Olympiads in Informatics, 2026, Vol. 20, 145–160

DOI: https://doi.org/10.15388/ioi.2026.10

Developing Computational Thinking for Participation in Programming Olympiads

Michael DOLINSKY
Faculty of Mathematics and Technologies of Programming, F. Skorina Gomel State University, Sovetskaya str., 104, Gomel. 246019. Republic of Belarus
e-mail: dolinsky@gsu.by

Abstract. This paper examines how to improve schoolchildren's ability to solve complex programming problems. The author classifies the problems of the International Olympiad in Informatics finals as such. Contrary to the opinion of many schoolchildren and teachers who believe that such skills cannot be developed through training, but are largely determined by hereditary factors, the author took as allies Polya, Altshuller, and Vygotsky, who developed a methodology for the targeted development of thinking. In addition, the article presents the author's experience in teaching programming and preparing students for successful performance in programming Olympiads, including recommendations for solving problems during Olympiads, developing skills for solving complex problems during training, and using Codeforces for cognitive and psychological training. In addition, the author describes in detail the system of developing thinking in teaching programming to schoolchildren from grades 1 to 11 based on the DL.GSU.BY - an instrumental distance learning system developed under the author's supervision.

Keywords: computational thinking, international olympiad in informatics, psychology of learning, DL.GSU.BY, Codeforces.com.

1. Introduction

From 1997 to 2024, 17 schoolchildren taught by the author (Dolinsky, 2024a) have won a total of 29 medals at the International Olympiad in Informatics. Of these, 7 are gold, 11 are silver, and 11 are bronze. At the same time, 2 schoolchildren won gold medals, 7 won silver medals, and 8 won bronze medals. It is important to note that the last gold medal was won in 2012, the last silver medal in 2018, and the last bronze medal in 2024. When the last participant was asked why he did not perform better, he answered that he could not come up with complete solutions to the problems. When solving them further, it turned out that he had enough theoretical knowledge to solve for silver, at least. The penultimate bronze winner (in 2021) answered the same question in a similar way and also added that, most likely, the skill of coming up with solutions is not trained but is innate. If we add that in 2024, three of the author's students won bronze medals, then, naturally, the author had a

question about how to overcome this barrier. This article is precisely the author's attempt to answer this question.

The International Olympiad in Informatics (IOI) was first held in 1989 in Bulgaria. The main goal of the IOI is to promote computer science (CS) among young people and to discover and stimulate young talent in CS. The IOI offers an annual competition in solving algorithmic problems, with solutions implemented in one of several modern programming languages (Verhoeff, 2009). 370 people from 94 countries took part in IOI 2024.

1.1 How to teach problem solving - optimistic foundations

The most famous work in this direction is Polya (1945), which outlines a methodology for teaching problem-solving in mathematics. Polya identifies four stages of problem-solving: understanding the problem, making a plan to solve it, implementing the plan, and analyzing the solution. It is also recommended to develop both heuristic and logical thinking skills. In addition, useful actions at each stage are specified. Particular attention is paid to the psychological aspects of problem-solving, with the final statement that teaching the art of problem-solving is also the education of the will.

Another well-known work in this direction is TRIZ - Theory of Inventive Problem Solving (Altshuller,2004). This is how the author of TRIZ describes the contents of his book. "The book is intended for a wide range of readers, primarily engineers, developers of new technology, inventors, and students of technical universities. The issues of creative process management are generally considered using inventive examples, so the book is also addressed to readers not involved in technical creativity. The book is of particular interest to scientists and researchers in cybernetics, artificial intelligence, and the psychology of thinking. The principles of thinking management in solving inventive problems (namely, principles, not specific formulas and rules) can apparently be transferred to the organization of creative thinking in any area of human activity."

Vygotsky (1997) is important in teaching schoolchildren how to think. According to Vygotsky, the learning process is understood as a collective activity that occurs in cooperation between an adult and a child. Only that learning is good that anticipates development; that is, learning should precede development and be its source. Then it awakens and brings to life many other functions that lie in the zone of proximal development (ZPD). According to Vygotsky, the ZPD is determined by the content of tasks that the child cannot yet solve independently but can solve in joint activities with an adult. What is initially accessible to the child under the guidance of adults then becomes his own property (skills, abilities).

1.2 Third-party recommendations for learning to solve complex programming problems

The knowledge requirements for participants in the International Olympiad in Informatics are set out in the IOI Syllabus (2025), which aims to determine: the minimum knowledge required for participation in the IOI; whether the proposed task is suitable for the IOI; and the areas of preparation for participants in the IOI. The IOI Syllabus contains the follow-

ing sections: Mathematics; Discrete structures: functions, relations, sets, graphs, and trees; Fundamentals of programming and algorithmization; Data structures; Development, testing, and debugging of programs.

An important tool for preparing schoolchildren for programming olympiads is Codeforces platform (Mirzayanov, 2020), which has been operating since 2010. It regularly hosts personal Olympiads of four difficulty levels. Based on the Olympiad results, the ratings of Olympiad participants are calculated and updated. The author's analyses are posted for all problems. The authors assign tags to all problems, indicating their topics, and over time, each problem receives a difficulty rating based on the number of people who have solved it. The modes for completing problem solutions, the ability to filter problems by tags and ratings, and virtual contests are supported. The site provides maintenance for users' personal blogs. As a result, the blogs contain a lot of useful educational and developmental information. The site provides an API for working with the information it collects. Thus, the site, among other things, supports two important tasks: systematic training and assessment of training effectiveness.

In particular, CF ratings can be used to predict results in upcoming Olympiads. For example, the paper IOI_vs_CF (2024) provides the following data on the relationship between CF ratings and IOI 2024 medals (Table 1):

Table 1

Relationship between CF ratings and IOI 2024 medals

Medal	Minimum	Median
Gold	2029	2469
Silver	1501	2274
Bronze	1637	2082

The IOI-CF-Rating (2025) paper provides data on the ratings of team participants from countries planning to participate in IOI 2025 (57 countries, 228 participants as of July 1, 2025).

Daniliuk (2018) explains how to read programming task statements. The result of reading the problem statement is usually a pure mathematical model, freed from the problem legend. Mirzayanov (2015) outlined his methods for finding solutions to complex programming problems: "Remember everything", "From the particular to the general", "Bold hypothesis", "To solve a problem, you must think like a problem", "Let's think together", "Choose a method", "Print and view", "Google". Maksimovski (2020) proposed a special strategy for solving problems. Kulkov (2016) collected a variety of techniques for solving non-standard programming problems. When solving the Olympiad, it is important to choose the right strategy for solving a set of problems. Korotkevich (2017), the winner of 7 IOI medals (the best participant in IOI history with 6 gold and 1 silver medals), presented his own strategy. Ray (2024) emphasizes the importance of automated testing to

identify errors. Sahgal (2022) suggests training on problems 200-300 points above your Codeforces rating to develop the skill of solving constructive problems. Luo (2023) gave interesting observations on problems and ways to prepare for their solution. The author gives an unconventional answer to the question “How to improve your Codeforces rating?” - solve math problems.

2. Author’s approaches to solving the problem

2.1 Work during the Olympiad

2.1.1 Recommendations for solving the Olympiad problem set

Read and understand the conditions of all the tasks. If possible, come up with and implement the simplest (brute force) solutions. Score points for each task. Find out the potential difficulty of each task. Work on each task in order of increasing difficulty as progress is made. If you hit a dead end, switch to another task after a predetermined period.

2.1.2 Recommendations for solving each problem of the Olympiad

1. Reading the problem text.
   You need to read more slowly, more carefully, more meticulously. DO NOT miss any important facts or comments. After reading, reformulate the condition IN WRITING, leaving the mathematical formulation of the problem. Then, reread the conditions to supplement your problem statement with the omitted facts. Emphasize the facts that should influence the decision. Set the complexity (N, NlogN…) of the complete solution.
2. Capturing potential ideas/solution algorithms.
   Immediately after reading the conditions, make a list of possible ideas applicable to this task, as complete as possible, so that the right idea is definitely included.
3. Develop a minimal set of tests on which you will check the solution (samples, base case, all possible special/edge cases).
4. Mental rejection of ideas by analyzing their application to a fixed set of tests, and selection of a working idea.
5. Slow, thoughtful coding, ideally eliminating errors in the code.
6. If necessary, built-in control of correct operation (assertions).

Globally

Perhaps you need to spend more time thinking and less time writing (especially debugging).

2.2 Preparation for the Olympiad

2.2.1 Direct development of skills for coming up with solutions to complex problems

Carry out the work described below at least once a week with a gradual increase in the rating of the selected tasks:

1. Choose on Codeforces THREE (like on IOI) problems
   - of your current rating (or 100-200 more)
   - on one of the topics:
   - graphs, complex data structures, dynamic programming, constructive
2. Two hours to think about solutions
3. Read the analyses
4. Write in the relevant DL forum topic in the form
   - link to task
   - task rating
   - result (I came up with it myself or understood the analysis)

2.2.2 Regular participation in rounds on Codeforces

Regular participation in Codeforces rounds brings benefits in the following aspects in preparation for international olympiads in computer science and programming:

- Introduction to modern ideas for solving problems.
- Psychological training in solving problems under time constraints and pressure from the fact that "there is something to lose", since unsuccessful performance leads to a decrease in the rating on Codeforces, sometimes quite significantly.
- Completing the tasks that were not solved at the Olympiad, from the zone of one's proximal development (if the author's analysis is understood).

3. Technology of regular long-term work in the process of preparation for the Olympics

3.1 Distance learning system DL.GSU.BY

The distance learning system DL.GSU.BY (hereinafter, DL, an abbreviation of the phrase Distance Learning) automates assignment issuance, solution checking, table construction for results and ratings, and personalization of learning (Dolinsky, 2022a).

3.2 Early start

The simple, intuitive interface of the DL system, as well as special task packages "Learning to work with a mouse", "Learning to think", "Learning to read", "Learning to count", "Propaedeutics of words", "Learning words", provides the opportunity to begin learning programming from the earliest age. In fact, the prerequisites are the ability to walk and talk. In practice, for about 20 years, all students of Secondary School 27 in Gomel have been learning text programming from the 1st grade, subject to their parents' written application. In addition, preschoolers and students of all schools in Gomel and the Gomel region are also accepted for optional classes.

3.3 Personalized online learning

System DL provides personalized learning for each student (Dolinsky & Dolinskaya, 2020). Tasks are issued automatically. If the task cannot be completed, the student can click the "I don't know" button, which opens the tree-like system of lead-in tasks. Each lead-in task can have its own "I don't know" button. At the same time, the lead-in tasks include an "I understand" button that lets you quickly return to the task that caused the problem.

In fact, a student can study independently both at school and at home.

3.4 Specialized package of tasks "Learning to think"

The tasks were developed under the author's guidance on the development of the following basic mental operations, grouped into 5 groups:

- Operations on pairs: comparison, ordering, association.
- Operations on sets: union, intersection, subtraction.
- Operations on a set: classification, structuring, generalization.
- Logical operations: negation, conjunction, disjunction, equivalence, and implication.
- Complex operations: synthesis, memorization, analysis, imagination, analogy, abstraction, positioning.

The "Learning to Think" task package (Dolinsky, 2014) consistently develops the skills of all these thinking operations, while the student is offered tasks in the form of drawings, and as the tasks are completed, you need to click in the right places or move the drawings using the mouse or the "Arrow" keys.

3.5 End-to-end targeted development of thinking in the process of learning programming

In the course "Informatics", primary school students undergo sequential personalized programming training in the topics of "Number", "Char", "String", "String Length", "Posi-

tion of Char in String", "built-in Delete procedure", "built-in Copy function", "built-in Pos function". One-dimensional array. Two-dimensional array. Geometry. Strings. Sorting. Text problems.

The training is structured in such a way. At the beginning, a task is given. If the student can solve the task, he solves it; if not, he presses the "I don't know" button and enters the task package, which gradually teaches him how to solve it. At the same time, the main principle is always maintained: first the task, and only if the student cannot complete it independently does he receive an explanatory task. That is, the student is constantly pushed to think. If you can come up with it yourself, you will move on faster.

In addition to the tasks of teaching writing programs, there are regularly interspersed tasks that develop the above-mentioned basic thinking operations, but based on fragments of programs.

3.6 Parallel development of mathematical problem-solving skills

In addition to the course "Informatics" (where students in grades 1-4 learn programming) and "Basic Programming" (where students in grades 5-11 learn programming), there is a course called "Mathematics". It offers math problems. There is no training or explanation. There are only problem conditions, and you are required to either enter an answer or choose an answer from several presented. An important advantage is the huge number of problems of varying complexity from various sources. Including:

- Kangaroo (2001-2024) for grades 1-10 (in Russian)
- Beaver (2013-2018) for grades 1-11 (in Russian)
- Kangaroo (Canada, Pakistan, Portugal, United Kingdom)
- Beaver (Australia, Canada, Pakistan, United Kingdom)
- Canadian Math Contests
- United Kingdom Math Contests
- Texas University Math Contests

In addition, for grades 1-4, packages of tasks have been completed that accurately monitor the acquisition of knowledge in the mathematics program. The topics and subtopics for which tasks have been consistently prepared for all grades are listed below: 10 tasks for each subtopic.

1st grade

1. Comparison. Space. Time: Identification and difference by 1 feature. Mutual arrangement of objects. Time concepts. Classification by 1 feature. Classification by 2 features. Ordering. Comparison. Naming numbers from 1 to 10. Naming numbers from 11 to 20. Counting objects
2. Single-digit numbers: Numbers from 0 to 10. Addition and subtraction up to 10. More, less, equal. Solving simple problems
3. Two-digit numbers: Numbers from 11 to 20. Addition and subtraction up to 20.

Point, line, curve, segment. Centimeter, decimeter. Hour

2nd grade

Addition. Subtraction. Numbers from 1 to 100. Equations and inequalities with + and -. Right angle, rectangle, square. Broken line. Perimeter of triangle, quadrilateral, rectangle, square.

3rd grade

Multiplication. Division. Equations and inequalities with multiplication and division. Speed. Simple problems on motion. Ray, types of triangles. Three/four-digit numbers. Meter, kilometer. Gram, kilogram, ton.

4th grade

Numbers up to a million. Addition of three- and four-digit numbers. Subtraction of three- and four-digit numbers. Multiplication. Division. Length: mm, cm, dm, m, km. Weight: ton, centner, kg, gram. Time: century, year, month, week, day, hour, minute, second. Area.

The tasks are based on flash technology, so they can be completed in many different ways, including: transferring a drawing to a specific area of the screen, connecting pairs of dots, indicating the appropriate drawing, coloring specific areas of the drawings in the desired colors, choosing the correct answer from the suggested list of options, transferring words to specific areas of the screen, entering the desired numbers in the specified input fields.

A note about the obsolescence of flash technology. The flash task designer has been developed since 2010, and a huge number (thousands) of educational and control tasks have been made with its help. Currently, flash technology is not supported by many browsers by default. At the same time, to use flash tasks, there is an option to enable flash support, which is available in classroom browsers and on schoolchildren's computers. For security reasons, students are advised to work only on DL in such browsers.

3.7 Study of algorithms and data structures

Teaching algorithms and data structures is focused on students mastering the IOI Syllabus.

The course "Basic programming" contains a package of assignments "Accelerated course - 2013", which contains assignments on the topics:

1. "Introduction to Programming" includes tasks that require entering initial data and outputting the answer, i.e., to solve them, you don't need to know anything except input and output operators. At the same time, these tasks serve as propaedeutics for knowledge that may be needed in the future, including mathematics, programming languages, and competitive programming. At the moment, the topic "Introduction to programming" contains tasks on the following subtopics: formatted output; algebraic formulas; numerical operations (AND, OR, XOR, DIV, MOD, SHL, SHR); built-in functions and procedures (ABS, SQR, Odd ROUND, TRUNC, ORD, CHR, UPCASE, STR, VAL, LENGTH, COPY, DELETE, INSERT, POS); number systems.

2. “One-dimensional array” includes tasks for solving which it is necessary to know the declaration of a one-dimensional array, the IF condition operators, and the FOR and WHILE cycles. At the same time, it contains tasks on both standard algorithms for processing one-dimensional arrays and tasks on the propaedeutics of useful knowledge. Currently, the topic “One-dimensional array” contains tasks on the following subtopics: sum; counting; maximum/minimum; maximum/minimum number; cycle range; even/odd positions; long arithmetic (addition, subtraction, multiplication of a number by a digit, divisibility signs), prefix/suffix sums, maximums, minimums; counting sort; cycle parameters; cyclic counting; dec; search; consecutive; all different.
3. “Two-dimensional array” includes tasks for solving which it is additionally required to know the declaration of a two-dimensional array and apply nested cycles. Currently, the topic “Two-dimensional array” contains tasks on the following subtopics: subarray - sum; subarray - counting; counting by rows; array generation; array modification; counting by array perimeter; prefix sums, maximums, minimums; comparison of array rows.
4. “Geometry” includes problems that require the ability to find the distance between two points, a point and a set of points, between all points of a set, and then apply previously studied algorithms on one-dimensional and two-dimensional arrays. In addition, this topic includes problems on basic geometric concepts such as perimeter and area. Currently, the Geometry topic contains problems on the following subtopics: rectangle, Manhattan distance, distances from one point to several points, distance between identical points of two sets, adjacent distances, distances between all pairs of points, distances between all pairs of points of two sets, and area of a polygon.
5. “Strings” includes tasks that require knowledge of the data types symbol, string, string array, and the ability to invent and debug your own algorithms. In fact, this topic is key to assessing the student’s potential. Currently, the “Strings” topic contains tasks on the following subtopics: cyclic shift to the right; if; reversing a string; counting in a string; maximum in a string; Ord; searching in a string; bracket strings; lengths of array strings; counting in an array of strings; generating a string; generating an array of strings; converting a sentence to an array of words; forming arrays of strings; subarray of characters; array of strings - editor; analysis of all cyclic shifts of a string.
6. “Sorting” includes tasks on the ability to apply the algorithm of sorting by exchange, bubble, counting, and includes tasks on the following subtopics: only sorting; fixed number; fixed numbers; post-condition; variable numbers; variable range of numbers; sorting with numbers; sorting by counting; compression of coordinates; all different in ascending order.
7. “Text Problem” contains problems with original texts of problem conditions from the Belarusian Republican or Regional Olympiad (usually one or two pages), in which the task is modified in such a way that its solution does not require knowl-

edge greater than that used to solve problems from topics 1-6. The main difficulty for the participant is to extract the algorithmic formulation of the problem from the text condition. At the moment, Topic 7 includes subtasks on the following algorithmic subtopics: one-dimensional array: sum, sum + condition, counting, maximum, maximum number, minimum, minimum number, conditional sum, conditional minimum, element selection, counting sort, divisors, condition, cycles; two-dimensional array: maximum number in column, row sum, maximum in array, adjacency matrix; row array: counting by array; counting by rows.

8. "Elements of Number Theory" includes both digit-searching problems and problems requiring preliminary study of the relevant theory. Currently, the topic "Elements of Number Theory" includes subtasks on the following topics: For loop; nested For loops; While loop; For + While; divisors; primality testing by definition; Sieve of Eratosthenes; number systems; bit processing; submasks.

Also in the course "Basic programming", there is a package of tasks "Olympiads 9-11 grades", which contains tasks on the following topics:

9. "The Greedy Algorithm" includes problems that require pre-sorting of the input data. Currently, this topic includes problems on the following subtopics: Quadratic sorting; Sort with numbers; Counting sorting; Quicksort; Quicksort + while; Quicksort with numbers; Comparison function; 1 2 3 sorting; Correct bracket sequence; Maximum depth of bracket expression; Two arrays; Bid selection; Deadline and price; Minimum cover; Brute force + greedy; Stack + greedy; Ad hoc.
10. "Queue" includes tasks that require knowledge of the theory on the topic to solve. Currently, this topic contains tasks on the following subtopics: knight; labyrinth; three-dimensional labyrinth; pieces; three-dimensional pieces; knight with a dynamic list of moves; numerical sequences; queue exploration; dec; 01-BFS; two queues; bit processing.
11. "Recursion" includes tasks that require using a recursive procedure or function call to solve the problem. Currently, the Recursion topic contains tasks on the following subtopics. The set of all subsets: output of one way to make a sum M, output of all ways to make a sum M, number of ways to make a sum M, number of ways to make a sum not less than M, maximum sum not exceeding M, minimum excess of a sum, subset with the maximum number of suitable elements, forbidden subsets, sum of K subsets. Combinations: quantity, output. Placements. Permutations. Permutations with repetitions. Bracket expressions. Gray code. Fast exponentiation. Number generation. By definition. All subsets of rows of a two-dimensional array. On a two-dimensional array. Divide and conquer. Recursion with memoization. Recursion with memoization by profile.
12. "Dynamic programming and recurrence relations" includes problems of the subtopic: one-dimensional array: all sums, subsequence of maximum length, number of subsequences of maximum length, subsequence of maximum length in O(N*LogN), knapsack, sum of several previous, sum of several previous with answer recovery, maximum of sums, partition into subarrays, prefix sums, prefix

maxima, suffix minima, permutation-number; two-dimensional array: sum of several previous, maximum of several previous, minimum of sum of several previous, sum of maxima of several previous, maximum frame, prefix sums.

13. "Graphs" includes tasks that require graph analysis and processing and contains the following subtopics. Vertex enumeration. Vertex degree. Adjacency matrix. Queue: checking a graph for bipartiteness, shortest paths (Dijkstra and Floyd algorithms), articulation points. Recursion: path with maximum number of edges, reachability matrix, sources and sinks, vertex reachability, unreachable vertices, connectivity, connected components, strongly connected components, dominance sets, finding cycles, Euler cycle, Hamiltonian cycle, path in a directed graph, topological sorting, maximum matching, Kuhn's algorithm. Definition of a tree, number of edges on the path to a vertex, tree diameter, least common ancestor, order of visiting vertices by depth-first search, centroid decomposition, Huffman character encoding, binary tree, quaternary tree. Minimum spanning tree: Prim and Kruskal algorithms. Disjoint set systems. Strategy games. Hidden graphs. Euler formula. Maximum flow.
14. "Complex data structures" includes tasks that require theoretical knowledge of the following topics to solve. Segment tree: without modification (maximum, minimum, sum, maximum of sums); single assignment: sum, minimum, maximum, minimum segment where there are all numbers from 1 to K); single increment (sum); increment on segment directly, increment on segment lazy propagation (sum, minimum, maximum, number of positives, access to element); assignment on segment (sum, number of segments of ones). Fenwick tree. Trie. Bit trie. Search tree. Suffix array.
15. "Complex Dynamic programming" includes tasks, the solution of which requires knowledge of the relevant theory and the ability to come up with a way to apply it to the following subtopics: DP by bit masks, DP on a tree, Binary lifting, DP by profile, DP on strings, DP on number, DP on bit number, inclusion-exclusion principle.

The tasks in the "Accelerated Course 2013" and "Olympiads 9-11 grades" packages are supplemented with tasks from regional programming olympiads.

3.8 Weekly training olympiads for beginners

To control the quality of teaching by students and teachers, DL holds weekly training Olympiads for beginners. Each of them contains 20 problems. The first 10 are on the topic of "Introduction to programming"; the next 5 are on the topic of one-dimensional arrays; the next 5 are one each on the topics of "Two-dimensional array", "Geometry", "Strings", "Sorting", and "Text problem". The problems are available for solution during the week, since they open on Thursday morning and close on the following Wednesday evening. Regional Olympiads for students in grades 1-4 of previous years are used as training Olympiads (Dolinsky, 2022b).

3.9 Weekly training olympiads for professionals

To control the quality of teaching by students and teachers, DL holds weekly training Olympiads for professionals. Each of them contains 15 problems on the topics listed above, 1-15. The problems are available for solution during the week, as they open on Thursday morning and close on the following Wednesday evening. Regional Olympiads for students in grades 5-11 of previous years are used as training Olympiads (Dolinsky, 2024b).

3.10 Regional programming olympiads for primary and secondary schools

During the academic year, 5 regional Olympiads are held for students in grades 1-4 and grades 5-11 (two in the fall and three in the spring). To ensure continuity of education, since 2016, the last 5 problems of the Olympiad for grades 1-4 have been identical to problems 3-7 of the Olympiad for grades 5-11. The problems and tests for them for all Olympiads are compiled by the author based on the following considerations. Each academic year, the best-prepared schoolchildren participate in specialized classes to prepare for regional, republican, and international Olympiads. Within the framework of these classes, problems from international and national Olympiads are solved every Sunday from 9.00 to 14.00. Upon completion of the solution, a discussion of the solution to the problems is held. During this discussion, problems on topics/subtopics that are not yet in the "Olympiads 9-11" problem set emerge. Such problems are sent to the "task bank" for use in subsequent regional Olympiads for grades 5-11 on topics 8-15. And problems on topics 1-7 are made as preludes to problems on topics 8-15.

3.11 Specialized classes to prepare for regional, national, and international Olympiads

As already mentioned above, training Olympiads are held every Sunday, close to the real Olympiads. From September to the end of November, team Olympiad problems are solved in teams of 3 students per computer in order to prepare for the Open All-Russian Team Olympiad of Schoolchildren in Programming. From December to the end of March, personal Olympiad problems are solved to prepare for the Gomel Regional and the Belarusian Republican Olympiads of Schoolchildren in Informatics. From April to the end of August, team Olympiad problems are solved again. However, as in personal Olympiads, the problems are programmed by each student independently on their own computer. At the same time, as in team Olympiads, it is allowed to discuss solutions with other training participants. In addition, weekly on Wednesdays, problems are solved in classes. It is clear that the problems can also be solved at home at any time after the Olympiad.

3.12 Seasonal Cups and Person of the Year

To increase schoolchildren's motivation to do their homework, awards are given quarterly to students who have solved the most problems (main problems; problems from "I don't know" are not taken into account). Students of grades 1-4 – in the "Informatics" course

(the three best + the four best from Gomel). Students of grades 5-8 in the "Basic Programming" course (the three best). Students of grades 1-8 in the "Mathematics" course (the three best). In the "Programming – Professionals (teams)" course, awards are given to 3 teams of 3 students for the fall, 9 best students for the spring and summer, as well as 9 participants who scored the most points in the winter in the above-described Sunday personal Olympiads in specialized classes.

In addition, once a school year at the beginning of September, the winner of the "Person of the Year" competition will be awarded in the following nominations: "Informatics", "Basic Programming", "Mathematics" – one person each who has solved the most problems in the corresponding course over the past academic year.

It is interesting to note that the award ceremony is being held by a Belarusian company OpenMyGame, founded by graduates of the above-described educational system, former winners of the Belarusian Republican Olympiads in Informatics.

3.13 Tracking the dynamics of ratings on Codeforces

To increase motivation, a system for automatically updating the Codeforces ratings table of Gomel and Gomel region students (Gomel_Codeforces, 2025) has been developed within the DL system. Each student sees not only their rating but also their place in the hierarchy of schoolchildren in Gomel and the Gomel region. The first 25 in Gomel qualify for the Gomel Regional Olympiad in Informatics. The first 25 in the Gomel region qualify for diplomas of the Gomel Regional Olympiad. The first 15 in the Gomel region qualify for participation in the Belarusian Republican Olympiad in Informatics. The first 11 people are eligible to receive a diploma of the Belarusian Republican Olympiad in Informatics - from 2018-2025, every year, 11 to 14 representatives of the Gomel region received diplomas of the Belarusian Republican Olympiad in Informatics and automatically received the right to enter any university in Belarus without exams.

3.14 Participation in official Olympiads

In October, the Gomel qualifying Olympiad for the Open All-Russian Team Olympiad of Schoolchildren in Programming (VKOSHP) is held, after which several Gomel teams advance to the VKOSHP finals. In 2023 and 2024, Gomel teams won VKOSHP medals.

In December, the Gomel City Olympiad of schoolchildren in informatics is held; in January, the Gomel Regional Olympiad; in March, the Belarusian Republican Olympiad. In 2023-2025, representatives of Gomel (the author's pupils) were selected for the Belarusian team at the International Olympiad in Informatics.

4. Conclusion

This paper examines how to improve schoolchildren's ability to solve complex programming problems. The author classifies the problems of the International Olympiad in Infor-

matics finals as such. Contrary to the opinion of many schoolchildren and teachers that such skills cannot be developed through training but are largely determined by hereditary factors, the author took as allies Polya, Altshuller, and Vygotsky, who, in their works, sought to develop a methodology for the targeted development of thinking. In addition, the article presents the author's experience teaching programming to schoolchildren and preparing them for successful performance in programming Olympiads, including recommendations for solving problems during Olympiads, developing skills for solving complex problems during training, and using Codeforces for cognitive and psychological training. In addition, the author's programming-teaching system from grades 1 to 11 is based on the DL.GSU.BY distance learning tool developed under the author's supervision is described in detail.

The paper was written in spring 2025. But in summer 2025, in Bolivia, two of the authors' pupils won medals: silver and bronze. So now statistics have changed: From 1997 to 2025, 17 schoolchildren taught by the author (Dolinsky, 2024a) have won a total of 31 medals at the International Olympiad in Informatics. Of these, 7 are gold, 12 are silver, and 12 are bronze. At the same time, 2 schoolchildren won gold medals, 8 - silver medals, 7 - bronze. It is important to note that the last gold medal was won in 2012, the last silver medal in 2025, and the last bronze medal in 2025.

From the review: "What I missed in the paper: a more accurate description of the number of people involved in this project and their workload, which seems enormous if it has to be done regularly by one or even a few people."

It seems enormous, but it is so.

The author has been teaching computer science to schoolchildren since 1993. Since 1999, the author has been leading the development of DL.GSU.BY, being a teacher at Gomel State University named after Francisk Skorina. All software is developed by schoolchildren, students, and postgraduates. Problems for olympiads of grades 1-4 are prepared by informatics teachers at Gomel School 27 (ideas) and the author (implementations). Problems for olympiads for grades 5-11 until 2016 were prepared by students and postgraduates; since 2016, they have been prepared by the author. Problems for weekly olympiads are prepared by the author. Math problems is prepared by students.

Classes for students in grades 1-4 of Secondary School 27 are taught by teachers from Secondary School 27. Classes for students in grades 1-4 of other schools, as well as for students in grades 5-11 of Secondary School 27 and other schools in Gomel and the surrounding region, are held weekly by the author on Wednesdays and Sundays.

Also, since the DL site is accessible on the Internet, any student can study on their own and any teacher can teach their students on it. Now more than 100,000 persons from more than 100 countries are registered at DL.

References

Altshuller H. (1994). The Art of Inventing (And Suddenly the Inventor Appeared). Translated by Lev Shulyak. Technical Innovation Center, Worcester, MA, 1994. ISBN: 0-9640740-1-X.

Daniliuk A. (2018). How to read problem statements. https://codeforces.com/blog/entry/62730

Dolinsky, M. (2014). Technology for the development of thinking of preschool children and primary school children. Olympiads in Informatics, 8, 63–68. https://ioinformatics.org/journal/v8_2014_63_68.pdf

Dolinsky, M. (2022a). Instrumental System of Distance Learning DL.GSU.BY and Examples of its Application. Global Journal of Computer Science and Technology Interdisciplinary, 2022. 22(1):45-53. https://globaljournals.org/GJCST_Volume22/6-Instrumental-System-of-Distance-Learning.pdf

Dolinsky, M. (2022b). Primary School Programming Olympiads in Gomel Region (Belarus). Olympiads in Informatics, vol. 16, pp. 107-123, 2022. https://ioinformatics.org/journal/v16_2022_107_123.pdf

Dolinsky, M. (2024a). Teaching Programming to Schoolchildren in Gomel (Belarus). WSEAS Transactions on Advances in Engineering Education, vol. 21, pp. 11-16, 2024. https://wseas.com/journals/articles.php?id=9134

Dolinsky, M. (2024b). High School Programming Olympiads in Gomel Region. Olympiads in Informatics, vol. 18, pp. 155- 166, 2024. https://ioinformatics.org/journal/v18_2024_155_166_Dolinsky.pdf

Dolinsky, M., Dolinskaya, M. (2020). The Technology of Differentiated Instruction in Text Programming in Elementary School Based on the Website dl.gsu.by, Olympiads in Informatics, 14, 37-46. https://ioinformatics.org/journal/v14_2020_37_46.pdf

Gomel_Codeforces (2025). Codeforces Ratings of the Gomel region schoolchildren. https://dl.gsu.by/codeforces/

Holyinq A. (2012). Sports programming and cool ideas. (In Russ). https://codeforces.com/blog/entry/5990

IOI Syllabus (2025). https://ioinformatics.org/files/ioi-syllabus-2025.pdf

IOI vs CF performance: 2024 Edition (2024). https://codeforces.com/blog/entry/134654

IOI 2025 Teams CF ratings. https://codeforces.com/blog/entry/142475

Korotkevich, G. (2017). My strategy at AtCoder & CS Academy. https://codeforces.com/blog/entry/53457

Kulkov, A. (2016). General ideas for solving non-standard problems. (In Russ). https://codeforces.com/blog/entry/48417

Luo, Z. (2023). The Reason You are Bad at Codeforces — You are Not Russian Enough. https://codeforces.com/blog/entry/126310

Maksimovski V. (2020). Tips on Constructive Algorithms. https://codeforces.com/blog/entry/80317

Mirzayanov, M. (2015). How to Come Up with Solutions to Problems: Techniques. https://codeforces.com/blog/entry/20548

Mirzayanov, M., Pavlova, O., Mavrin, P., Melnikov, R., Plotnikov, A., Parfenov, V., Stankevich A. (2020). Codeforces as an Educational Platform for Learning Programming in Digitalization. Olympiads in Informatics, 2020, Vol. 14, 133–142. https://ioinformatics.org/journal/v14_2020_133_142.pdf

Polya, G. (1945). How to solve it. Princeton University Press, 148 p. https://math.hawaii.edu/home/pdf/putnam/PolyaHowToSolveIt.pdf

Ray, S. (2024). How to Solve Questions [Dominater Version]. https://codeforces.com/blog/entry/133289

Sahgal, S. (2022). How to improve my Constructive Algorithm Thinking? (In comments). https://codeforces.com/blog/entry/107782

Verhoeff, T. (2009). 20 Years of IOI Competition Tasks. Olympiads in Informatics, 2009, Vol. 3, 149–166. https://ioinformatics.org/journal/INFOL047.pdf

Vygotsky, L. S. (1997). Educational Psychology. CRC Press, 416 p. https://doi.org/10.4324/9780429273070

Michael Dolinsky is a lecturer in Gomel State University “Fr. Skoryna” from 1993. Since 1999 he is a leading developer of the educational site of the University (dl.gsu.by). Since 1997 he is heading preparation of the scholars in Gomel to participate in programming contests and Olympiad in informatics. He was a deputy leader of the team of Belarus for IOI’2006, IOI’2007, IOI’2008 and IOI’2009. His PhD is devoted to the tools for digital system design. His current research is in teaching Computer Science and Mathematics from early age.