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).

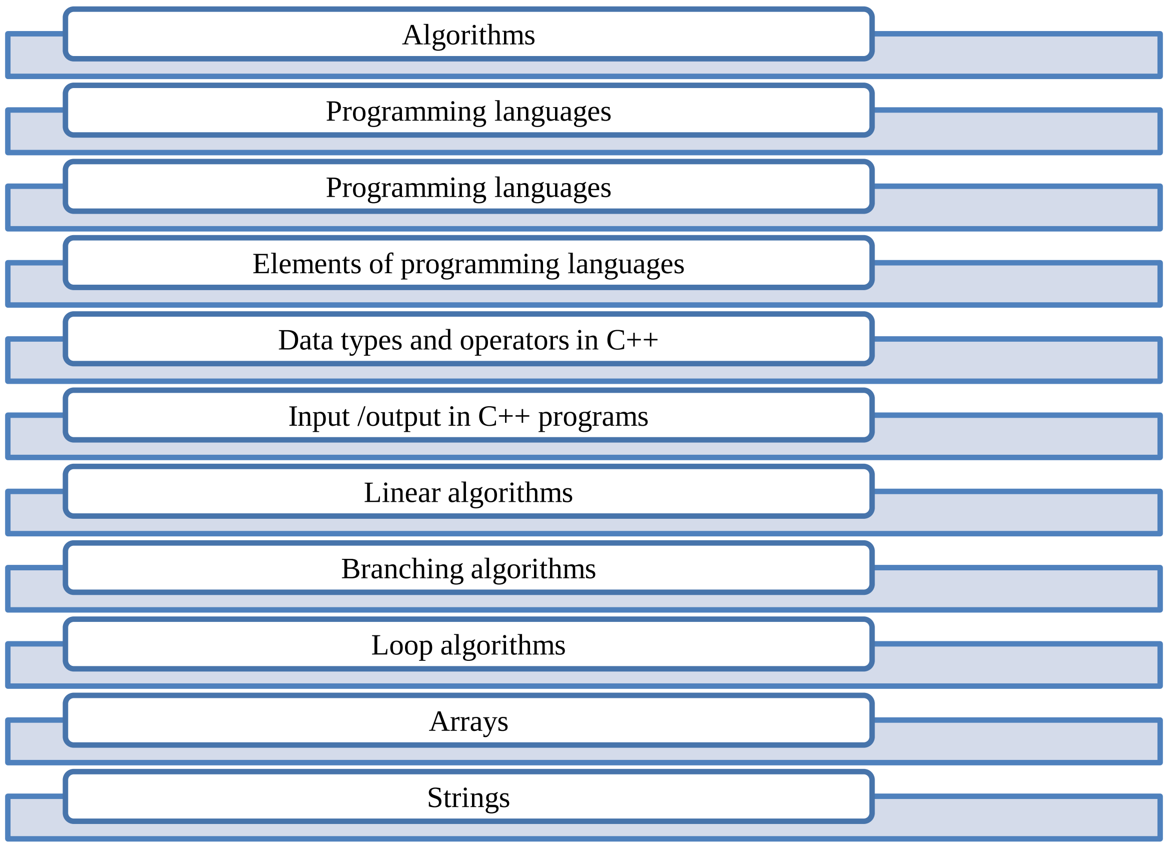

Figure 1. The important themes in first year of programming learning

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

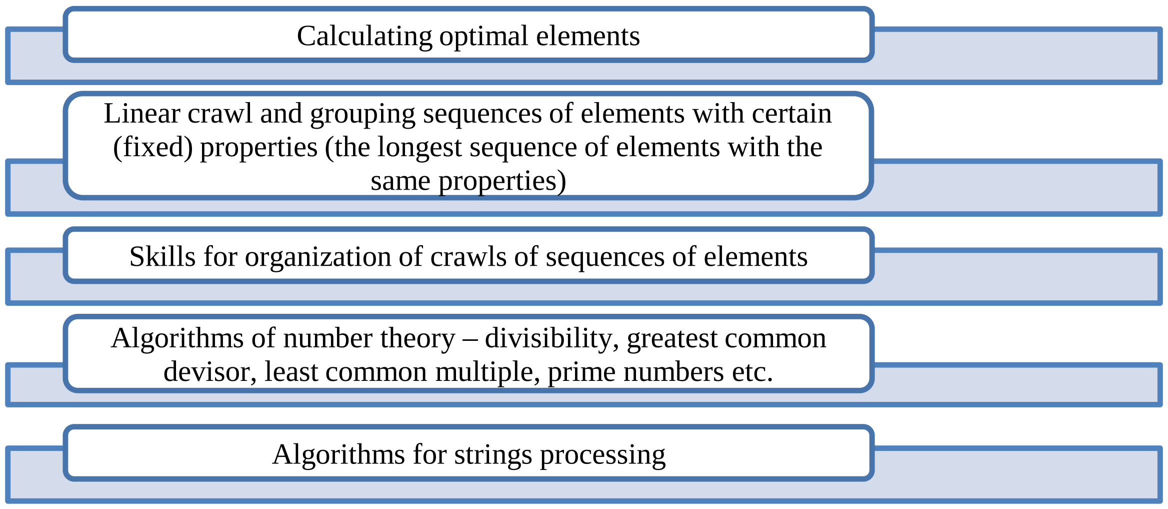
Calculating optimal elements
Linear crawl and grouping sequences of elements with certain (fixed) properties (the longest sequence of elements with the same properties)
Skills for organization of crawls of sequences of elements
Algorithms of number theory – divisibility, greatest common devisor, least common multiple, prime numbers etc.
Algorithms for strings processing

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)

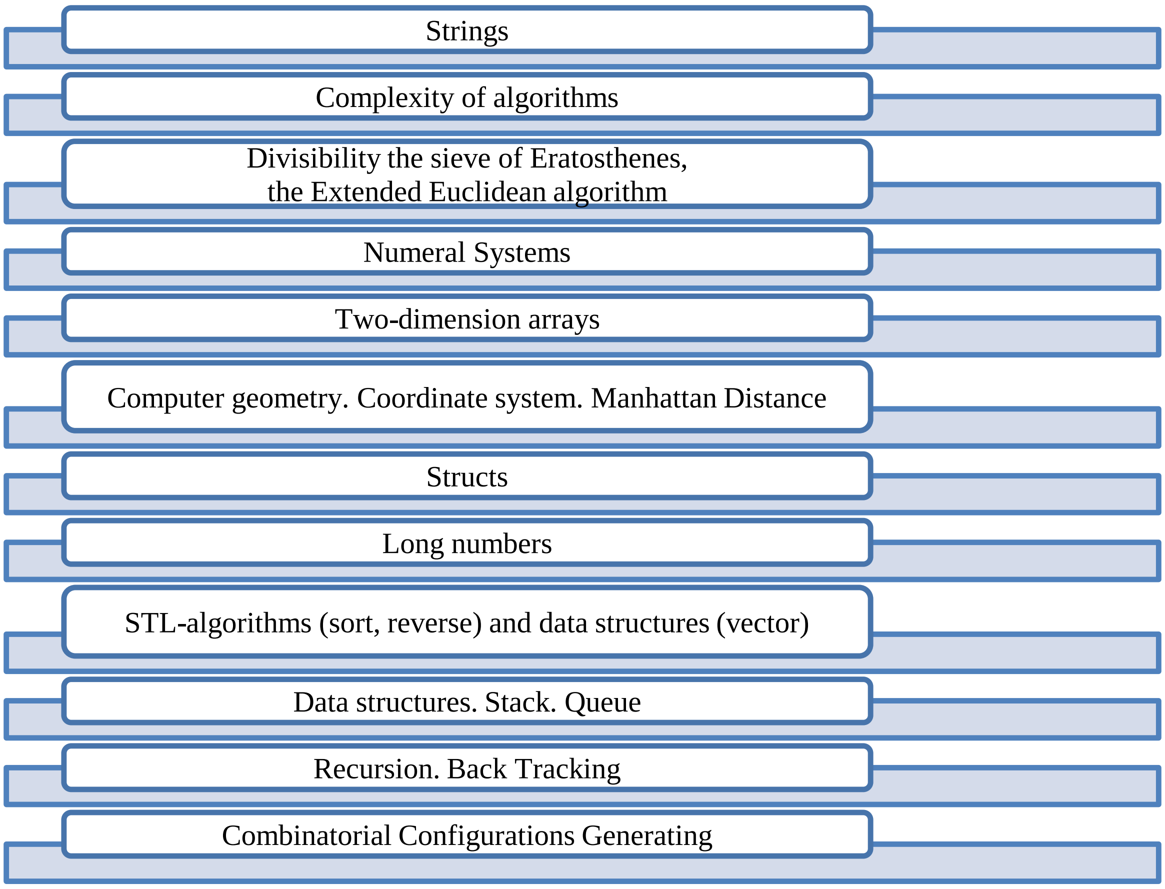
Strings
Complexity of algorithms
Divisibility the sieve of Eratosthenes, the Extended Euclidean algorithm
Numeral Systems
Two-dimension arrays
Computer geometry. Coordinate system. Manhattan Distance
Structs
Long numbers
STL-algorithms (sort, reverse) and data structures (vector)
Data structures. Stack. Queue
Recursion. Back Tracking
Combinatorial Configurations Generating

Figure 3. The important themes in the second year

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

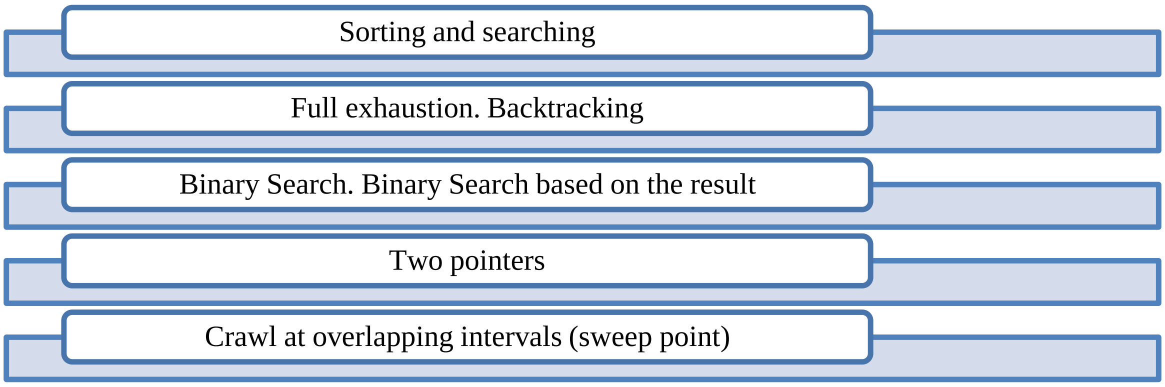
Sorting and searching
Full exhaustion. Backtracking
Binary Search. Binary Search based on the result
Two pointers
Crawl at overlapping intervals (sweep point)

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.