Problem Solving Olympics: an inclusive education model for learning Informatics

Roberto Borchia, Antonella Carbonaro, Giorgio Casadei, Luca Forlizzi, Michael Lodi, Simone Martini

Problem Solving Olympics: an inclusive education model for learning Informatics

Roberto Borchia[1], Antonella Carbonaro[2], Giorgio Casadei[2], Luca Forlizzi[3], Michael Lodi[4], and Simone Martini[4]

[1] Istituto Tecnico Industriale "Q. Sella", Biella, Italy
[2] Università di Bologna, Dip. di Informatica–Scienza e Ingegneria, Italy
[3] Università dell'Aquila, Dip. di Ingegneria e Scienze dell'Informazione e Matematica
[4] Università di Bologna, Dip. di Informatica–Scienza e Ingegneria, Italy; and INRIA Sophia-Antipolis, France

Abstract. The paper presents the *Olimpiadi di Problem Solving*, a mild and inclusive competition aimed to promote computational thinking and general problem-solving in Italian schools. We describe motivation, teaching strategies behind the initiative, as well as its structure, organization and give some sample of the problems proposed to students. We also present some data that show the broad participation of Italian schools to the initiative, and a preliminary analysis of the results obtained by the students in the last 5 editions, which suggests that the competition fosters deep learning of computational thinking knowledge and skills.

Keywords: Inclusive Competitions · Computational Thinking · Problem-solving.

1 Introduction

The *Olimpiadi di Problem Solving - Informatica e pensiero computazionale nella scuola dell'obbligo* ("Problem Solving Olympiad - Informatics and computational thinking in compulsory education") (henceforth: OPS) is an initiative of the Italian Ministry of Education, University and Research (henceforth: MIUR), to foster both problem-solving and team-working skills in a single action.[5]

Usually in the education world the expression "Olympics of X" means a competition aimed to single out and promote exceptional skills at the individual level in the age of 16-18. However:

- OPS are mainly training activities: the core of OPS are the training contests, followed by two rounds of competitive contests (regional and national);
- OPS are mainly team-work activities;
- OPS are aimed at all the compulsory education age range; activities are organized at three *levels*: the last two years of primary school (4-5 grade), lower secondary school (6-8 grade), the first two years of upper secondary school (9-10 grade);

[5] The project guide is entrusted to the Directorate-General for Education Guidelines (*D.g. per gli ordinamenti scolastici e per l'autonomia scolastica*) of MIUR.

- OPS are aimed to reach all students, and not only exceptional ones: each school is encouraged to participate with as many teams as possible (for each level);
- OPS are gender neutral: each team is required to have people of both genders.

It was noted by Nobel laureate Carl Wieman [11] that competitions outside of school fail to improve STEM education; since 2008 MIUR has promoted OPS as training/competition *within* the school system to pave the way for a more widespread introduction of Informatics in curricula and to foster the learning of effective competences for general problem-solving.

The remainder of this paper is organised as follows. Section 2 discusses the variety and diffusion of contests focusing on Informatics organized worldwide in the last decades. Section 3 describes main features and goal of OPS, and the structure of the competitions. Furthermore, it proposes some examples of standard problems and presents how the training and the contests, at the different levels, are organized. Section 4 presents some data about participation in the initiative and a preliminary analysis of the results obtained by the students in the last editions, to evaluate the training effect on students on their problem-solving skills. Section 5 proposes the question of digital gender gap and evaluates the involvement of girls and boys in OPS. Finally, some considerations close the paper.

2 Promoting Informatics through competitions

Several contests focusing on Informatics have been organized worldwide in the last decades. They vary in design, difficulty levels, and type of audience they cater to.

Such events mainly result from two attitudes of mind: one focused in selecting particularly talented students and another one aimed at spreading the basic concepts of the discipline to a vast audience of students. This section only shares a few examples to show the variety and diffusion of such events. See [1] and [8] for recent reviews about contests, classification criteria and a selection of existing online programming contests.

ACM International Collegiate Programming Contest (ICPC)[6] is the most sought after programming competition in the world. Every year, the ICPC contests include contestants from over 2000 universities and colleges from across the world. The contestants receive problems that they have to solve, providing a program written in a widespread programming language such as C, C++ or Java.

IEEE also proposes a contest, namely the IEEEXtreme Programming Competition[7], which lasts 24 hours and is dedicated to teams of students. All the teams receive a set of programming problems and as for ACM-ICPC, they have to solve the greatest number of problems.

[6] https://icpc.baylor.edu/
[7] https://www.ieee.org/membership/students/competitions/xtreme/index.html

Among the events targeted to secondary school students, the International Olympiad in Informatics (IOI)[8] is the most popular worldwide. Like in ICPC, contestants are challenged to provide, in few hours, programs written in mainstream programming languages that solve problems of algorithmic nature. Unlike ICPC and IEEEXtreme Programming Competition, IOI is an individual contest. A lot of countries do have online programming contests to make the selection for their national team to be sent to the IOI. Just to mention some of them: USA Computing Olympiad (USACO), Italian Olympiad in Informatics, France-IOI, Indian Computing Olympiad (ICO), French-Australian Regional Informatics Olympiad (FARIO). All those contests are following the same philosophy as the IOI, encouraging young talent from high schools to programming challenges.

The Bebras contest [4] is a competition organized on an annual basis in several countries since 2004. In the past few years, the number of the Bebras participants has been notably growing. The contest has been spreading over 40 countries involving various activities, e.g. several rounds of the contest, discussion on Informatics topics, task solving seminars, teacher workshops, and task developing events. The Bebras challenge has been developed for all primary and secondary school pupils. The second full week of November is the official time for the Bebras challenge (first round) in participating countries. Several countries have established a second round of the Bebras challenge, usually at the end of January or beginning of February, and it is dedicated for the best participants of the first round challenge.

It is a well-known consensus in the community around Informatics contests that the difficulty of these contests progressively increases, as observed, for example, by Verhoeff et al. in [10]. Kelevedjiev and Dzhenkova [6] mention that in many countries the age at which children start with programming is already as low as ages 11 to 12. Thanks to this early start, the set of topics used in contest tasks is growing. Kiryukhin in [7] describes the development of the Russian Olympiad in Informatics: since 1995 the focus in programming contests started to shift from "implement a correct algorithm" towards the much harder "implement a correct algorithm that is as efficient as possible". As a result, and even more so than previously, only the best students are involved in these kind of competitions. All the other students lose the opportunity to develop, in an engaging way, fundamental problem-solving knowledge and skills required for living and working in our society, having applicability in the entire sphere of economic, social, political and cultural life. The dissatisfaction with this state of affairs motivated MIUR, back in 2008, to start OPS with the goal to spread problem-solving and computational thinking across all the compulsory education age range, which in Italy goes up to grade 10. Moreover, OPS are aimed to reach all students, and not only exceptional ones.

[8] http://ioinformatics.org/index.shtml

3 OPS: an inclusive education model for learning Informatics/computational thinking

3.1 Main features and Goals

The main goal of OPS is to disseminate problem-solving skills and computational thinking [12] in Italian education.

Since no explicit provision of problem-solving teaching is present in the Italian guidelines for primary and secondary education (while *computational thinking* has been recently introduced, for the moment only as a transversal competence), OPS promote activities in this area with a trans-disciplinary approach, trying to captivate teachers of various disciplines. Being activities directly offered to classes as contests, they are appealing and have an immediate impact on the participating youngsters [3]. However, no real successful participation is possible without support and training from local teachers. This implies that teachers should develop and promote curiosity, interest and consensus in their own classes. Teachers themselves are therefore stimulated to engage in the game even if skills and competences needed are foreign, at least explicitly, even for them.

The collaboration is recognized as one of the most effective means of spreading problem-solving skills and also allows for significant motivational and relational benefits. For this reason, the OPS foster collaboration by offering mainly team work activities. In designing the teaching strategy for OPS we have taken into account the dual nature of team competition (competitive and collaborative); actually this strategy offers many possibilities when facing a heterogeneous group of student [9].

Problem-solving as interpreted in OPS is intended as one of the most effective paths to the acquisition of computational thinking attitudes. For this purpose, one component of OPS is the ability to understand simple formal structures (such as terms, lists and graphs), and to follow, simple (and less simple) algorithms expressed in semi-formal languages, thus forming the background on which an explicit teaching of programming and Informatics may be based. This component is further strengthened (in grades 9 and 10) by administering some problems that require the execution of (easy) calculations to be repeated a number of times so large to make the solution infeasible unless one writes and executes a program.

Another important feature of the OPS, which differentiates it from most of the similar initiatives, is the fact that they favor incremental learning by offering explicit training sessions for both students and teachers during every entire school year. After each training session, the scientific committee of the OPS releases the solutions and the comments to the solutions for the various exercises proposed. Both the involved students and the teachers judge in a very positive way the usefulness of the solutions for the incremental acquisition of skills.

We may therefore summarize the main goals of OPS:

– offer explicit problem-solving training to students at the compulsory education age range (including team-working skills);

- involve the teachers in such a way that they also acquire and consolidate the needed skills (to use in their everyday teaching);
- use semi-formal setting and pseudo language to embed problem solving in the context of computational thinking with the perspective of (eventually) acquiring programming skills;
- offer training sessions during the entire school year to both students and teachers.

3.2 Joining the OPS

Participation is on a voluntary basis. Italian teachers are expected to participate in "additional activities" as well as to teach their discipline. MIUR promotes various initiatives that qualify as additional activities: OPS is one of them. In September/October, MIUR issues the annual bylaw for OPS, with the dates of the contests. Since participation in the project is the responsibility of the scholar institution, teachers have to negotiate adhesion with the principal.

Involved teachers (often in mathematics, but also in other disciplines) form teams within their classes; again participation of the students is voluntary, but very often discussion about OPS problems, assignment of exercises and study of problem-solving techniques are extended to the whole class and are not limited just to the team(s): this fact is the main added value of OPS.

3.3 Structure of Competitions

At the beginning, OPS were exclusively based on contests where students receive problems that are asked to solve in a well-defined time frame. Students participating to contests of this kind are divided in three *levels*: (i) grades 4-5; (ii) grades 6-8; (iii) grades 9-10.

In the first editions, OPS consisted in three series of *team contests*, one for each level. Each team is made up of 4 students, possibly with gender equality. In the last few years two more series of contests of the same nature have been added, for individual students of the second and third levels. In each OPS edition, all series have the same number of contests, usually six (see Table 1). All the contests except the last two are *training contests*, which means that the score achieved by a team (or an individual) is only provided as a feedback. The last two contests are *competitive*: the first one is used to select, for each series, the best teams (or individuals) of each Italian region, while the final one decides the winners.

The team contests consist of 12 problems: the amount of work to be done in the allotted time (120 minutes for the training contests and 90 minutes for the competitive ones) to solve them largely exceeds the effort that can be expected from a single participant; hence planning, collaboration, specialization and team-working are necessary. The individual contests consist in 8 slightly simpler problems, in the same amount of time (120 and 90 minutes).

The given problems are prepared by means of a controlled process, with a strict time schedule, which involves two committee: the *problem editors* (mainly

university researchers with experience in computer science education) define the problems and write texts and commented solutions, while the *problem reviewers* (mainly selected teachers with experience in the OPS topics) check for errors and difficulty.

More recently, along with the above "classic" OPS, that remain the core contests of the competition, other team contests have been added as "side" activities:

- *Coding* (grades 1-5 and 6-8), where students engage with creative programming activities, realizing a project on a topic chosen by the committee, with suitable environments (e.g. Scratch);
- *Programming* (grades 9-13), where students solve a complex programming task (e.g. implement a graph algorithm);
- *Making* (grades 1-5, 6-8, 9-13) where student realize a physical project using electronic boards (e.g. Arduino) on a topic chosen by the committee.

In the following we refer only to "classic" OPS, with particular focus on team contests, the original and distinguishing contests of the OPS.

3.4 Topics

OPS proposes at the three levels the same kinds of problems, with differences (among the levels) in size or abstractness. Each year five o six topics are chosen: from these, "standard" problems come up. In the last school years the chosen topics where: formal deduction from a set of rules, paths in a maze or in a chessboard, knapsack, hierarchies, graph covering, graph traversal, cryptography, subsequences and reading/tracing of procedures written in a pseudo-code.

A semi-formal syntax about terms, lists, strings is used in problems and answers. Additional topics are comprehension of a text (in Italian) and non "standard" problems, each time of a new kind (and often formulated in English), to stimulate creativity and the ability to deal with new situations.

3.5 Examples of standard problems

Examples of a "standard" problem are shown in Appendix - figures 4, 5, and 6 (originally in Italian, translated for this work).

The first problem was part of the second training for primary school: problems, on the same subject (graphs), in following contests were gradually more complex, and the solution had to be chosen among up to a dozen of paths. The second problem was part of the fifth training for secondary school. The third problem was part of the final contest for lower secondary school: as you can see it involved a lot of programming structures (e.g. variable assignments, vectors, while loops, if-then-else), introduced gradually during the training contests.

3.6 Commented solutions

A key feature of the OPS is that, along with the solution to each problem, after each contest, "comments" are provided; often this section makes up an outline for the in-depth study and analysis that teachers are invited to follow along with the students. In this way, knowledge and skills are spread among teachers and students. As you can see in the examples in Appendix, as the level and the difficulty grows, the comments become much longer and elaborate.

3.7 The training contests

In a training contest, on a fixed date, problems are proposed on a secure web platform and are accessible for a suitable interval of time (e.g. 9:00 - 17:00). Schools connect to the platform when they deem suitable (within the given interval of time) and, under the supervision (and the guarantee of fairness) of local teachers, the teams and the individual students participate to the contest. Problems are administered and their solutions should be entered within 120 minutes from the first access. During the contest each team/individual is allowed to use any additional material as it sees fit (books, notes, additional PCs, browsing the Internet, etc.), the ability to quick retrieve and organize relevant information being one of the key components of effective problem-solving. When the access to problems is closed, correct solutions and comments are made available; then answers are evaluated and teams ranked.

Three to five different training contests are organized in this way from November to March: the difficulty of the problems is gradually increasing. Then, for each level, each school selects teams and individual students for a regional competition; the winning teams and individual students of each region (20) meet in Cesena for the final competition.

3.8 The final contest

The final contest is a fierce competition; around twenty teams and twenty individual contestants for each level are invited in Cesena. The final contest takes place in the Department of Computer Science and Engineering, University of Bologna, under control of a Jury and with strict rules.

1. Each team has the same type of workplace available, which includes one workstation.
2. Teams are allowed to bring books, notes and an extra PC.
3. Participants are not allowed to communicate with other persons during the contest, except with members of their own team and the Jury.
4. Use of hardware other than watches, medical equipment, the PC and the provided workstations is not allowed.
5. A team can be disqualified for any activity that jeopardizes the contest.
6. When a team presume that a problem is ambiguous or incorrect, the team can ask a clarification to the Jury. The Jury will respond and if the answer is relevant to all teams, the Jury will communicate the answer to all teams.

The Jury determines the final team ranking as follows.

- Teams are sorted by the number of accepted solutions, in non-increasing order.
- When two or more teams have the same number of accepted solutions, these teams will next be sorted by total time, in ascending order. The total time is the time passed between the start of the contest and the time at which the last solution was submitted.
- There is no penalty for unsolved problems.

4 Participation and results

OPS were proposed, organized and developed with the guidance of several "working hypotheses", among which those that participation to OPS: (i) fosters and develops problem-solving skills, both at the personal and the class level; (ii) enhances the success level in standardized performance tests, like Italian INVALSI, or OECD's PISA; (iii) is an effective way to induce computational thinking habits in the students; (iv) facilitates the introduction of explicit, disciplinary teaching of computer science. Although we have qualitative indicators confirming these hypotheses (especially the feedback from instructors who have proposed OPS for more than 10 years), available data are not sufficient to discuss here all these far-reaching goals. We report only on the first one.

The first 10 editions of the OPS have recorded a noteworthy participation, both in terms of students and teachers involved, as shown in Table 1. These data show that the OPS are among the most successful initiatives, in terms of participation, for the diffusion of computational thinking in Italy.

Table 1. Participation to the first 10 editions of the OPS

School Year	Number of Rounds	Teachers	Students
2008/2009	3	550	10200
2009/2010	5	500	9850
2010/2011	6	400	7400
2011/2012	6	630	11400
2012/2013	4	880	16500
2013/2014	6	400	15100
2014/2015	7	500	12000
2015/2016	6	830	16700
2016/2017	6	850	23357
2017/2018	6	730	22225

4.1 First results

The comparative impact of OPS on participants is difficult to establish due to a selection effect: in fact, participation in OPS is voluntary, so we expect

that the best and most innovative teachers are better represented among those participating in OPS. Moreover, we cannot rule out that participating teams are made up of some of the smarter students in their classes. This selection bias is difficult to asses and avoid in data analysis; a comparison of the educational careers of those who have participated in OPS and students from another sample is an extremely delicate task (see Section 6).

On the other hand, the training effect on students is clearly visible, especially because problems, from contest to contest, become progressively more difficult. Some examples can be given, by considering the average scores obtained by the teams who participated in the project during the last 5 editions[9] on algorithmic problems, which are the most indicative of computational thinking competencies. For grades 4-5 (see Figure 1) the trend, in each edition, has no marked fluctuations. This, related to the progressive increase in the difficulty of the problems, indicates a development of the problem-solving skills of the pupils. It can be noted that in the second or third contest there is often a slight lowering of the performances, which corresponds to the introduction of problems that are considerably more difficult than those proposed in the first contest of each edition.

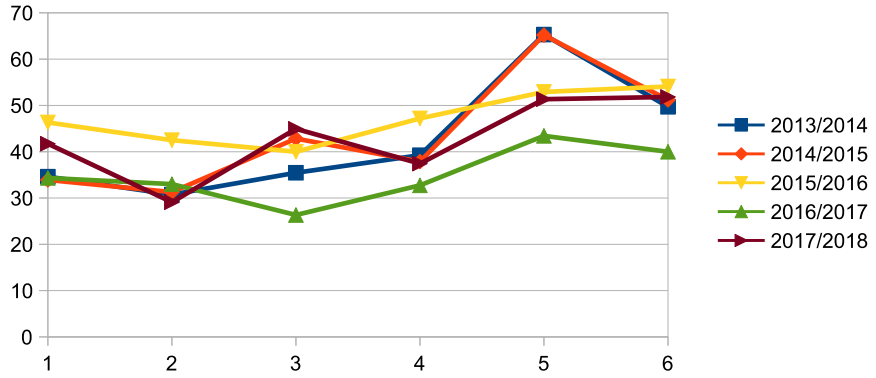


Fig. 1. Average score in algorithmic problems for grades 4-5

Data for grades 6-8 (Figure 2) highlight, in most of the editions, two contests in which the scores decrease, which is explained by a more rapid increase in the difficulty of the problems in this level. The trend, however, remains positive and strong growth in the two competitive contests.

[9] These editions are comparable since all of them had 6 rounds, with the only exception of 2014/15. For that specific school year, we noted that two consecutive intermediate rounds had very similar results at all levels, so we normalized it to 6 rounds too.

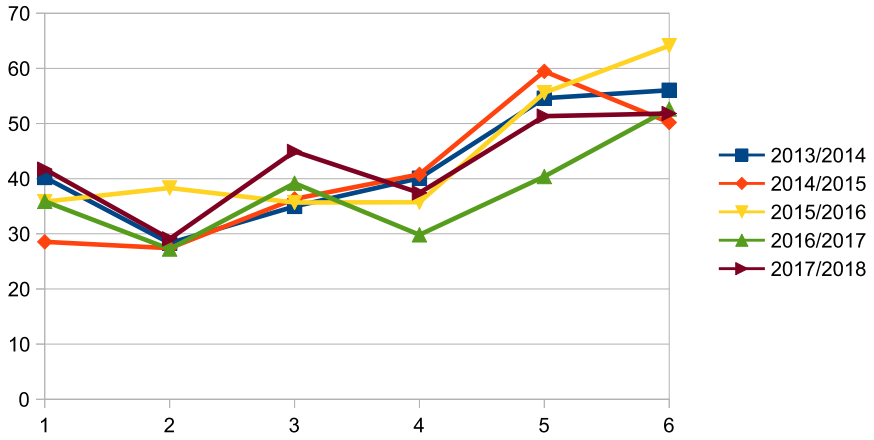


Fig. 2. Average score in algorithmic problems for grades 6-8

At the third level (Figure 3) we register more differences between one edition and another. This can be explained by the fact that the set of problems proposed at this level changes more from year to year. In any case, the trends are similar to those shown in the other levels, with a more marked improvement in correspondence with the competitive contests, typical of the greater commitment that students to this level of maturation can profuse.

Another impact difficult to assess is on teachers; it was noted that the competitions (at least in computer science) encourage teachers to learn more to support their students; currently, besides a qualitative confirmation, it does not seem possible to measure this effect.

5 OPS and gender

A recent European Parliaments Policy Department for Citizens Rights and Constitutional Affairs study attempts to reveal the links between the different factors which prevent women from having equal access to digital technology [5]. While ICTs are recognised as having the potential to promote gender equality and womens empowerment, a digital gender divide has been identified, whereby women access and use ICTs less than men. Increasing the number of women in ICT-related education could be a policy consideration. Women may also have more to gain from ICT than men, in time, freedom and opportunities. Addressing the underlying causes of digital gender disparities is vital, as dealing with the symptoms without fighting the causes would lead to superficial measures.

We attempted to evaluate the involvement of girls and boys in OPS by calculating the number of participants per gender on the three school levels. The

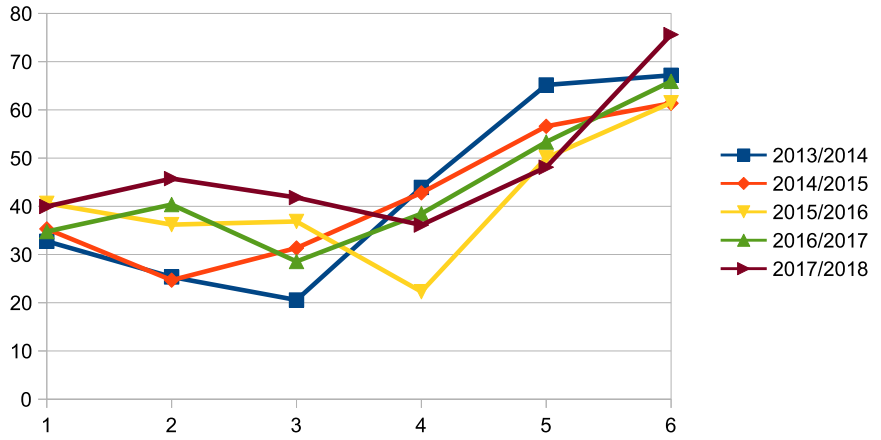


Fig. 3. Average score in algorithmic problems for grades 9-10

goal is to try to understand at what time the disinterest of the girls towards the computer disciplines occurs, so as to suggest the best time to propose remedies. We evaluated the numerical participation of girls and boys on the three levels of school. It is almost equal in elementary school; then the number of girls involved in competitions concerning first and second grade of secondary school drastically decreases. These results are similar to those found in another, independent MIUR initiative to promote Informatics and computational thinking [2].

Based on our experience, the measures that can be taken to combat digital gender gap must be implemented in the early years of elementary schools. Doing it later can be useless because the factors that cause the digital gender gap have already pushed away the female component.

6 Conclusions

We presented OPS, an Italian initiative to disseminate problem-solving skills and computational thinking activities with a trans-disciplinary approach. OPS are targeted at all the compulsory education age range and are aimed to reach all students, and not only exceptional ones. In the past few years, the number of participating students and teachers has been notably growing, as a testament to the validity of the initiative.

Among the advantages of the OPS approach we may cite: interactivity, collaborative work inside the group, active participation, challenge versus duties, and motivation for the students to explore their own topics, on problem-solving skills and computational thinking, to support the challenges. The best feedback obtained is probably the feeling of the involved students and teachers.

Preliminary data show an increase in scores of algorithmic problems from the first contest to the last in each school year, at all levels. Given that the difficulty of problems increases from one context to the next, this may indicate an increase in algorithmic problem solving skills in pupils.

Large direct feedback from the teachers involved in OPS makes evident a positive correlation between participation to OPS and general performance of students (as measured in standard tests, like PISA). This has been a working hypothesis of our work since the firsts editions of OPS. Turning this hypothesis into a scholarly acceptable evaluation is one of the main directions of our current work on the evaluation and assessment of OPS.

References

1. Combéfis, S., Wautelet, J.: Programming trainings and informatics teaching through online contests. Olympiads in Informatics **8**, 21–34 (2014)
2. Corradini, I., Lodi, M., Nardelli, E.: Computational thinking in italian schools: Quantitative data and teachers' sentiment analysis after two years of "programma il futuro". In: Proc. 2017 ACM Conf. on Innovation and Technology in Computer Science Education. pp. 224–229. ITiCSE '17, ACM, New York, NY, USA (2017), http://doi.acm.org/10.1145/3059009.3059040
3. Dagienė, V.: Sustaining informatics education by contests. In: Teaching Fundamentals Concepts of Informatics, 4th International Conference on Informatics in Secondary Schools - Evolution and Perspectives, ISSEP 2010, Zurich, Switzerland, January 13-15, 2010. Proceedings. pp. 1–12 (2010)
4. Dagienė, V., Futschek, G.: Bebras international contest on informatics and computer literacy: Criteria for good tasks. In: Mittermeir, R.T., Sysło, M.M. (eds.) Informatics Education - Supporting Computational Thinking. pp. 19–30. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
5. Davaki, K.: The underlying causes of the digital gender gap and possible solutions for enhanced digital inclusion of women and girls. Tech. rep., Policy Department for Citizen's Rights and Constitutional Affairs – European Parliament's Committee on Women's Rights and Gender Equality (2018)
6. Kelevedjiev, E., Dzhenkova, Z.: Tasks and training the youngest beginners for informatics competitions. Olympiads in Informatics **2**, 75–89 (2008)
7. Kiryukhin, V.M.: The modern contents of the russian national olympiads in informatics. Olympiads in Informatics **1**, 90–104 (2007)
8. Raman, R., Vachharajani, H., Achuthan, K.: Students motivation for adopting programming contests: Innovation-diffusion perspective. Education and Information Technologies (Apr 2018), https://doi.org/10.1007/s10639-018-9697-3
9. Verdú, E., Lorenzo, R.M., Revilla, M.A., Regueras, L.M. (eds.): A new learning paradigm: competition supported by technology. CEDETEL, Sello Editorial, Madrid (2010)
10. Verhoeff, T., Horváth, G., Diks, K., Cormack, G.: A proposal for an ioi syllabus. Teaching Mathematics and Computer Science **4**(1), 193–216 (2006)
11. Wieman, C.: Applying new research to improve science education. Issues in Science and Technology **29**(1), 25–32 (2012), http://www.jstor.org/stable/43315691
12. Wing, J.M.: Computational thinking. Commun. ACM **49**(3), 33–35 (Mar 2006), http://doi.acm.org/10.1145/1118178.1118215

Appendix

PROBLEM

In a map there are towns and roads between them: this situation can be abstracted with a *graph* in which *nodes* represent cities and the *arcs* represent roads. An arc (road) of the graph can be described by a term like:

arc(n4,n8,5)

which means that there is an arc between node n4 and node n8 and its length is 5.
Two nodes connected by an arc are said *adjacent*.
A *path* is (described by) a list of adjacent nodes. A *simple* path does not contain repeated nodes.

Consider a graph described by the following list of terms:

arc (n1,n5,3) arc(n3,n1,6) arc(n2,n4,2)
arc(n6,n3,6) arc(n5,n2,5) arc(n4,n6,1)

Draw the graph and determine:

1. the list L1 describing the *shortest* path between n1 and n6 and calculate its length K1;
2. the list L2 describing the *longest* simple path between n1 and n6 and calculate its length K2.

Enter your answer in the following table.

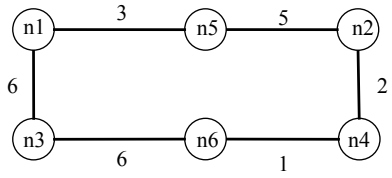
L1	
K1	
L2	
K2	

Solutions:

L1	[n1, n5, n2, n4, n6]
K1	11
L2	[n1, n3, n6]
K2	12

COMMENTS TO THE SOLUTION

After a few attempts the graph can be drawn in such a way that the arcs do not intersect, as in the following figure.



Note that a major difficulty lies in drawing the graph so that the arcs do not intersect; it is necessary to proceed by trial and error, by placing nodes in various ways until the drawing is satisfactory; *this can be done in many ways*.
Note also that the lengths that appear in the terms (that represent roads) are *not* proportional to those of the arcs of the graph (that are "*symbolic*" straight line segments).
The graph describes a "circuit" and it is easy to see that the possible paths between n1 and n6 and the relative lengths are:

path	**length**
[n1, n5, n2, n4, n6]	11
[n1, n3, n6]	12

Fig. 4. Example of a standard problem (team contest, primary school, grades 4-5, second training contest)

PROBLEM

A *deduction rule* can be described by a term like

rule(3,[a,b],c)

which means that the rule (number) 3 permits to deduce **c** from (that means knowing) **a** and **b**; **c** is said *consequent*; **a** and **b** are said *antecedents*.

Rules can be chained for complex *deduction procedures*; a deduction procedure can be described by a list of rule numbers.

Consider the following rules:

rule(1,[f,g],h)	rule(2,[q,f],b)	rule(3,[p,q,c],a)	rule(4,[m,p],q)
rule(5,[a,b,c],d)	rule(6,[m,p],f)	rule(7,[p,h],c)	rule(8,[m],p)
rule(9,[e,f],c)	rule(10,[f,b],i)	rule(11,[c,e],r)	rule(12,[f,s],e)
rule(13,[e,t],b)	rule(14,[a,u],f)	rule(15,[e,b],v)	rule(16,[u,d],t)

Determine:

1. the list L1 that describes the shortest procedure to deduce **a** knowing **c**, **m**;
2. the list L2 that describes the shortest procedure to deduce **c** knowing **m**, **g**;
3. the list L3 that describes the shortest procedure to deduce **d** knowing **m**, **g**;

Criterion for the lists. List the (numbers of the) rules in the order that corresponds to the sequence of application of the rules: the first element (from the left) of the list must be the number that corresponds to the first rule to be applied; if there are several rules that apply at the same time, give priority to the one with lower number.

L1	
L2	
L3	

SOLUTION

L1	[8,4,3]
L2	[8,6,1,7]
L3	[8,4,6,1,2,7,3,5]

COMMENTS TO THE SOLUTION

To solve the problem, you can use the backward (or top-down) method, that is start from the unknown and try to find a rule to derive it. If there is a rule whose antecedents are all known (the data), the solution is found; otherwise you look for a rule whose antecedents are not *all known* and you continue to look for rules to derive the unknown antecedents (that appear in the rule).

For the first question, it is immediate to check that **a** appears as consequent only of rule 3 which has as its antecedent **p**, **q** and **c**: the first two are unknown, the last is given; **p** is deductible only by the rule 8 directly from **m** (given); **q** is deductible only with rule 4 from **m** (given) and **p**, just deduced. The procedure is illustrated by the tree in the figure below. The list L1 is [8,4,3].

For the second question, we immediately verify that **c** appears as the consequent in the rules 7 and 9. The rule 9 has antecedents **e** and **f**, both unknown; **e** is deductible by the rule 12 which has antecedents **f** and **s**: the latter can not be deduced with any of the given rules. Therefore, to deduce **c**, only rule 7 is usable; this rule has antecedents **p** and **h** both unknown; **p** is deductible only with rule 8 directly from **m** (given); **h** is deductible only with rule 1 from **f** (unknown) and **g** (given). At the end **f** is deductible by rule 6 from **m** (given) and **p** (already deducted) or by rule 14 from **a** and **u** (both unknown): it is clear that we must choose the first alternative. The proceedings is illustrated by the tree in the figure below. The list L2 is [8,6,1,7].

Fig. 5. Part A

To answer the third question, we immediately verify that d appears as a consequent only in rule 5 that has antecedent **a**, **b** and **c** all the unknowns:

- to deduce **a** can be used partially what was said for the first question; rule 3 has antecedent **p**, **q** and **c**: **p** is deductible only by rule 8 directly from **m** (given); **q** is deductible only by rule 4 from **m** (given) and **p**, just deducted. **c** is deductible by the rule 7 or by rule 9: the first should be applied because of the two antecedents, **p** and **h**, one has already been deducted; **h** can be deducted only by rule 1 that has antecedents **f** (unknown) and **g** (given); **f** is deductible only by rule 6 that has antecedents **m** (given) and **p** already deducted;
- **b** can be deducted only by rule 2 which has antecedent **q** and **f**: both already deducted;
- **c** has already been deducted .

The overall process is illustrated by the tree to the right of the figure below. The list L3 is [8,4,6,1,2,7,3,5]: to build it is essential to keep in mind the criterion for the list given in the problem.

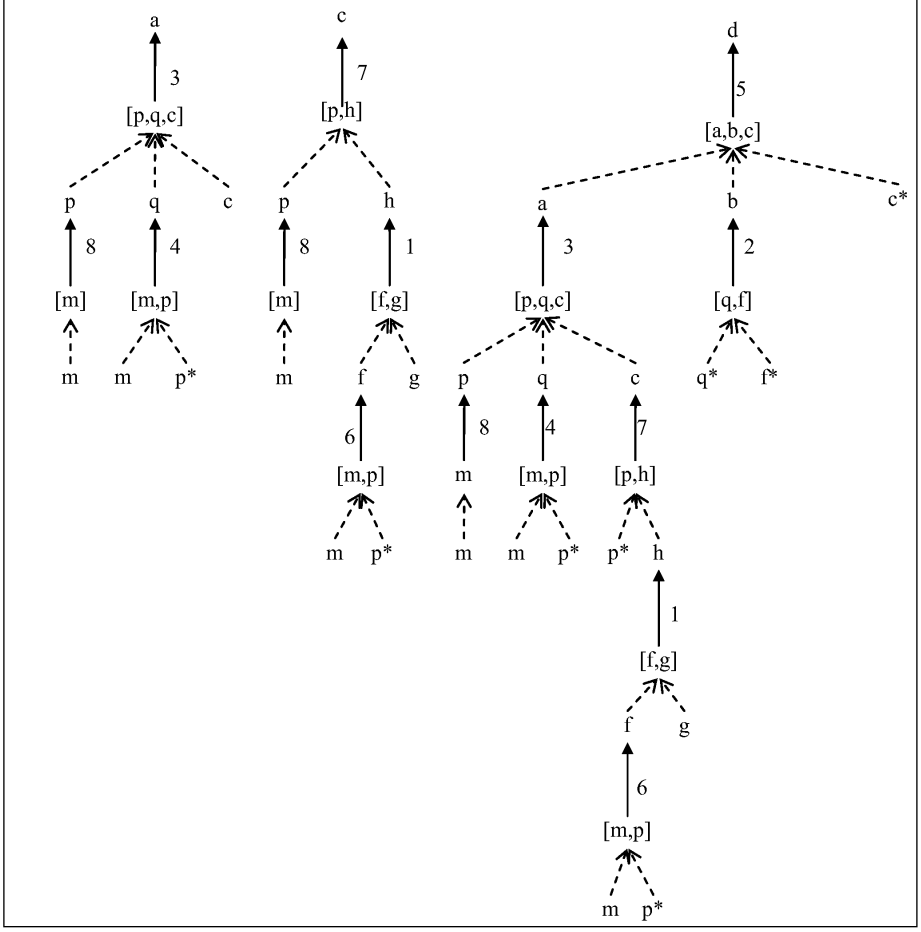


Fig. 5. Part B. Example of a standard problem (secondary school, grades 9-10, fifth training contest)

PROBLEM
Consider the following procedure ALFA.

```
procedure ALFA;
variables I, N, K, S integer;
variables A(1:8) vector of integer;
A ← [1,12,23,34,45,56,67,78]
I ← 1;
N ← 50;
K ← 0;
S ← 0;
T ← 0;
while K = 0 do;
        if A(I) < N
                then    S = S + A(I);    I = I+1;
                else    K = I;
        endif;
endwhile;
while K = 0 do;
        T ← T + A(K);
        K ← K + 1;
endwhile;
output I, K, S, T;
endprocedure;
```

Determine the output values of I, K, S, T and enter them in the following table.

I	
K	
S	
T	

SOLUTION

I	6
K	9
S	115
T	201

Fig. 6. Part A

COMMENTS TO THE SOLUTION

The program calculates in S the sum of the values of the vector A that are lower than N, and in T the sum of the values greater (or equal) to N. The variable I will indicate the index (the position) of the first value greater than or equal to N in the vector.

The values of I, K, S before the first while loop and after each of the iterations of the (body of) cycle are shown in the following table.

	Value of I	Value of K	Value of S
before the loop	1	0	0
after the first iteration	2	0	1
after the second iteration	3	0	13
after the third iteration	4	0	36
after the fourth iteration	5	0	70
after the fifth iteration	6	0	115
after the sixth iteration	6	6	115

The values of K and T before the second while loop and after each of the iterations of the (body of) cycle are shown in the following table.

	Value of K	Value of T
before the loop	6	0
after the first iteration	7	56
after the second iteration	8	123
after the third iteration	9	201

Fig. 6. Part B. Example of a standard problem of pseudo-code reading (lower secondary school, grades 6-8, national final contest)