIE – PREMISĂ PENTRU UN ÎNVĂȚĂMÂNT GEOGRAFIC DE PERFORMANȚĂ

Rezumat. Cei trei piloni ai reformelor educaționale din ultimele decenii în Europa și în lume, pe care se bazează sistemele naționale de management și asigurare a calității, inclusiv în Republica Moldova, sunt calitatea, echitatea și eficiența. Această aliniere este necesară nu numai pentru a asigura o integrare reală și funcțională, din punct de vedere educațional, a Republicii Moldova în Uniunea Europeană, ci și pentru ca inițiativele din acest domeniu să fie în consonanță, din punct de vedere teoretic și metodologic, cu ceea ce se întâmplă acum în lume. Efortul de cercetare în calitatea educației nu sugerează o agendă coordonată sistematic. De aceea, informațiile integrate din știința cognitivă în conceptualizarea supradotării și proiectarea programelor educaționale vizate copiilor supradotați sunt mai mult ca bine venite.

Cuvinte cheie: educație, educație geografică, rezultate ale studenților, evaluare, curriculum.

Introduction

The three pillars of the educational reforms of the last decades in Europe and in the world, on which is based the national systems of management and quality assurance, including the Republic of Moldova one, are quality, equity and efficiency. This alignment is necessary not only to ensure a real and functional integration, from an educational point of view, of the Republic of Moldova into the European Union, but also, for the initiatives in this field to be consonant, from the theoretical and methodological points of view, with what is happening now in the world. The need for unitary understanding of the education quality appeared from the fact that ,, it does not exist yet, a unitary concept of quality, this being judged according to the values promoted in the society and at the level of the school organization, educational politics and strategies existing at national, regional and local

levels, existing situation, defined by contextual and situational factors, the evolution of the concept of „quality”” [1, p.2].

Methods and materials

In order to carry out this study we opted for qualitative methods of investigation, namely the case study. The case study was focused on the school discipline Geography and the elements underlying the quality assurance of education through geography. Other methods used were the analysis of existing curricular documents, observation of curricular and extracurricular activities.

Results and discusion

Some defining elements of quality in education are highlighted (fig. 1). The most important of these elements is considered the completeness. The quality of object, process or phenomenon does not refer to characteristics treated separately, but integrated. So, other two important elements of the quality in education results - hierarchy and the capacity for change. The quality change could be spontaneous or directed. In the last case we can refer to the quality management capacity. Another important elements is the axiological character - different individual perception, adaptability to some or other objectives, conditions, needs of a particular person, social group or society in general. The degree of satisfaction of needs is determined by the degree of the quality expression. It creates the possibility to measure quality, both quantitatively and qualitatively.

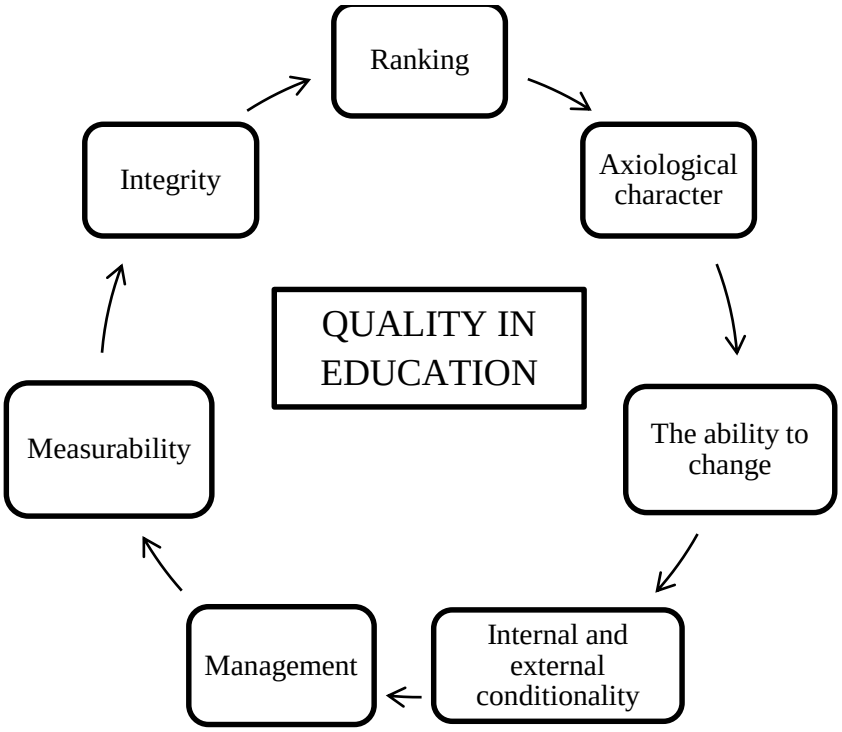


Figure 1. The defining elements of quality in education

In the opinion of Constantin Șerban Iosifescu, a product or service is defined as "quality" only if, on the one hand, it meets certain standards according to which it has been defined (it fulfills the functions for which it was created) and, on the other hand, if it meets or exceeds the expectations of the beneficiaries, if he/she likes it. We can have quality in

education only if what the school offers meets the stated or implicit needs of individuals or of the community and, if, on the other hand, the beneficiaries of the educational services are satisfied with the educational services offered. Establishing and adhering to quality principles is essential for any organization that wants to offer quality products or services. The quality of the education offered by the school can be increased by several quality indicators, among which the quality of the students activity, by:

- enriching the students' learning experiences through new learning strategies and methods, through interaction with other students, etc .;
- increasing student performance - school results, exam results, education for using of new information and communication technologies, education for society, ecological education, demographic education, etc.

The literature aimed to educate students capable of higher performance identifies certain fundamental premises of the curricular theories in this field, the most important ones aiming to training differentiation and the curricular adaptation to their development needs in confluent approaches that allow accelerated and advanced learning and extended experiences [2, p .116].

The main directions of curricular adaptation refer to curricular acceleration and enrichment. The curricular acceleration implies educational strategies by which the advanced contents and advanced development of skills and abilities is offered to student, before the age at which they are expected to be acquired. Two forms of acceleration are essential in the practice of gifted education: content-based and school-based acceleration. Curriculum enrichment typically uses three major types of activities: students are exposed to a variety of disciplines, study topics, people, events, etc .; offering learning experiences that promote intellectual and emotional development; the acquisition of advanced content at a certain school discipline.

There are several national and international normative documents of educational policies at the geographic education base in the Republic of Moldova, among which:

- Education Code of the Republic of Moldova, no. 152 of 17.07.2014, as subsequently amended;
- The reference frame of the National Curriculum, approved by Ministry of Education order no. 432 of May 29, 2017;
- International Charter of Geographical Education, 1992 and 2016;
- Key competences for lifelong learning, adopted by the European Parliament and Council in Brussels on May 22, 2018;
- Framework plan for primary, gymnasium and lyceum education, approved annually by the Ministry of Education, Culture and Research order;
- Geography Curriculum, 2010 and 2019 editions.

Geography is a school discipline that has a specificity highlighted by two significant characteristics:

a) the favorite object of study is the interference (interaction) between nature and society, geography being in this way a science of both nature and society;
b) the main methodological dimension represented by the cartographic method.

Nowadays we are in a stage of conceptual searches and of permanent renewal of the geographical content in the general education. This approach is demonstrated against the background of great changes in teaching-learning-evaluating geography in general education. From a school discipline oriented to the formation of a scientific-geographical representation of the world, geography has become a discipline, the content of which has a praxiological and unique character of geographical knowledge and play a great role in the formation and development of world culture, its importance in the life and activity of the contemporary man.

The Geography Curriculum offers premises for accelerating and enriching the curriculum, based on the following conceptual benchmarks: conforming the Geography curricular concept to the latest achievements and trends in the field of education sciences and geographic sciences at national and international level, firstly by focusing on skills training and designing the system of study purposes (skills, learning products etc.), by taking over good practices and adjusting them to the geographical reality of our country; reconceptualizing of school geography and capitalizing on new opportunities for studying geography, at a higher quality and efficiency level, by optimizing the system of competences, scientific contents, teaching-learning-evaluation activities, in light of current rigor; making interdisciplinary connections, by correlating the Geography curriculum with other school disciplines curriculum in general education, both by formulating competences, and by adjusting the recommended contents, activities and learning products, etc.

The geography curriculum included the units of competence, activities and learning products which are directly connected each other. At the same time, there is a correlation with the content units that the teacher must perform individually for each lesson. The learning activities are formulated in a diversified way, focusing on the interaction of the student with constructed supports, individual or group investigation, the development of critical thinking and active involvement in the process of its formation.

Within the geography lessons, many learning situations of a practical nature can be organized, in which the practical activities aim to acquire procedural knowledge (to analyze a landscape; to interpret a map; to construct a diagram, to measure an environmental indicator with an instrument or device, etc). When we designe a situation where students acquire this kind of knowledge, the most commonly used verbs are: to estimate, to calculate, to measure, to elaborate, to elaborate, to orient, to use, to apply, to comment, to solve, to combine, to model, to experiment, to represent etc.

With reference to geographic education, quality expresses the integrated character of education through geography, determining its capacity (or inability) to meet existing and potential needs of personality and society. The quality of the geographic education has a

hierarchical character, that is, it implies an interdependence between several component elements, and the contribution of each component in the achievement of the integrated character is different (fig.2).

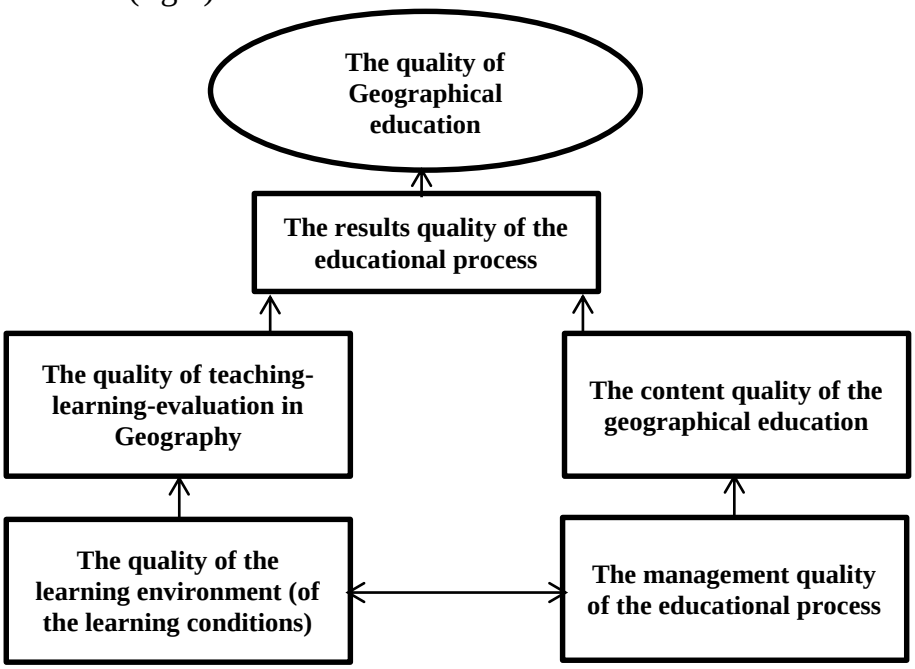


Figure 2. The quality of geographical education

The results quality of the educational process in geography is of particular importance. Secondly, there is the quality of the teaching-learning-evaluation process in geography and the quality of the content of geographic education. The third position in the hierarchy is occupied by the quality of the learning environment and the management quality of the educational process and of the education system generally.

Research shows that the efficiency of geography learning depends on the teachers qualification, the curriculum content and textbooks, the use of learning means and methods, the educational management, achieved at all levels. In order to analyze the situation of the geographical education, it is proposed to apply all the factors mentioned above and not just to limit at the assimilation level assessment of the content elements in the curriculum and textbooks. Not all of the mentioned factors can be measured from the perspective of quality indicators. So, through the concept / expression "quality of school geographic education" is reflected the contemporary and traditional pedagogy approaches through geography (teaching methodology in geography). In this case, the main quality indicator serves the ***students' school results*** (students' performance / achievement). They could be subjected to dynamic analysis and can be grouped into three categories: summative, formative and diagnostic [3, pag.79].

The evaluation of the educational results / success of the students are closely related to the research of the education role by geography in the formation of the student's personality. The evaluation of the training results involves a multilateral character: the students' knowledge about the world, using / application of the scientific knowledge methods by them; understanding the importance of scientific knowledge for human life

and activity. In the Geography curriculum, at the end of each class there is the *Purpose* section, which includes the specific geographical skills and attitudes that the student will acquire at the end of each study year. The factual results of learning are presented in the form of knowledge, skills, abilities, attitudes and values.

Conclusions

Nowadays, the curriculum of geography presents the advantage of providing study content in an accelerated manner, but also offering enriched learning experiences aimed at developing higher-level cognitive skills which could ensure the transfer of acquired knowledge and quality assurance in geography education**.** At the same time, the importance of motivation, focused on the idea of self-directed learning, in which intrinsic motivation predominates, must be emphasized. One of the basic features of the education quality evaluation through geography is the need for permanent monitoring on the success of students in geography. In order to ensure a qualitative educational process in geography and to achieve a curricular acceleration, oriented to students, it is recommended to make field applications. These field applications will be more effective if they have an integrated character through the inclusion of other school subjects: biology, physics, chemistry, history and so on. Curricular acceleration in geography could be achieved, also, by inviting well-known researchers at lessons, scientists who would hold guest lectures.

References

1. Iosifescu Ş. (coord.). Managementul şi cultura calităţii la nivelul unităţii şcolare. Bucureşti, 2005. 145 p.
2. Fritea I. Abordări curriculare în educația elevilor capabili de performanțe superioare. În: Chiș, V., Albulescu, I. (coord.). Înnoirea educației. Studii de pedagogie și didactică aplicată. Casa Cărții de Știință. Cluj-Napoca, 2011. p. 116-121.
3. Хлебосолова О.А. Качество школьного географического образования и способы его оценки. Рязань, 2007. 152 с.
4. Санина С.П. Проблемы обучения географии: обзор зарубежных исследований. Электронный журнал «Современная зарубежная психология», 2019. Том 8. № 1. с. 17—27.
5. Chiș V., Albulescu I. (coord.). Cercetări și aplicații în științele educației. Casa Cărții de Știință. Cluj-Napoca, 2010. 390 p.
6. Curriculum Național. Disciplina Geografie. Chișinău, 2019.
7. Curriculum la Geografie axat pe formarea de competențe. Chișinău, 2010.

Acta et Commentationes, Sciences of Education, nr. 4 (18), 2019 ISSN 1857-0623
p. 120-124 E-ISSN 2587-3636

CZU: 372.8004 DOI: 10.36120/2587-3636.v18i4.120-124

PROBLEMS OF JUNIOR CONTESTANTS TRAINING FOR PARTICIPATION IN NATIONAL AND INTERNATIONAL PROGRAMMING COMPETITIONS

Sergiu CORLAT, Technical University of Moldova
Lilia IVANOV, National Agency for Curriculum and Evaluation

Abstract. In the article it is made an analysis of current situation in preparation of young (junior) students for participation in Informatics national and international competitions. There are identified problems of qualitative preparation and models of solutions for these problems. In the same time there are classified decisive factors in future extension of juniors preparation.

Key words: Programming, Juniors competitions, specific competences, training cells, didactic strategies.

PROBLEME IDENTIFICATE ÎN CADRUL PREGĂTIRII JUNIORILOR PENTRU CONCURSURI NAȚIONALE ȘI INTERNAȚIONALE DE PROGRAMARE

Abstract. În articol se face o analiză a situației actuale în pregătirea elevilor de vârstă școlară mică (juniori) pentru participarea la competiții naționale și internaționale de informatică. Sunt identificate probleme ce apar în pregătirea calitativă a juniorilor și modele de soluții pentru aceste probleme. În același timp, sunt clasificați factorii decisivi pentru viitoare aextinderea în pregătirea juniorilor.

Cuvinte cheie: programare, competiții pentru juniori, competențe specifice, celule de antrenament, strategii didactice.

Introduction

In the last decade there has been a significant increase in the number of programming contests at all levels: from institutional to international ones. There are several reasons for this growth: expanding the role of digital technologies in human activity; modernization of national curricula; the introduction of new school disciplines oriented towards the development of digital skills, including in primary school; the emergence of a large number of specialized educational resources, adapted to target groups of different ages; development of web platforms for automatic evaluation.

The increase of the number of programming competitions leads to the involvement of an increasing number of participants, who have different levels of training in the Informatics field, but also a significant age gap. The increase of the number of participants leads to an increase in the complexity level of the proposed problems. At the level of international competitions this means solving an identical set of problems by the participants between the ages of 15 (or even less) and 19 years, where, obviously, the participants of younger age are disadvantaged compared to their older colleagues. Increasing age differences imply reverse processes - reducing complexity or dividing competitions into sections for seniors (16 - 19 years) and juniors (pupils with the age up to 15.5 years).

Statistical data

If the training methodologies of the seniors are well defined and tested at national level for about 30 years, the preparation of the juniors for competitions of different levels is at the moment spontaneous and little organized.

Thus, a statistics of the juniors' participation in the official international competitions in the period of 2015 denotes an acute insufficiency of the pupils who possess a sufficient training level for these competitions (the Balkan Informatics Olympiad for juniors - diagram 1; the European Informatics Olympiad for juniors - diagram 2)

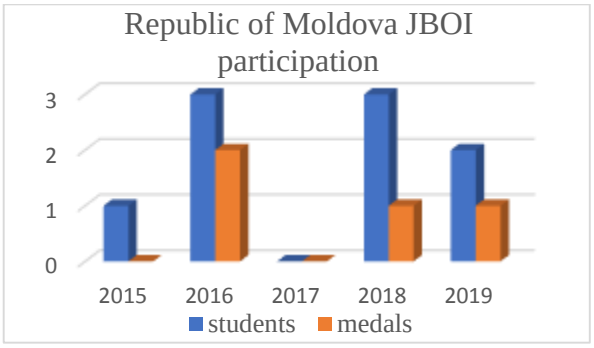


Diagram 1: Statistics of pupils' participation from the Republic of Moldova in the Balkan Informatics Olympiad for Junior, 2015 - 2019

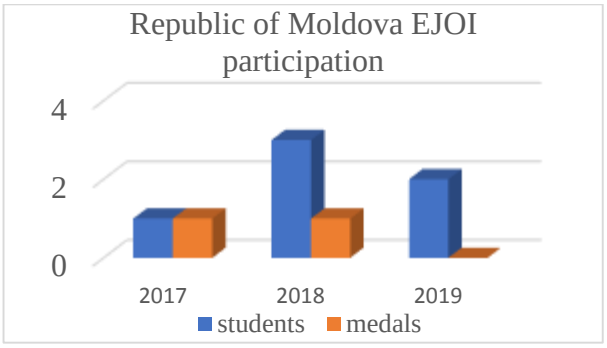


Diagram 2: Statistics of pupils' participation from the Republic of Moldova in the European Informatics Olympics for Junior 2017 - 2019

The comparative analysis of the number of participants in the Republican Informatics Olympiad shows a significant difference between the number of senior and junior participants (diagram 3). An additional problem is the participants' age limit in the programming competitions for juniors: most juniors with high scores reach the age limit in the period between the national competition and the period of international competitions, thus becoming ineligible to participate in the latter.

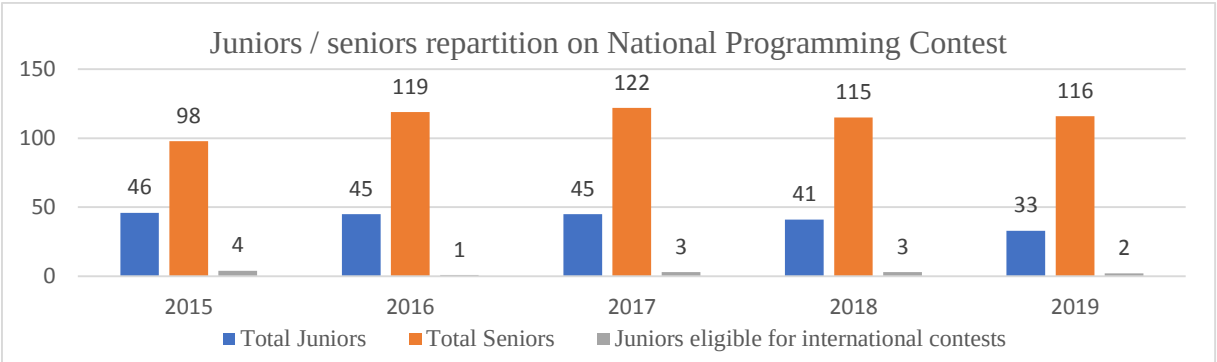


Diagram 3. Number of senior participants (grades 10-12) and juniors (up to grade 9), including eligible ones for the participation in international programming competitions for juniors

Why so few juniors?

The main reasons:

• Studying the programming elements starting with the 9th grade (according to the Informatics curriculum in force during the analyzed period);

• the "expert" of the performance training in other real disciplines (Physics, Chemistry, Biology) who selects the pupils starting with grades 5 - 6.
• the level of teachers' training and their ability to perform the training performance**.**

What to do?

a. Modernization of the curriculum - offering the possibility to include modules for the development of algorithmic thinking and programming skills in a high level language from an early age. Created preconditions: the inclusion of the Digital Education discipline in the primary education, with the visual programming module "Gândim Digital?Thinking digitally" (2018), the modularization of the curriculum in the Informatics discipline in the gymnasium grades with the option of choosing the programming modules starting with the 7th grade (2019). The programming modules form a continuous series of training, starting from the programming of the elementary algorithms and to the techniques and complex data structures, in the XII th form: My first programs (grade VII at choice), *Algorithms and executors, implementation of algorithms (grade VIII at choice), Implementation of algorithms, structured data processing (grade IX at choice), Programming languages, data, elementary data structures (grade X), Principles of computer systems design (grade XI), Subprograms, programming techniques, modeling and numerical calculation (grade XII).* The new curriculum also involves the integration of programming elements into educational activities in other real disciplines, in particular - Mathematics and Physics, in order to demonstrate the necessity, applicability and importance of programming in the digital society. The sensitive component of this curriculum is the didactic framework, which can "avoid" the programming modules in grades VII - IX, these being optional, in favor of the application modules, close to the field of digital technologies: editing different types of digital content, using office applications, and so on.

b. For **the teachers'active involvement,** a new approach of their training is needed, both of the teachers in the educational field, and of the students from the profile faculties. For the first category, it is necessary to include in the training program the type modules: fundamentals of programming and competitive programming. The training of the students from the profiling faculties, both at the first cycle, and second cycle-Master, can benefit from the extension of the range of compulsory and optional courses (modules), which contain both motivational pedagogical elements and programming elements of different complexity levels.

c. Creation of early training centers in the field of Informatics, which may include programming courses, oriented to the age category of 11-14 years. Identification of mentors for the programming modules, activities, psycho-pedagogical approaches for the initial students' motivation and training of self-motivation capacities, the competitive character and the critical thinking. Achievements: Based on the resources

of the Technical University of Moldova and the ICT Tekwill performance center, the *"Programming fundamentals" and "Competitive programming*" courses have been launched, to which students from Chisinau educational institutions have access. In the development process: launch of the TCPAD Center (Training Center in Programming, Algorithmics, Data) for the creation of an ecosystem of mentors - students - pupils - ICT companies, in which to develop young IT talents. The activity of these performance centers is to be guided by the international performance curriculum in Informatics (for seniors and juniors separately [4]).

d. Elaboration of educational resources that should allow both teacher (mentor) -assisted learning and self-instruction. Certainly, the resources will be adapted to the age category, to which they are addressed. In the development process: TwentyTu Moldova project, which intends to develop national resources for the courses: *Procedural programming in C / C ++; Algorithms & Data Structures; Programming in Python; Artificial Intelligence etc.* The resources developed will correlate with the requirements of the national curriculum in the discipline, but may include much wider areas of knowledge, for those pupils who want an active participation not only in national competitions but also in international ones.

e. Promotional activities and events. The increase of the number of participants in the programming contests for juniors will only be possible if all the activities launched or designed previously will involve a sufficiently large number of children aged 11-14, who, in fact, form the target group. For this, it is necessary to organize periodic discussions, lessons, practical sessions and microcompetitions of programming for beginners, managed locally by Informatics teachers, pupils from the upper classes or students - graduates of the educational institution. These will be followed by unofficial regional competitions dedicated specifically to this age category. Partnerships between school institutions and IT companies can solve logistical and financial problems related to the organization of competitions of this kind. The emergence of a "critical mass" of junior prepared for the participation in regional and national programming competitions will certainly be an argument for the Olympic Informatics Council to launch the necessary steps to modify the model of selection of participants for national competitions and selection of Olympic groups of seniors and especially - of juniors, so that the number of junior participants in national Informatics competitions is at least equal to that of the participants from the high school classes.

f. Identification of sustainable financing sources. The realization of the proposed projects involves significant costs, which cannot be fully covered only by educational institutions. The need to identify external financing is an open problem that, at the moment, is trying to be solved with the help of IT companies (example of the Tekwill Center, the TwentyTu project, the Orange foundation, etc.) but their opening for the collaboration with educational institutions is insufficient to solve this problem.

Conclusions

The juniors' training for performance in Informatics is a process of major importance and is the foundation of the pupils' successes in the national, regional and international programming competitions. Although the first steps to systematize the training process have already been taken, there are several problems that can be solved only in time, through the joint effort of educational institutions, performance educational centers, universities, ICT companies as well as through the effort of teachers and parents. Among these:

- Ensuring equal and free access to universal educational resources for the pupils of all ages and from all localities. The universal resources involve not only the support in training and self-training, but also possibilities to practice and simulate competitions on web platforms, with automatic evaluation of the results.
- Creating performance cells within educational institutions (local target group) or universities (regional, national target group), training mentors to administer the cells, providing the necessary logistical, didactic, methodological support.
- Active involvement of ICT companies in organizing events and activities to promote IT performance, programming competitions for juniors, organizing thematic trainings for training programming skills and developing algorithmic thinking.
- Elaboration of national documents for performance training - Curriculum - separately for juniors and seniors.
- Modification of the model for admitting junior students to local, regional and national competitions in order to reduce the numerical gap and increase the number of juniors eligible for training within the camps for the Olympic groups and participation in international programming competitions.

Bibliography

1. Gremalschi A., Corlat S., Prisacaru A. Olympiads in Informatics in Republic of Moldova. In: IOI Olympiads in Informatics, vol. 10, 2016. p 255 – 262.
2. Кирюхин В.М.. Методика проведения и подготовки к участию в олимпиадах по Информатике. Бином, 2011.
3. Vorderman C. Computer Coding for Kids: A Unique Step-by-Step Visual Guide, from Binary Code to Building Games, 2014.

Web links:

4. https://tekwill.md
5. https://twentytu.md
6. https://atic.md
7. https://ioinformatics.org/page/syllabus/12