

Team Competition in Mathematics and Informatics “Ugāle” – Finding New Task Types

Mārtiņš OPMANIS

*Institute of Mathematics and Computer Science, University of Latvia
29 Raina Boulevard, Riga, LV-1459, Latvia
e-mail: martins.opmanis@lumii.lv*

Abstract. Existing olympiads in mathematics and informatics are fixed form competitions for individuals with quite stable lists of task types. Outside the scope of these competitions falls a lot of interesting and challenging tasks like puzzles, games, logic tasks, and practical tasks outside the classroom. Team competitions offer a new dimension in a task solving process where successful collaboration between team members is one of basic requirements for achieving high results. This paper describes an annual (since 1996) Latvian team competition in mathematics and informatics for high-school students called “Ugāle”. Classification of the main task types is given and representatives of these task groups are given. Suitability of different task types in different contests is discussed. The evolution of the form and content of the competition is described.

Key words: olympiads in informatics, team competition, classification of competition tasks, grading.

1. Introduction

Most of scientific olympiads are organized on an individual basis. The most popular of them in the field under investigation are the International Olympiad in Informatics (IOI, 2009) and the International Mathematical Olympiad (IMO, 2009). Both olympiads have supporting infrastructure at regional (i.e., Baltic) and national (i.e., Latvian) level (NMS, 2009; LIO, 2009). Individual competitions (different contests and olympiads) have a long lasting history in Latvia. Annual state and open olympiads in mathematics with more than 3000 participants are organized by prof. Agnis Andžāns (Ramāna and Andžāns, 2002). Latvian olympiads in informatics have been organized since 1987, and since 1992 Latvian team participates on IOI. Latvia was one of the three founders of the annual Baltic Olympiad in Informatics at 1995 (BOI, 2004; BOI, 2008).

There are several well-known team competitions for high-school students in informatics. The most popular are the IPSC in Slovakia (IPSC, 2009) and the Open Team Cup in Russia (Open Team Cup, 2009). In mathematics the most important team event in Latvia is the “Baltic Way” competition for secondary school students in mathematics (Baltic Way, 2008).

At the IOI and IMO tasks are quite frozen in their form. Despite the theoretical possibility to use interactive and open input tasks, for the last two years (at IOI'2007 and

IOI 2008) only the so-called batch tasks were offered. At the IMO the use of computers, calculators or other electronic devices is prohibited, therefore it is practically impossible to discuss “changes to the competition format”. At the same time at the IOI there are no tasks which can be solved without computer.

It is quite clear that the mathematical and informatics competitions cover separate areas, leaving relatively big part of tasks uncovered. Some kinds of tasks do not fit into the IOI curricula (Verhoeff *et al.*, 2006), other tasks are deemed unsuitable for competition by general audience, for example, such kinds as data processing, numerical puzzles, logical games and cryptarithms. At usual informatics olympiads the so called “open ended problems” (Kemkes *et al.*, 2007) are not used. Different task types and their suitability for competitions as well as sources of inspiration are discussed by Burton and Hiron (2008), and Pankov and Orusulov (2007).

The idea of team events at competitions in informatics is discussed by Burton (2008).

2. Task types – Human Brain Versus Computer

If we look at tasks from the viewpoint of usability of a computer in their solving process, we can group these tasks in three main groups:

Gr1) Tasks which can be solved without usage of computer and where computer cannot give reasonable help to speed up the solution process. In the solution process pure mathematical skills are necessary and it is nearly impossible (at least for contestants) to use computer for essential help in the solution process. For example, the tasks “Prove that there is no largest prime number” or “Prove that among any five integers you can find three with sum divisible by 3” lie in this group.

Gr2) Tasks which can be solved only by use of computer. Besides native “write a program which solves a task for any input data” also a lot of “find all numbers with particular properties” tasks lie in this category (if exhaustive search is necessary and a relatively simple computer program can give the necessary results in reasonable amount of time).

Gr3) Tasks which may be solved either by help of a computer or without it.

The presence of a particular task at some competition assumes it belongs to one of these groups. It is obvious that tasks with completely unknown solution cannot be placed in any of these categories.

The so called “batch” tasks used at the IOI always belongs to Gr2. Moreover, the IOI rules prescribe usage of tasks with strictly specified programming languages and tools. Some years ago the so called “open input” tasks were included in the list of possible task kinds. As a rule, these tasks allow solving of some subtasks without computer or using different tools beyond the IOI toolset and therefore can be classified as Gr3 tasks. One of such tasks “Table” was suggested by the author and included in the programme of BOI 2003 (BOI, 2003), but the achieved results were lower than at the usual “batch” tasks – there were no contestants either at the on-site or offline competition who got full score.

The best result for several subtasks was achieved by different contestants. The last open input task included in the official programme of IOI was the task “Forbidden subgraph” at IOI’2006 (IOI, 2006). Output-only tasks are discussed by Vasiga *et al.* (2008).

At the IMO as well as at the World Sudoku Championships (WSC) and other competitions where human skills to solve particular tasks are examined, use of electronic devices is prohibited (IMO Regulations, 2009; Sudoku, 2009) and it may seem that here all tasks “by definition” lie in Gr1. This is not true at least for SUDOKU – a computer program able to solve classic SUDOKU puzzles is relatively simple (Brouwer, 2006). Thereby, allowing usage of computer will kill competition and make it completely uninteresting – hence banning electronic devices is the only possible solution. In other cases it is not always obvious which way of solution is more preferable. Many tasks in the World Puzzle Championships (WPC, 2009) are quite interesting to solve by computer programming or may be reformulated to become such.

There is also another pitfall – if usage of computers is allowed then there may be a tendency to look at all tasks as Gr2 tasks – not trying to think about problem more deeply without touching the keyboard. “When all you have is hammer, everything looks like nail”. Several nice examples of such tasks are given by Ginat (2008). If you have calculator at hand it is quite hard to press yourself to make mental calculations. On the other hand, if there is no calculator and you are asked to provide some simple calculations, you may try to find some electronic device which helps you to make these simple calculations instead of using your own brain. You somehow get obsessed with the idea “give me calculator and I will show you how to get the result”. All these thoughts can be transformed to a higher level if we replace calculator by computer.

3. History of “Ugāle” Competitions

Ugāle is town with 2677 inhabitants (2003) in the Western part of Latvia. In 1996 a teacher of Ugāle secondary school, Aivars Žogla, came up with idea of a joint competition in mathematics and informatics where together with classical mathematical problems a computer program for strategic game must also be written. The other main difference from classic olympiads was the idea of making this a team competition where up to three contestants solve tasks as one team.

Every team consists of three contestants and one (at the final round) or two (at the semi-finals) computers. During 5.5 hours teams must solve tasks (usually 8 to 10) given by a jury.

Competition format has changed during these years. The first two competitions (1996 and 1997) were organized as one-round competitions. Since 1998 a preliminary semi-final round is organized and the best teams together with a host team participate in the final round which usually takes place in May in the premises of Ugāle secondary school. The number of participating teams in the final round is fixed (12), but a number of teams in semi-finals vary year by year and in recent years was between 55 and 75 teams (see Fig. 1).

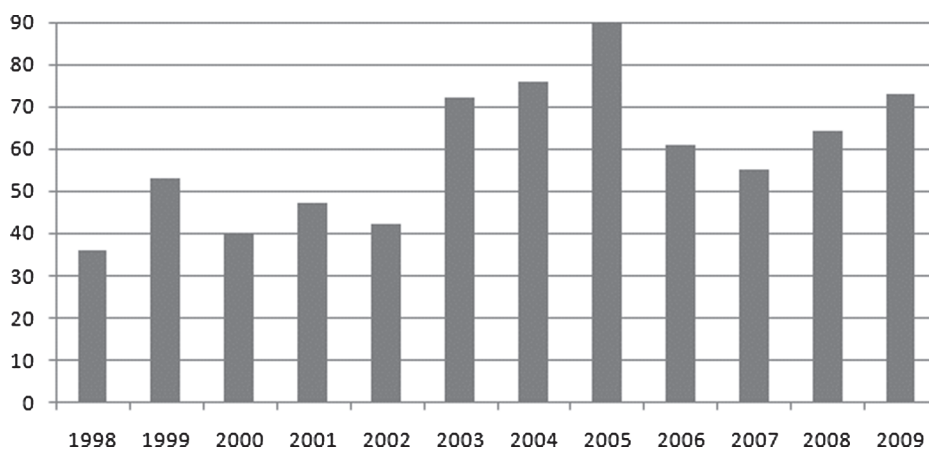


Fig. 1. Number of participants in semi-finals of Ugāle competition.

In the final round each team works in a separate classroom and there are no barriers for free communication between team members. However, they can also work as three individuals and during these years only some tasks forced cooperation of team members. During all the problem solving time, the captain of the team can come to the jury and ask questions concerning task descriptions and clarify technical details (like what must be submitted as solution, is it allowed to use some complementary thing not mentioned in task description, etc.).

4. Tasks and Grading

As in every scientific olympiad the main idea of “Ugāle” competitions is to give the opportunity to work on unusual tasks, still using basic knowledge obtained at school. However, the chosen competition format also influences the allowed kinds of tasks. Correctness of solutions at the final round must be checked by jury members during a limited amount of time (approx. two hours). Therefore, to lower the pressure on the jury, answers and solutions must be short in their form. It is quite hard to grade “classical” mathematical solutions like proofs in such a short amount of time and such tasks are not included in a competition’s task set. Also traditional IOI batch tasks are not included since there are a lot of pure programming competitions where such tasks are used.

However, the mathematical and informatics part of the competition is not lost. Simply, during the task selection process, tasks are chosen so that the answer to every of them is simple and short in form. The basic principle from math task solving: “Find all solutions and prove that there are no other” is kept alive, clearly stating tasks where it is enough to find any one particular solution and assuming finding all possible answers if such statement is omitted.

Mathematical tasks are formulated so that the answer is simple number, some sequence of numbers or set of possible answers. Finding the correct answer involves all

steps necessary for a full solution and from that viewpoint are of the same level of difficulty (there is no “shortcut” to the correct answer like guessing).

The real challenge is to prepare a task which could be solved either by using pure mathematic skills (without computer) as well as by writing a correct computer program which gives the answer in few competition hours.

In contrast to informatics olympiads, there is no constraint that written program must run in particular time (usually seconds or even its parts) and space limits during program execution. There are also no limitations on the software used during competition. Contestants may use the usual IOI programming languages as well as packages like Mathematica, MS Office, etc. A list of software, suitable for solving mathematical tasks is given by Turskiene (2002).

All teams are in the same conditions and therefore open-ended tasks of kind “find as good as possible a solution” are also acceptable. Such tasks are graded in manner that the team with highest achieved result are awarded maximum points, and other teams are awarded points proportionally to their score.

As a rule the full solution of any task can get maximum 100 points. If there are sub-tasks, then it is clearly stated how many points are given for the solution of a particular subtask. In early competitions there were attempts to differentiate points given per task, but such praxis (as well as at the IOI) was not continued. Therefore, the determination of difficulty level of a particular task is one of the essential competition components.

All answers and results must be written by teams on a special form (“answers sheet”) and as a rule this is the only source for grading (another source may be computer program or some result file in electronic form on some media).

5. Task Categories

The first competitions in 1996 and 1997 now can be considered as warm-up competitions. The first competition task set contained nine pure mathematics tasks offered by prof. A. Andžāns and one programmable game task. The second competition contained too large a number (19) of tasks and lot of effort was wasted, because it was impossible to solve a reasonable amount of tasks in the given time. In recent years at the final round 10 tasks are offered and the obtained results show that such a number is reasonable.

In total 244 tasks were used during years from 1996 till 2009 (at the moment of writing of this paper only the semi-finals of 2009 have finished). Classification of tasks was done by including every task in one or more categories.

It can be seen that some groups of tasks are quite stable, because every year (or even more – in every competition) at least one task from a particular group is included in the task set. Other groups are disappearing or appearing. For example, the currently existing format excludes tasks with mathematical proofs or argumentation, which was regular part of earlier competitions.

In the following chapters main task groups are described. The total number of tasks is given in brackets.

5.1. Data Processing/Mining Tasks (24)

One task of this kind has been included in every task set starting from 1997. The main idea of such tasks is to get some numerical results from given data (usually as one or two text files). These tasks assume the usage of spreadsheet processing systems (like Microsoft “Excel”) or DBMS (like Microsoft “Access”), however solutions may be found with other tools (like programming in the languages from the IOI list). An example of such a task is “Football” from semi-final of Ugāle’2004:

In the text file futbols.txt there is information about all games of a football championship. During the championship two rounds of matches are played – each team plays two games with each other – one at home and second as visitor. For a win a team is awarded 3, and for loss – 0 points. If game ends in draw, each team gets 1 point. Each file row contains information about one game in the following format:

Host team name – Visitor’s team name; Goals scored by host team; Goals scored by visitor’s team; (further names of scoring players as well as minute of match when goal was scored are given. At the beginning all the host team and then all the visitor team players are listed).

For example, one row can look like:

Badgers - Monkeys; 1; 3; Apse; 88; Lipenbergs; 37;
Priede; 8; Millersons; 55

Your task is to find answers to the following questions:

1. *How many teams participated in championship?*
2. *How many goals were scored by visitor teams?*
3. *How many goals were scored in total?*
4. *How many matches ended in draw?*
5. *How many goals were scored by Smits?*
6. *In which game was the maximum number of goals was scored?*
7. *How much goals were scored in the first halves of matches (till the 45th minute, inclusive)?*
8. *Who scored the maximum number of goals? How many? In which team he plays?*
9. *How many players scored just once?*
10. *How many different draw results were in all championship matches?*
11. *Which team won the championship and how many points did it obtained?*
12. *Which team got last place and how many points did it obtained?*

5.2. Cryptarithms (15)

A usual cryptarithm is a well-known type of puzzles. The solution is a correct arithmetical expression. In the task digits are substituted by letters or different symbols and it is known that equivalent digits are substituted by the same letter and different letters covers different digits. Several modifications of the classical format also are used.

5.4. Geometry (36)

The next four sections describe task types that have become classic at mathematical olympiads. Algebra, Geometry, Combinatorics and Number theory are four “whales” of mathematical competitions (Andžāns and Ramāna, 2002). For example, the task set of the mathematical team competition “Baltic Way” contains exactly five tasks from each of these four groups. Despite their apparent solidness, A. Toom characterizes classical geometry (together as word problems) as the “Cinderella of American education” (Toom, 2005).

Geometrical tasks are especially suitable for development of abstract thinking and demand different knowledge from other branches of mathematics. Geometry has a lot of faces and every task set must offer at least one geometrical task. To give a bit wider insight, instead of one representative task, several task examples will be given.

In the first competitions, classic olympiad tasks were used, but in recent years they have slightly changed to an expected form of result which usually is some numerical value. Task “Two triangles” (semi-final of Ugāle’2008) is one of the representatives:

The intersection of two triangles is hexagon with inner angles (in this order): 87° , 141° , 105° , 137° , 104° and 146° . Calculate the angles of these triangles!

If you remember the general remark concerning multiple solutions, this is exactly the case, because this task has more than one solution and in the task description there is nothing allowing taking for granted that it is enough to present just one of them. Looking for suitable tasks, the content of tasks from olympiads in mathematics are also taken in account. Obviously, the task types at olympiads are not a solid matter – they changes over time (see Fig. 3).

Construction tasks have not been presented at the Latvian Olympiad in Mathematics for the last three decades. So it was good reason to use such tasks at Ugāle competition. Task “Elegant pentagon” (final of Ugāle’2004) is one of them:

Let’s say that a convex pentagon is “elegant” if the following conditions are satisfied:

- 1) *it can be inscribed in circle,*
- 2) *the length of all sides and radius of the circumscribed circle can be expressed in whole centimetres,*
- 3) *all sides and radius of the circumscribed circle are of different length.*

Let’s say that a convex pentagon is “partly elegant” if only the first two conditions are satisfied. Your task is to construct either an elegant (100 points) or partly elegant (30 points) pentagon.

In the middle of the 20th century in the schools of Latvia there were quite popular tasks concerning measuring distances in nature. At that time such tasks were included in secondary school curricula. Tasks of this type may be mention the measurement of height of a particular tree, width of a river, distance to a far object, etc. In the author’s

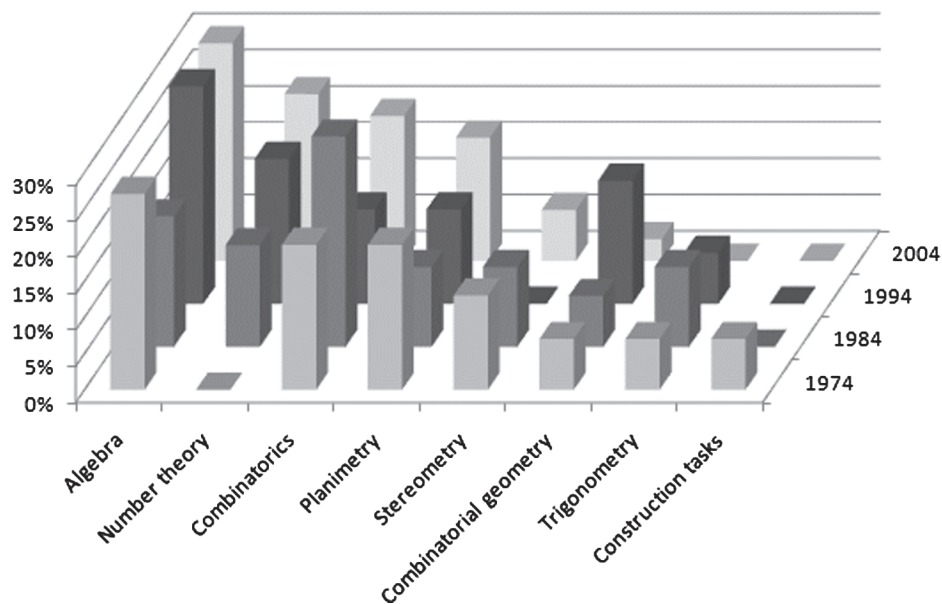


Fig. 3. Task types at Latvian Mathematical Olympiads (Bonka, 2004).

opinion tasks of this kind are very close to the nature of geometry (the word itself means measuring the earth) and are undeservedly forgotten. An attempt to check these skills in the contemporary youth was done by including the task “Distance in nature” in the final round of 2005:

You are given a rope with length equivalent to the width of the Ugāle secondary school front door).

At both sides of school building there is one marked lamppost (see Fig. 4). Calculate the straight distance in ropes between these two lampposts without the destruction of the school building!

Only those solutions where the difference between jury’s and submitted solution will be less than doubled length of rope will be graded. If you wish to suggest the use of other tools beside rope, consult the jury in advance!

This task became quite popular at this competition and teams showed good results – all teams got points and 4 out of 12 teams got a full score. A similar task was included in the task set two years later.

One more (and again completely different) geometrical task is the task “Pencil” at the final of Ugāle’2003:

“A pencil, the cross-section of which is a regular hexagon with side length equal to 0.5cm, was sharpened by a cone-shaped sharpener with the angle between the generatrix and the altitude equal to $\pi/8$ in a way that the length of the pencil didn’t change. How much of the pencil (in cubic centimetres) was removed? Provide as an answer a decimal fractional number, the more precise the better. Maximum points will be awarded for correct 9 digits after the decimal point.”

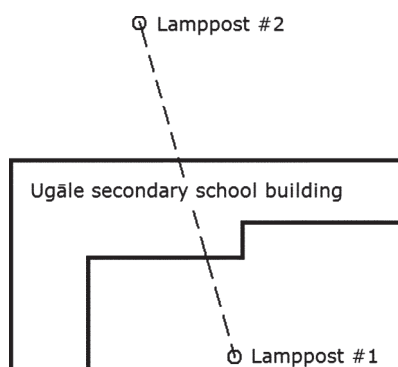


Fig. 4. Configuration of school building and lampposts.



Fig. 5. Sharpened hexagon pencil.

The small waves at the end of the sharpened part (see Fig. 5) make calculations non-trivial and the task – interesting. The number of decimal digits necessary for full points does not allow neglecting this area.

5.5. Number Theory (31)

These tasks are appropriate for Ugāle competitions due to their natural relationship between mathematics and programming – tasks can be solved in various ways, but the best results takes some combination of skills from both disciplines.

Two examples from this task group will be given.

The task “Twenty-digit number” (final of Ugāle’2001, idea from (Puzzles, 2000)):

Arrange two of each of the digits 0 to 9 to form a 20-digit number. Your number may not begin with a zero.

You are then scored on your number as follows:

For every n ($n > 1$) consecutive digits where the first digit is not 0 that form a square number, you score n points.

For example, if your number was 98543676011023475928, you will score two points for 36 and three points for 676 – for a total of 5 points. You may not count 01 as a two-digit square.

What is the maximum number of points you can score?

This task can be easily turned into an open input task by specifying different sets of usable digits. However, the essence of the task remains the same – contestants are in the

same starting positions and still must compete for a better result where the possibility of getting the perfect result during the limited competition time is unclear.

The task “Ugāle Primes” (semi-final of 2005):

Let's name an n -digit prime $\overline{p_1p_2\dots p_n}$ as a Ugāle's prime if and only if all prefixes of this number $\overline{p_1}, \overline{p_1p_2}, \overline{p_1p_2p_3}, \dots, \overline{p_1p_2\dots p_{n-1}}$ also are primes. For example, 71 and 311 are Ugāle primes (because 71, 7, 311, 31 and 3 are primes), but 27, 43 and 307 – are not (27, 4 and 30 are not primes). Find one, as big as possible Ugāle's prime!

5.6. Combinatorics (40)

As well as the task from the previous section, combinatorial tasks are solvable in various ways. As an example task “APRICOT” (final of Ugāle' 1998) is given:

A word consisting of n different letters is given. This word is written on a sheet of paper with squares so that every letter is written in its own square without empty squares in-between. The same word is written in the row below in the same manner just starting writing one square to the left. So this word is written on the next lines until it is written $n + 1$ time.

In the Fig. 6 there is given an example for $n = 7$.

Your task is to calculate the number of different ways the given word can be read on the sheet if you can start from the first square of any row and go to a neighbour square to the right or down and direction of reading may be changed no more than twice. One of the ways to read the word is denoted in the figure by darker squares (direction of reading is changed once).

Solve this task for a) $n = 7$, b) $n = 36$, c) $n = 711$, d) $n = 1492$.

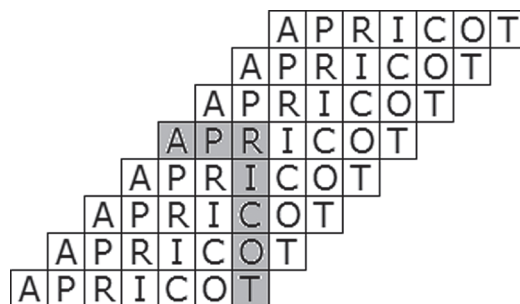


Fig. 6. Example for $n = 7$.

5.7. Algebra (14)

Classic tasks for contestants of mathematical olympiads. Task “System” (semi-final of Ugāle’2001):

Find a solution in integers:

$$\begin{cases} z^x = y^{2x}, \\ 2^z = 4^x, \\ x + y + z = 20. \end{cases}$$

5.8. Logic (37)

There are a lot of classic reasoning puzzles, where some clues are given and you must find out hidden consequences. One of the famous representatives of this kind is puzzle “Who Owns the Zebra?” (Zebra, 2009). But such tasks are not represented at usual olympiads. However, in some countries such tasks are used in theoretical or preliminary rounds of olympiads in informatics (Anido and Menderico, 2007).

At the end of the 90s a very interesting task “Self-referential aptitude test” by Propp (2009) was published and became a source of inspiration for several tasks.

Interestingly, solutions to such tasks can also be found by quite simple computer programs (however, these programs are quite unusual) and, therefore, such tasks also lie on the edge between mathematics and informatics.

Task “Concrete logic” (final of Ugāle’2002):

Answer the 20 questions below by “yes” or “no” so that all the answers do not contradict each other.

1. *Are the answers to Questions 6 and 7 equivalent?*
2. *Is “no” the answer to Question 1?*
3. *Are the answers to Questions 4 and 20 different?*
4. *Are the answers to Questions 3 and 20 different?*
5. *Are this and the answer to Question 19 different?*
6. *Is “yes” the answer to Question 2?*
7. *Is “yes” the answer to Question 15?*
8. *Are the answers to Questions 11 and 19 equivalent?*
9. *Is “yes” the answer to Question 10?*
10. *Is “no” the answer to Question 13?*
11. *Is it true that Mr. Bērziņš doesn’t like strawberries?*
12. *Is “yes” the answer to Question 16?*
13. *Is “yes” the answer to Question 12?*
14. *Are this and the answer to Question 11 equivalent?*
15. *Is it true that “no” is the answer to at least half of all the questions?*
16. *Is it true that “yes” is the answer to at least half of all the questions?*
17. *Are the answers to Questions 9 and 4 equivalent?*
18. *Is “yes” the answer to Question 7?*

19. Is it true that the name of Mr. Bērziņš is Jānis?
 20. Are the answers to Questions 3 and 4 different?

5.9. Analysis of Algorithms (10)

Understanding programs written by other authors also is part of programmer's everyday job, but such tasks are not presented at competitions in "pure" form when fragment of code or pseudo code is given. Task is in finding something in or related to the given code: finding error, constructing the worst case counterexample, estimation of overall complexity, implementing the same algorithm in a more effective way. In some countries analysis of algorithms is part of the preparation work for the IOI (Forišek, 2007). An example of task where analysis of a given program must be provided is task "Sorting" (final of Ugāle'2003, author Aivars Žogla):

In the file SORT.PAS an algorithm is given that sorts the elements of number array $A[0..n-1]$ from position low till position $high$ in non- decreasing order. By taking $low = 0$ and $high = n - 1$, the entire array will be sorted.

Your task is to find an array containing each of the numbers from 1 to 20 exactly once, for which sorting by calling procedure `sort(A, 0, 19)`, uses the maximum number of array element comparison operations (these rows are marked by `{}`). For example, sorting the array $A=\{3,4,1,2,5\}$, uses 7 comparison operations.*

```
{ ===== Start of SORT.PAS ===== }
const MAXN = 100;

type TArray = array[0..MAXN] of integer;

procedure swap(var A:TArray; i, j:integer);
var temp:integer;
begin
  temp := A[i]; A[i] := A[j]; A[j] := temp;
end;

procedure sort(var A:TArray; low, high: integer);
var i,j,middle:integer;
begin
  if high - low < 5 then
    for i := low + 1 to high do
      begin
        j := i;
        while j > low do
          if A[j - 1] > A[j] then
            {*}
              begin
                swap(A, j - 1, j);
                j := j - 1;
              end
            else j := low;
          end
        end
      end
  end
```

```

else
begin
    middle := (low + high) div 2;
{*}    if A[middle] < A[low] then swap(A, low, middle);
{*}    if A[high] < A[low] then swap(A, low, high);
{*}    if A[high] < A[middle] then swap(A, middle, high);
    swap(A, middle, high - 1);
    i := low;
    j := high - 1;
    repeat
        repeat
            i := i + 1;
{*}        until A[i] >= A[high - 1];
        repeat
            j := j - 1;
{*}        until A[j] <= A[high - 1];
            if i < j then swap(A, i, j);
        until i >= j;
        swap(A, i, high - 1);
        sort(A, low, i - 1);
        sort(A, i + 1, high);
    end;
end;

begin
end.
{ ===== End of SORT.PAS =====}

```

Another task of this kind is “*Function of functions*” (final of Ugāle’2005):

Function $f(x)$ is defined for all integers from 1 to 2000000000 and function values are positive integers. Function values are known for one hundred argument values (Table 1).

Your task is to write as short as possible program in one of the programming languages Pascal, C or C++, which implements this function for all x values given in the table above. Do not worry about the values other than given, because your program will be tested only with the argument values given in the table.

Function source code must be presented in one separate file and usage of other data files is prohibited. Program must read one x value from the standard input and the corresponding $f(x)$ value must be written to the screen and program must exit without waiting for additional user input (such as pressing a key). Program may use only the modules and libraries included in compiler’s standard configuration.

If for any of the hundred given argument values the answer will be different from the value given in the table, the score for the task will be 0 points. Execution time for one particular test case must not exceed one second.

Programs with lower amount of source code in bytes will get higher scores.

Table 1

x	$f(x)$	x	$f(x)$	x	$f(x)$
1	2	3071	83	987654329	4312901
2	1	3073	439	987654331	9588877
3	2	3381	1127	987654791	31397
5	2	3383	199	987654793	31397
7	2	3385	677	987654803	31397
9	3	3403	83	987654809	83227
10	5	3419	263	987654821	31397
11	3	3493	499	987654857	141093551
13	3	173005	34601	987654861	329218287
15	5	173007	57669	987654865	197530973
16	8	173009	10177	987654881	57559
17	3	173021	409	987654883	31397
21	7	173027	2437	987654901	31397
22	11	173029	7523	987654971	48341
23	3	173049	57683	987688883	6059441
33	11	173051	1321	987688885	197537777
67	7	173419	4687	1999999001	285714143
77	11	173973	57991	1999999003	44711
105	35	173975	34795	1999999005	666666335
107	7	173983	9157	1999999009	32786869
109	7	173989	677	1999999015	399999803
111	37	9173003	1310429	1999999019	155171
115	23	9173005	1834601	1999999027	153846079
117	39	9173009	23581	1999999037	42553171
119	17	9173011	3023	1999999039	4640369
123	41	9174653	5639	1999999957	7782101
125	25	9174661	20899	1999999961	54054053
126	63	9174667	295957	1999999967	285714281
131	11	9174671	834061	1999999969	181818179
137	11	9174673	6323	1999999975	399999995
499	19	987654321	329218107	1999999981	285714283
3055	611	987654325	197530865	1999999997	37735849
3059	437	987654327	329218109	1999999999	64516129
3061	53				

Contestants were also supplied by table values in a separate text file.

The simplest approach would be trying to code the given values without any investigation. However, much better result could be obtained, by discovering some regularity.

Besides that, for choosing the right approach, a contestant's deep understanding of the possibilities of different languages and compilers was a great advantage. It is quite easy to understand the usability of such tasks in industry – in a world of microprocessors a requirement to fit in a given amount of memory is usual.

5.10. Programming (25)

One of the programming tasks different from the usual IOI tasks is task “*Smart program*” (final of Ugāle’1998):

Write a program in Pascal, C or BASIC, which

- *outputs on the monitor the number 1998 only once;*
- *if the program code is modified by replacing one symbol by another in such a way that program still compiles and executes, it still only once outputs the number 1998.*

This task is quite tricky and needs an extremely deep knowledge of the chosen programming language. In some sense this task is designed for hackers not algorithmists. The essence of the task is different from the usual programming tasks (who care about your source code and tries to break it by changing it in any other competition?) and every single space character in code can be used against you.

Despite the short codes submitted, grading was done in a special way by the best possible jury members – three IOI medallists (Krišs Boitmanis (silver on IOI’97, bronze on IOI’96), Renārs Gailis (silver on IOI’97) and Juris Kriķis (bronze on IOI’96)). Every submitted program was investigated by looking through carefully and trying to break the code. Because there were only 12 teams, this evaluation was successfully completed and only those programs where experts did not find any faults got full score. Technically possible, but hard to imagine would be the testing of this task without such a group of experts, because “grading” includes the possibility or impossibility of finding a counterexample (a symbol which can be changed to show that the second condition is not satisfied).

One of programs in PASCAL which twelve years after the actual competition *seems to be* a correct solution:

```
var sk1,sk2:string;
Begin
sk1:='1998';
sk2:='1998';
if sk1<>sk2 then writeln('1998') else writeln(sk1);
End.
```

5.11. Dominoes (12)

The author has noticed that there is quite a big gap between the contestants’ generation and the author’s one in the sense of basic knowledge in the field of logical or board games. It is nearly impossible to say which rules of logical games are known to the general audience and which must be explained in task formulations. It is not easy to say whether all contestants are familiar with the rules of chess or checkers or not. In this situation classical dominoes are widely used in the competition every year and a set of dominoes is necessary attribute for every team.

Dominoes are quite simple and at the same time they possess a quite high combinatorial power. The following task “Area of dominoes” is the only one which was included

in a task set twice – at the final of Ugāle’2005 it was not solved and was included in the task set of the next year’s semi-finals where it was solved by only one team out of 61:

All 28 pieces of the usual dominoes set must be placed in the area shown in Fig. 7 so, that:

- *every piece covers exactly two squares,*
- *if two pieces share common edge, then in both halves the number of points must be the same,*
- *in all four horizontal rows (indicated by arrows) the total number of points must be the same.*

It is enough to show one such distribution.

Task “Who can find more?” (final of Ugāle’1997):

*Fill a rectangle consisting of 8×7 squares with the numbers from 0 to 6 (in each square there must be one number) so that this rectangle can be covered by pieces of one set of dominoes (every square is covered by half of one dominoe, all squares are covered) in **as many ways as possible**.*

After the competition this task was published on the website of the Latvian Olympiad in Informatics (More, 2009) and eight years after competition the solution (see Fig. 8) with **793648** different coverage was found by former IOI medallist Jānis Sermuliņš (gold on IOI’97 and IOI’99, bronze on IOI’98):

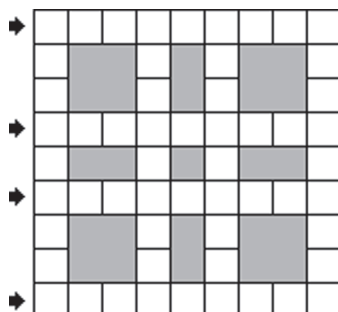


Fig. 7. Area of dominoes.

```

1 3 1 2 2 2 3 4
4 1 1 1 2 3 4 2
4 4 0 1 0 2 6 4
4 0 0 0 6 0 2 6
5 4 0 5 0 3 6 1
1 5 5 5 3 6 5 6
5 2 5 3 3 3 6 6

```

Fig. 8. Best known solution of the task “Who can find more?”








This task allows a lot of possible modifications.

- The same task but coverage must be unique.
- The same task but coverage must be not unique.
- Reverse task: Some distribution of numbers can be given and the task is to count the number of different coverages by the domino set.

It is quite clear that the original formulation and the reverse task needs to be solved by a computer program.


The two simple modifications can also be solved without computer. However, a computer program, if available, can give great help in checking generated solution candidates.

Dominoes can also be used in other ways. Let's say that some positive integer n is *representable* by dominoes if it is possible to make a row of dominoes from one set and the sequence of points in the domino halves in this row corresponds to digits in the number n (one leading empty half is allowed if number of digits in n is odd). A row of such domino pieces will be named a “representation of n ”.

So, a representation of 1205 is     and a representation of 105 –   .

It is clear that it is not possible to represent numbers with digits greater than 6 as well as numbers 453354 and 606, because the corresponding row of pieces cannot be built from one set of dominoes.

Using the above definition one can also define the simultaneous representation of several numbers – if all numbers can be represented by pieces of one set (there are no pieces necessary for representation of several numbers). For example, simultaneously representable are the numbers 122461, 33440001 and 224611.

Contrary, the numbers 134 and 3310 are not representable simultaneously, because both representations need the piece .

These definitions were necessary to formulate the task “Number and its power” (final of Ugāle’1997):

Find an integer n which is simultaneously representable together with some of its powers (n^2, n^3, \dots) and the number of domino pieces used in these two representations is as big as possible.

After the competition this task also was published on the website and six years after competition four solutions with 26 used domino pieces (Domino, 2009) were found by the former participant of IOI’98 and IOI’99 Dmitrijs Rutko. One of these solutions is as follows: 51531210022106635 and its square 2655465606342463301645403311023225. However, it is not proved that 26 pieces is the absolute maximum.

5.12. Other Types of Tasks

Tasks come into fashion and go out of it. For example, SUDOKU tasks were quite popular for several years. As it is mentioned above, the simple SUDOKU task is not a hard task for a good programmer. However, the SUDOKU theme can be used in tasks, exploiting popularity of these tasks which allows us to expect that the rules are well-known and teams will be encouraged to solve such tasks.

Task “Reverse SUDOKU” (final of Ugāle’2006):

There is given solution of the classic 9×9 SUDOKU (at competition one particular SUDOKU solution was given – M.O.). Clear as many squares as you can so that remaining digits (which looks like classic SUDOKU problem) still give this unique solution.

Grading: Only solutions with fewer than 51 digit in the table will be graded. Solutions with fewer digits in the table will score more points. If the SUDOKU solution will be not unique, points will not be given.

There is no well-known algorithm for this task solving. At the same time it is known that theoretical limit is about 17 remaining digits (Sudoku, 2009).

Of course, the categories described in previous sections do not cover all tasks used in Ugāle competitions. Some of them are very specific and there is not an obvious category where to include such tasks. Several tasks with special software provided are not included due to the length of description and many technical details. A similar task type is described by Ribeiro and Guerreiro (2007). Among tasks not presented in this paper are black box testing tasks, tasks where some erroneous proof is given and the error must be found, tasks tightly coupled with physics and where cryptography must be mentioned.

Till now “Ugāle” competitions are competitions for Latvian students, so the main website and all available materials are in Latvian only (Ugāle, 2009). However, English speaking students can try to solve task set of the final of Ugāle’2003 (unofficial translation of tasks is done by I. Stepanovs) (Ugāle, 2003).

6. Conclusions

During its 14 years the idea of the Ugāle competition has shown its vitality. Definitely there is space for competitions with tasks different from the well-known olympiads. A lot of creative tasks was tested in such an experimental environment as Ugāle competition is. Unusual tasks with unusual grading schemas are challenging for contestants as well as for authors of tasks. The work of jury is not simple, because every new task type needs careful investigation and different solutions with or without a computer, as well as grading schemas, must be checked carefully.

Acknowledgements. I would like to thank prof. Kārlis Podnieks for his valuable comments.

References

- Andžāns, A. and Ramāna, L. (2002). Latvian–Icelandic project LAIMA. In A. Andžāns and H. Meissner (Eds.), *Creativity in Mathematics Education and the Education of the Gifted Students: Proceedings of the International Conference*. Riga, University of Latvia, pp. 13–14.
- Anido, R.O. and Menderico, R.M. (2007). Brazilian olympiad in informatics. *Olympiads in Informatics*, **1**, 12.
- Baltic Way (2008). *Baltic Way Mathematical Team Competition 2008*. Gdansk, Poland.
<http://www.balticway08.math.univ.gda.pl/?p=history>

- BOI (2003). Task “Table”. *9th Baltic Olympiad in Informatics*, Tartu, 2003.
<http://www.ut.ee/boi/xxx/TABLE.eng.pdf>
- BOI (2004). *The 10th Baltic Olympiad in Informatics*, Ventspils, 2004. <http://www.boi2004.lv/>
- BOI (2008). *The 14th Baltic Olympiad in Informatics*, Gdyna, 2008. <http://b08.oi.edu.pl/>
- Bonka, D. (2004). IKT ietekme uz matemātikas padziļinātas izglītības sistēmu Latvijā (Influence of ICT on Advanced Math Education System in Latvia). In *Starptautiskās konferences LatSTE'2004 rakstu krājums*, Rīga, LU, p. 89.
- Brouwer, A.E. (2006). Sudoku puzzles and how to solve them. *Nieuw Archief voor Wiskunde*, **57**(4), 258–259.
<http://www.math.leidenuniv.nl/~naw/serie5/dee107/dec2006/brouwer.pdf>
- Burton, B.A. (2008). Breaking the routine: events to complement informatics olympiad training. *Olympiads in Informatics*, **2**, 11–12.
- Burton, B.A. and Hiron, M. (2008). Creating informatics olympiad tasks: exploring the black art. *Olympiads in Informatics*, **2**, 26–33.
- Domino (2009). *The best known known solution of task “Domino numbers”*.
<http://vip.latnet.lv/ljo/Neatdz/dom2lab.htm>
- Forišek, M. (2007). Slovak IOI 2007 team selection and preparation. *Olympiads in Informatics*, **1**, 59.
- Ginat, D. (2008). The unfortunate novice theme of direct transformation. *Informatics in Education*, **7**(2), 173–180.
- IMO (2009). *International Mathematical Olympiad*.
<http://www.imo-official.org/>
- IMO Regulations (2009). *Regulations for an International Mathematical Olympiad*, Item D4.
<http://olympiads.win.tue.nl/imo/imoregul.html>
- IOI (2006). *IOI'2006 Task “Forbidden subgraph”*. <http://www.ioinformatics.org/locations/ioi06/contest/day1/forbidden/forbidden.pdf>
- IOI (2009). *International Olympiad in Informatics*.
<http://www.ioinformatics.org>
- IPSC (2009). *Internet Problem Solving Contest*. <http://ipsc.ksp.sk/>
- Kemkes, G., Cormack, G., Munro, I. and Vasiga, T. (2007). New task types at the Canadian computing competition. *Olympiads in Informatics*, **1**, 81.
- LJO (2009). *Latvian Olympiads in Informatics* (in Latvian). <http://www.ljo.lv>
- More (2009). *The best known solution of task “Who can find more?”*.
<http://vip.latnet.lv/ljo/Neatdz/domlab.htm>
- NMS (2009). A. Liepas Neklātieņu matemātikas skola (A. Liepas Correspondence mathematical school, in Latvian).
<http://nms.lu.lv/>
- Open Team Cup (2009). *Командный Открытый Кубок по программированию* (Open Team Cup in programming). <http://shade.msu.ru/~ejudge/>
- Propp, J. (2009). *Self-Referential Aptitude Test*.
<http://www.cs.berkeley.edu/~lorch/personal/self-ref.html>
- Pankov, P.S. and Orusulov, T.R. (2007). Tasks at Kyrgyzstani olympiads in informatics: Experience and proposals. *Olympiads in Informatics*, **1**, 132.
- Puzzles (2000). “20-Digit Challenge”, newsgroup “rec.puzzles”.
 Posted April 13, 2000 at <http://groups.google.com>
- Ramāna, L. and Andžāns, A. (2002). Advanced mathematical education in Latvia. In A. Andžāns and H. Meissner (Eds.), *Creativity in Mathematics Education and the Education of the Gifted Students: Proceedings of the International Conference*. University of Latvia, Riga, p. 6.
- Ribeiro, P. and Guerreiro, P. (2007). Increasing the appeal of programming contests with tasks involving graphical user interfaces and computer graphics. *Olympiads in Informatics*, **1**, 149.
- Sudoku (2009). *Rules 4th World SUDOKU Championship*.
http://www.szhk.sk/images/stories/dokumenty/pravidla_sutaze.doc
- Toom, A. (2005). *Word Problems in Russia and America*, pp. 4–9.
<http://www.de.ufpe.br/~toom/travel/sweden05/WP.PDF>

- Turskiene, S. (2002). Computer technology and teaching mathematics in secondary schools. *Informatics in Education*, **1**, 150.
- Ugāle (2003). *Tasks and Complementary Files of "Ugāle'2003" final*.
http://vip.latnet.lv/lio/lietas/u03_final_english.zip
- Ugāle (2009). *Team Competition in Mathematics and Informatics "Ugāle"* (in Latvian).
<http://www.uvsk.lv/p113.htm>
- Vasiga, T., Cormack, G. and Kemkes, G. (2008). What Do Olympiad Tasks Measure? *Olympiads in Informatics*, **2**, 184.
- Verhoeff, T., Horvath, G., Diks, K. and Cormack, G. (2006). A proposal for an IOI Syllabus. *Teaching Mathematics and Computer Science*, **4**(1), 193–216.
<http://ioinformatics.org/admin/isc/iscdocuments/tmcs-2006-i-verhoeff-horvath-diks-cormack.pdf>
- WPC (2009). *World Puzzle Championship*.
<http://www.worldpuzzle.org/championships/index.html>
- Zebra (2009). *Task "Who Owns the Zebra?"*
http://orion.math.iastate.edu/burkardt/puzzles/zebra_puzzle.html



M. Opmanis is researcher at the Institute of Mathematics and Computer Science of University of Latvia. He is deputy team leader of Latvian IOI team since 1996 and was team leader of Latvian team at Baltic Olympiads in Informatics since 1995 till 2007. M. Opmanis was head of jury of Baltic Olympiad in Informatics at BOI'1996, 1999 and 2004.