УДК 519.688

1.2.2. Математическое моделирование, численные методы и комплексы программ (Физико-математические науки)

О РАЗРАБОТКЕ КОМПЛЕКТА КОМПЬЮТЕРНЫХ ПРОГРАММ ДЛЯ АВТОМАТИЗИРОВАННОЙ ПОДГОТОВКИ ПРАКТИЧЕСКИХ ЗАДАНИЙ ПО РАЗДЕЛУ «ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ» ДИСЦИПЛИНЫ ВЫСШАЯ МАТЕМАТИКА И ПРОВЕРКИ ВЫПОЛНЕННЫХ ОБУЧАЮЩИМИСЯ РАБОТ

Лаптев Сергей Владимирович
к.ф.-м.н., доцент
ФГБОУ «Кубанский государственный аграрный университет», 350044, Россия, г. Краснодар, ул. Калинина 13, E-mail: mail@kubsau.ru

Михайленко Евгений Владимирович
к.ф.-м.н., доцент, заместитель начальника кафедры
ФГКОУ ВПО «Краснодарский университет МВД России»,350005, Россия, г. Краснодар, ул. Ярославская, 128, E-mail: post@krdu-mvd.ru

В статье представлен созданный авторами новый программный продукт для автоматизированной подготовки практических заданий по разделу «Задачи линейного программирования» дисциплины высшая математика и проверки выполненных обучающимися работ. Автор раскрывает предпосылки, описывает основные методы и технологии, применяемые при разработке комплекта программ. В ней изложены основные принципы разработки представляемых программных продуктов, описаны структура типовых программных разработок и содержание входящих в них программных модулей, исследуются алгоритмы генерирования различных классов задач по разделам высшей математики: Описываемые приложения иллюстрируются рисунками и фрагментами кодов описываемых программных приложений. Для профессорско-преподавательского состава, инженеров-программистов образовательных организаций

Ключевые слова: АВТОМАТИЗАЦИЯ, АЛГОРИТМ, ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ, МАКРОС, СИМПЛЕКСНЫЙ МЕТОД, ПРОГРАММНЫЙ МОДУЛЬ

http://dx.doi.org/10.21515/1990-4665-203-039 ‘

UDC 519.688

1.2.2. Mathematical modeling, numerical methods and software packages (Physical and mathematical sciences)

ON THE DEVELOPMENT OF A SET OF COMPUTER PROGRAMS FOR AUTOMATED PREPARATION OF PRACTICAL TASKS IN THE SECTION "LINEAR PROGRAMMING PROBLEMS" OF THE ACADEMIC DISCIPLINE HIGHER MATHEMATICS AND CHECKING THE WORK COMPLETED BY STUDENTS

Laptev Sergey Vladimirovich
Cand. Phys.-Math. Sci., associate professor
Kuban State Agricultural university, 350044, Russia, Krasnodar, Kalinin st., 13, E-mail: mail@kubsau.ru

Mikhaylenko Evgeny Vladimirovich
Cand. Phys.-Math. Sci., associate professor,
Deputy Head of a Department
Krasnodar University of Ministry of the Interior of Russian Federation, 350005, Russia, Krasnodar, Yaroslavskaya 128. E-mail: post@krdu-mvd.ru

The article presents a new software product created by the authors for the automated preparation of practical assignments in the section "Problems of Linear Programming" of the discipline of higher mathematics and checking the work completed by students. The author reveals the prerequisites, describes the main methods and technologies used in the development of a set of programs. It sets out the main principles of developing the presented software products, describes the structure of typical software developments and the content of the software modules included in them, and examines the algorithms for generating various classes of problems in sections of higher mathematics: The described applications are illustrated with figures and code fragments of the described software applications. For the teaching staff, software engineers of educational organizations.

Keywords: AUTOMATION, ALGORITHM, LINEAR PROGRAMMING, MACROS, SIMPLEX METHOD, PROGRAM MODULE

The specificity of studying mathematical disciplines in higher educational institutions of the Ministry of Internal Affairs of Russia consists not only in the acquisition of certain theoretical knowledge by cadets and students. Stable skills of applying the studied theoretical material by students to solve practical problems are important.

An important place in the process of studying mathematical disciplines by students is the preparation of assignments for practical classes and independent work. Traditionally, for these purposes, teachers select tasks from well-known or independently written textbooks, manuals, collections of problems and other educational and methodological publications. However, the effectiveness of using such materials is extremely low. It is necessary to note the significant limitation of the number of typical problems in these publications. Teachers are forced to issue a general list of problems for all students during practical classes or for independent preparation, or at best, assignments with several options. In this regard, students lose motivation to independently complete such general "for all" assignments. Often, there is a simple copying of completed work from conscientious classmates or even rewriting of solutions from previous years [2].

Finally, checking such work creates serious difficulties for the teacher as well. Analyzing students' handwritten work requires enormous labor costs. In addition, it is simply impossible to evaluate the quality of work completed by students outside of class time, at best through the joint efforts of the entire study group[2].

Our article presents a set of programs that allow automatic creation of a large number of typical calculation tasks, and after their completion by students, also check and evaluate them. The described software product is implemented in the Visual Basic for Application environment, which is the main toolkit of office packages. All programs are written as macros called by the teacher when generating tasks and verifying them.

The essence of the project is to generate numerical and symbolic values for typical tasks selected by the teacher [1]. When software processes a document with an unfilled task template, pre-prepared lists of students are automatically processed (Fig. 1).

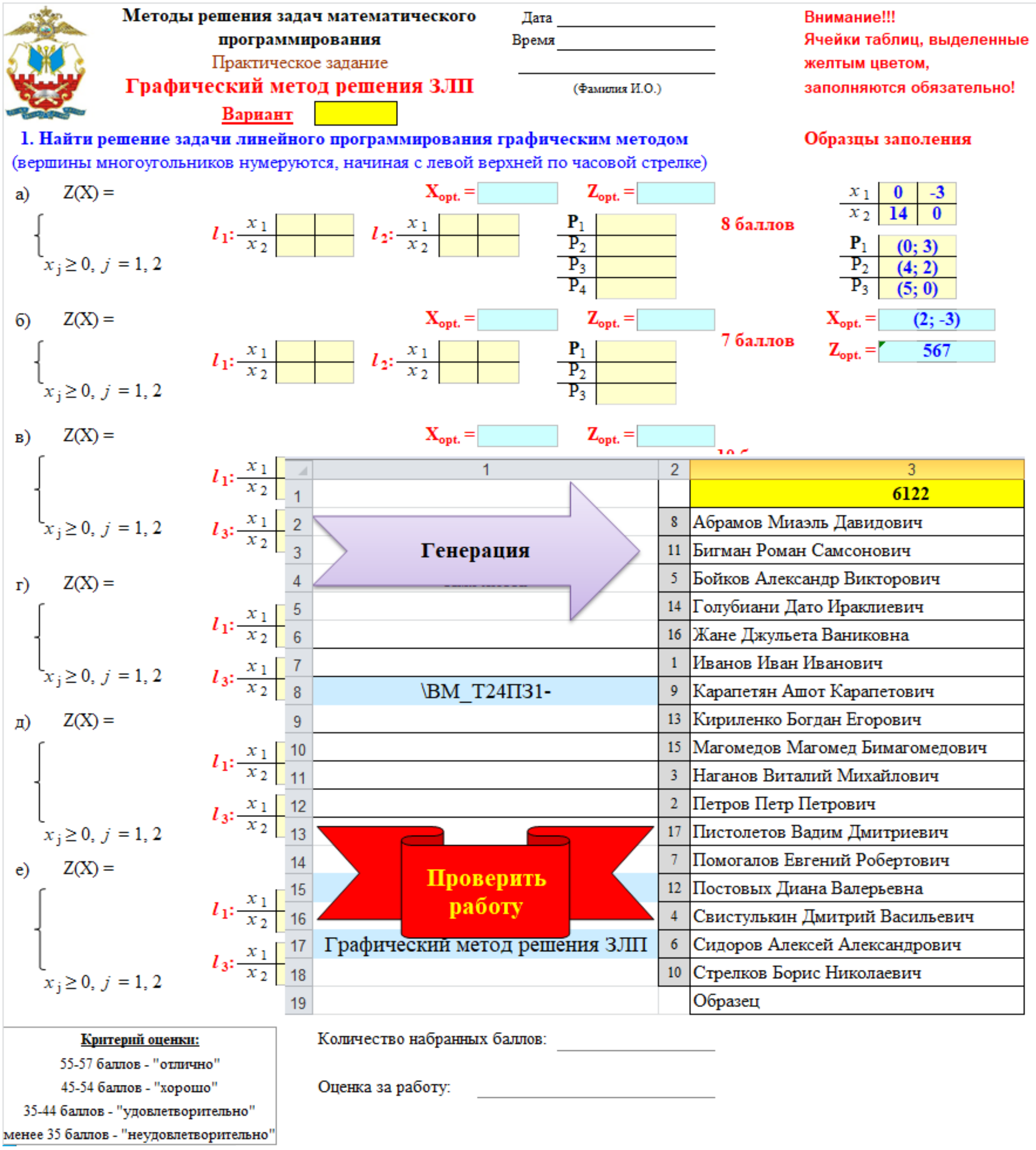


Fig. 1. Blank assignment template and list of trainees.

In case of creating a task for one student, the teacher needs to select the cell with the required surname and click on the banner "Check work", and to quickly generate tasks for the entire study group, it is enough to select the group number. The number of group lists on the sheet is not limited.

The date and time automatically filled in when creating an electronic assignment, as well as the student's last name and initials, displayed in the upper right corner of the template, are used to calculate the version number. During the automated check of the work, the compliance of the version number with the time and last name of the student specified in the work is verified [5]. This procedure allows avoiding forgeries of completed work, and if all four components are inconsistent: version, date, time and the record identifying the student, the work is not checked. The parameters of each task in the proposed works are generated according to developed algorithms, which necessarily include the version number [3].

Assignments are distributed by teachers usingeducational electronic course management systems MOODLE, and even more often with the help ofcloud technologies. Access to the assignment database and checked work is open to all students 24/7. The teacher creates a virtual interactive group, all students connect to it, who can then use the current educational resources posted by the teacher: multimedia slides, lecture texts, individual assignments for independent preparation, examples of problem solving, reference materials. With the help of "cloud" technologies, you can also ask the teacher and other group members questions, send completed work for checking, organize video and audio conferences.

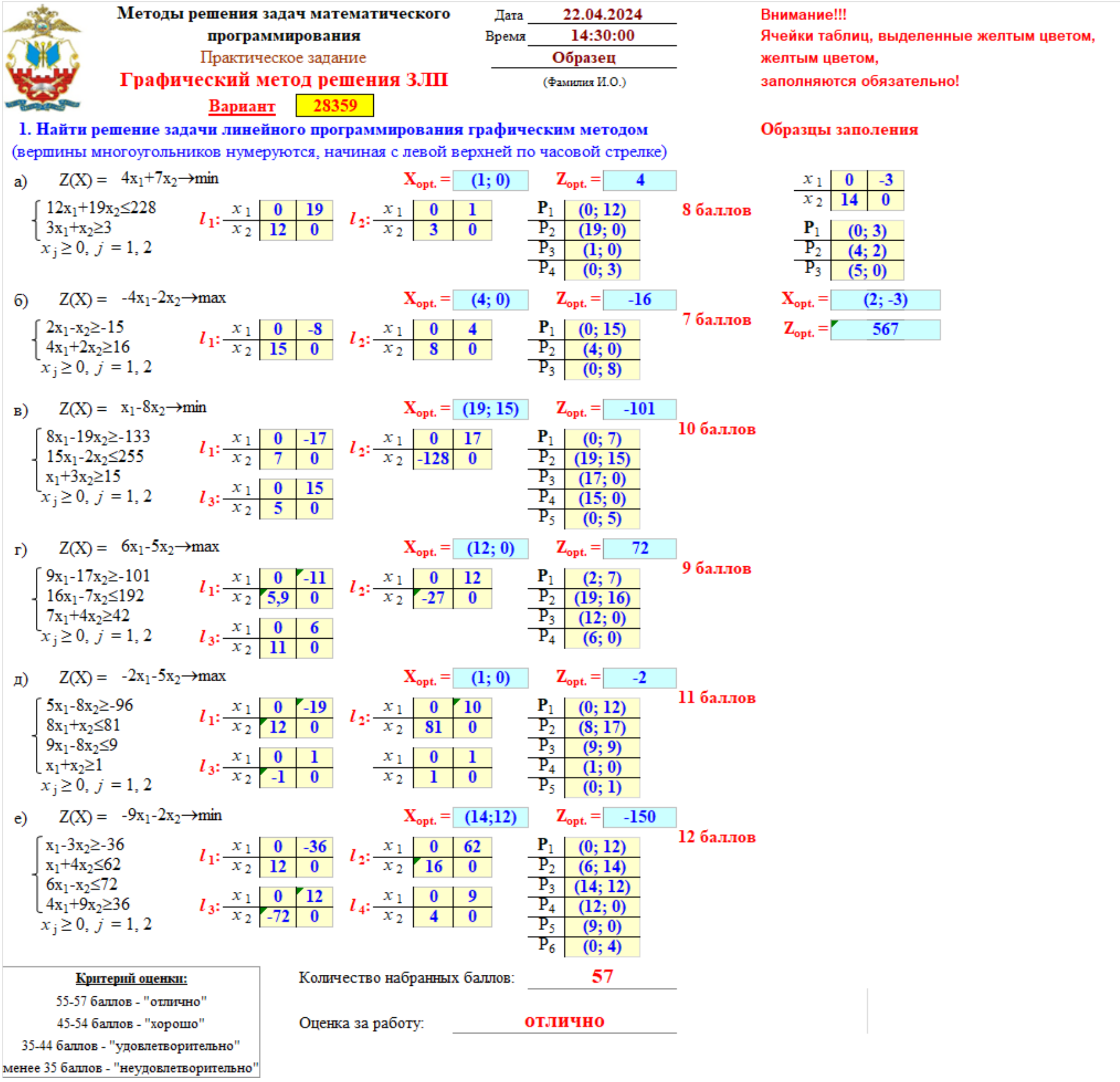
In the process of working on the creation of software products in various areas of mathematics, we had to face a number of problems, most of which were successfully resolved. Let's consider an example of developing software for creating calculation tasks for cadets of the Krasnodar University of the Ministry of Internal Affairs of Russia, studying in the specialty 10.05.05 Security of information technologies in the law enforcement sphere.

The described set of computer programs is intended for the teacher to generate calculation tasks for finding optimal solutions using linear programming methods. In addition, the process of verifying the results of the work performed by students is fully automated.

Six software modules with templates linked to them include practical work for training students in the graphical method for solving linear programming problems (LPP), acquiring skills in using the Jordan-Gauss method for solving systems of linear algebraic equations, single substitution transformations and simplex transformations. An important part of the software package are modules aimed at mastering the simplex method for solving LPP and solving dual problems.

The module "Graphical method for solving linear programming problems" (Fig. 2) is designed to solve linear programming problems of two variables. The method is based on the possibility of graphically depicting the region of admissible solutions to the problem and finding the optimal solution among the vertices of the resulting polygon [4].

When solving the generated problems, the student must determine and derive the coordinates of any points along which the lines are constructed, limiting solutions of inequalities, the vertices of a polygon, which is the area of feasible solutions of a linear programming problem, as well as optimal solutions. The task is designed for students to complete during a practical lesson.



Методы решения задач математического программирования

Практическое задание

Графический метод решения ЗЛП

Вариант 28359

Дата 22.04.2024

Время 14:30:00

Образец

(Фамилия И.О.)

Внимание!!!
Ячейки таблиц, выделенные желтым цветом, желтым цветом, заполняются обязательно!

1. Найти решение задачи линейного программирования графическим методом
(вершины многоугольников нумеруются, начиная с левой верхней по часовой стрелке)

Образцы заполнения

x_1	0	-3
x_2	14	0

P_1 (0; 3), P_2 (4; 2), P_3 (5; 0)

$X_{opt.}$ = (2; -3), $Z_{opt.}$ = 567

а) $Z(X) = 4x_1+7x_2 \to \min$

$12x_1+19x_2 \le 228$; $3x_1+x_2 \ge 3$; $x_j \ge 0,\ j=1,2$

l_1: x_1: 0, 19; x_2: 12, 0. l_2: x_1: 0, 1; x_2: 3, 0.

$X_{opt.}$ = (1; 0), $Z_{opt.}$ = 4

P_1 (0; 12), P_2 (19; 0), P_3 (1; 0), P_4 (0; 3) — 8 баллов

б) $Z(X) = -4x_1-2x_2 \to \max$

$2x_1-x_2 \ge -15$; $4x_1+2x_2 \ge 16$; $x_j \ge 0,\ j=1,2$

l_1: x_1: 0, -8; x_2: 15, 0. l_2: x_1: 0, 4; x_2: 8, 0.

$X_{opt.}$ = (4; 0), $Z_{opt.}$ = -16

P_1 (0; 15), P_2 (4; 0), P_3 (0; 8) — 7 баллов

в) $Z(X) = x_1-8x_2 \to \min$

$8x_1-19x_2 \ge -133$; $15x_1-2x_2 \le 255$; $x_1+3x_2 \ge 15$; $x_j \ge 0,\ j=1,2$

l_1: x_1: 0, -17; x_2: 7, 0. l_2: x_1: 0, 17; x_2: -128, 0. l_3: x_1: 0, 15; x_2: 5, 0.

$X_{opt.}$ = (19; 15), $Z_{opt.}$ = -101

P_1 (0; 7), P_2 (19; 15), P_3 (17; 0), P_4 (15; 0), P_5 (0; 5) — 10 баллов

г) $Z(X) = 6x_1-5x_2 \to \max$

$9x_1-17x_2 \ge -101$; $16x_1-7x_2 \le 192$; $7x_1+4x_2 \ge 42$; $x_j \ge 0,\ j=1,2$

l_1: x_1: 0, -11; x_2: 5,9, 0. l_2: x_1: 0, 12; x_2: -27, 0. l_3: x_1: 0, 6; x_2: 11, 0.

$X_{opt.}$ = (12; 0), $Z_{opt.}$ = 72

P_1 (2; 7), P_2 (19; 16), P_3 (12; 0), P_4 (6; 0) — 9 баллов

д) $Z(X) = -2x_1-5x_2 \to \max$

$5x_1-8x_2 \ge -96$; $8x_1+x_2 \le 81$; $9x_1-8x_2 \le 9$; $x_1+x_2 \ge 1$; $x_j \ge 0,\ j=1,2$

l_1: x_1: 0, -19; x_2: 12, 0. l_2: x_1: 0, 10; x_2: 81, 0. l_3: x_1: 0, 1; x_2: -1, 0. x_1: 0, 1; x_2: 1, 0.

$X_{opt.}$ = (1; 0), $Z_{opt.}$ = -2

P_1 (0; 12), P_2 (8; 17), P_3 (9; 9), P_4 (1; 0), P_5 (0; 1) — 11 баллов

е) $Z(X) = -9x_1-2x_2 \to \min$

$x_1-3x_2 \ge -36$; $x_1+4x_2 \le 62$; $6x_1-x_2 \le 72$; $4x_1+9x_2 \ge 36$; $x_j \ge 0,\ j=1,2$

l_1: x_1: 0, -36; x_2: 12, 0. l_2: x_1: 0, 62; x_2: 16, 0. l_3: x_1: 0, 12; x_2: -72, 0. l_4: x_1: 0, 9; x_2: 4, 0.

$X_{opt.}$ = (14;12), $Z_{opt.}$ = -150

P_1 (0; 12), P_2 (6; 14), P_3 (14; 12), P_4 (12; 0), P_5 (9; 0), P_6 (0; 4) — 12 баллов

Критерий оценки:
55-57 баллов - "отлично"
45-54 баллов - "хорошо"
35-44 баллов - "удовлетворительно"
менее 35 баллов - "неудовлетворительно"

Количество набранных баллов: 57

Оценка за работу: отлично

Rice. 2. Completed and verified module template "Graphical method for solving LLP".

Let's consider the principles of automatic filling of the task template using the example of the code for generating "Task 1v". In the task condition, we must specify the target function $Z(x_1, x_2)$ and derive a system of restrictions $\begin{cases} \varphi_i(x_1,\ x_2) = 0,\ i = 1,\ 2, \dots, k; \\ \varphi_j(x_1,\ x_2) \le (\ge) 0,\ j = k+1,\ k+2, \dots, n \end{cases}$, as well as the conditions for non-negativity of variables x_1 And x_2 .

Let's start with the constraint system. To build the system, given that the range of admissible values (RV) is limited by the first quadrant of the plane

x_1Ox_2, we will determine the coordinates of the five corner vertices of the ODR. Due to the fact that the vertex traversal according to the problem condition is performed clockwise, we will place the first and fifth vertices on the axis Ox_2, we will define the second vertex arbitrarily in the first quadrant, taking into account that the ODR pentagon is convex, and the third and fourth vertices must be on the axis Ox_1.

Figure 3 shows a fragment of the code for defining the array of coordinates of the vertices of the ODR. In the body of the Do .. Loop Until cycle, we generate the coordinates of the vertices with the condition of convexity of the ODR. The number of the V variant serves as the generation parameter.

```
s = "x"                        'Обозначение переменной
Do
                               'Координаты 1-й угловой вершины
  mP(1, 1) = 0
  mP(1, 2) = Rnd(v) * 1000 Mod 19 + 1
                               'Координаты 2-й угловой вершины
  mP(2, 1) = Rnd(v) * 1000 Mod 19 + 1
  mP(2, 2) = Rnd(v) * 1000 Mod 19 + 1
                               'Координаты 3-й угловой вершины
  mP(3, 1) = Rnd(v) * 1000 Mod 19 + 1
  mP(3, 2) = 0
                               'Координаты 4-й угловой вершины
  mP(4, 1) = Rnd(v) * 1000 Mod 19 + 1
  mP(4, 2) = 0
                               'Координаты 5-й угловой вершины
  mP(5, 1) = 0
  mP(5, 2) = Rnd(v) * 1000 Mod 19 + 1

Loop Until mP(1, 2) > mP(5, 2) And mP(2, 2) > mP(1, 2) And _
           mP(2, 1) > mP(3, 1) And mP(3, 1) > mP(4, 1)
```

Fig. 3. A fragment of the program code for determining the coordinates of the ODR vertices

Given the coordinates of the vertices of the ODR, we calculate the coefficients of the inequalities of the constraint system. The first boundary of the ODR pentagon is drawn through the first and second vertices, the second boundary will pass through the second and third vertices, the third and fifth boundaries are already determined by the condition of non-negativity of the variables, and the fourth, determining the third inequality, is determined by the fourth and fifth vertices.

Figure 4 shows a fragment of the program code for determining the inequality coefficients of the constraint system and outputting these inequalities. To calculate the inequality coefficients in general form for two points, the previously written ABC_ObsUr2Points procedure is called. To output the constraint system to the sheet, the LinNer3 function is used in the body of the For .. Next cycle.

```
                        'Коэффициенты неравенств системы ограничений
mAS(1) = Chr(179)       '">="
Call ABC_ObsUr2Points(mA(1, 1), mA(1, 2), mA(1, 0), _
                mP(1, 1), mP(1, 2), mP(2, 1), mP(2, 2))
mAS(2) = Chr(163)        '">="
Call ABC_ObsUr2Points(mA(2, 1), mA(2, 2), mA(2, 0), _
                mP(3, 1), mP(3, 2), mP(2, 1), mP(2, 2))
mAS(3) = Chr(179)        '">="
Call ABC_ObsUr2Points(mA(3, 1), mA(3, 2), mA(3, 0), _
                mP(4, 1), mP(4, 2), mP(5, 1), mP(5, 2))

                         'Вывод системы ограничений
For i = 1 To 3
  t = LinNer3(mA(i, 1), mA(i, 2), 0, -mA(i, 0), i * 2 + 34, 2, mAS(i), s)
Next i
```

Fig. 4. Fragment of the program code for determining and deriving inequalities of the system of constraints

And finally, we need to calculate the coefficients and output the target function (Fig. 5). In the body of the Do .. Loop Until cycle, the coefficients are generated z_1 And z_2 objective function $Z(x_1, x_2)$, and then the coefficients in the first three vertices of the ODR pentagon are checked. Using the LinNer3 function, we output the target function to the sheet.

```
                                'Расчет и вывод целевой функции
Do
  prop = True
  z1 = Rnd(v) * 1000 Mod 20 - 10
  z2 = Rnd(v) * 1000 Mod 20 - 10
  If z1 * mA(1, 1) + z2 * mA(1, 2) = 0 Then prop = False
  If z1 * mA(2, 1) + z2 * mA(2, 2) = 0 Then prop = False
  If z1 * mA(3, 1) + z2 * mA(3, 2) = 0 Then prop = False
Loop Until prop = True And z1 * z2 <> 0
                                'Вывод целевой функции
t = LinNer3(z1, z2, 0, "min", 33, 4, Chr(174), s)
```

Fig. 5. Fragment of the program code for calculating and outputting the objective function

Now let's consider the program for checking the completed tasks. In the same way as when loading, we generate five corner vertices of the ODR polygon with the V parameter and calculate the inequality coefficients from the constraint system and the objective function. The Rnd function used, when used at the beginning of the program with a negative argument and then running the Randomize instruction, produces the same pseudo-random sequences each time using the same arguments as when generating. Therefore, there is no need to analyze the task data, it is enough to know the variant number and use the already written algorithms again.

Having received the parameters of the problem condition, we will check the points entered by the student to construct the boundaries (Figure 6).

The coordinates of the points are substituted into the calculated equations of the lines, and if they satisfy the equation, the variable m is increased by one, otherwise the corresponding cells are crossed out. If two points for one line have correct coordinates, the user gets one point.

The main results of the task are finding and entering into the template the coordinates of the corner points and the optimal solution. Figure 7 shows a fragment of the program code for checking and verifying the optimal solution.

We have already generated the coordinates of the vertices of the ODR polygon, therefore, to verify the filled fields, it is sufficient to compare their contents with the known coordinates in the For .. Next cycle. Verification of the optimal solution comes down to calculating it by the enumeration method and comparing it with the corresponding filled fields.

```
                        'Проверка точек для построеня границ
m = 0                   '1-я прямая
For j = 1 To 2
 x1 = Cells(36, j + 7)
 x2 = Cells(38, j + 7)
 If mA(1, 1) * x1 + mA(1, 2) * x2 + mA(1, 0) = 0 Then
     m = m + 1
   Else
     r = krest(36, j + 7)
     r = krest(38, j + 7)
  End If
Next j
If m = 2 Then mark = mark + 1

m = 0                   '2-я прямая
For j = 1 To 2
 x1 = Cells(36, j + 11)
 x2 = Cells(38, j + 11)
 If mA(2, 1) * x1 + mA(2, 2) * x2 + mA(2, 0) = 0 Then
     m = m + 1
   Else
     r = krest(36, j + 11)
     r = krest(38, j + 11)
  End If
Next j
If m = 2 Then mark = mark + 1

m = 0                   '3-я прямая
For j = 1 To 2
 x1 = Cells(41, j + 7)
 x2 = Cells(43, j + 7)
 If mA(3, 1) * x1 + mA(3, 2) * x2 + mA(3, 0) = 0 Then
     m = m + 1
   Else
     r = krest(41, j + 7)
     r = krest(43, j + 7)
  End If
Next j
If m = 2 Then mark = mark + 1
```

Fig. 6. Checking the coordinates of points for constructing boundaries

```
                        'Проверка угловых точек
For i = 1 To 5
  Ps = "(" + st(mP(i, 1)) + ";" + st(mP(i, 2)) + ")"
  Qs = without_spaces(Cells(2 * i + 34, 16))
  If Ps = Qs Then mark = mark + 1 Else r = krest(2 * i + 34, 16)
Next i

                        'Проверка оптимального решения
Xopt = 1: Zopt = z1 * mP(1, 1) + z2 * mP(1, 2)
For i = 2 To 5
  Z = z1 * mP(i, 1) + z2 * mP(i, 2)
  If Z < Zopt Then
      Zopt = Z
      Xopt = i
  End If
Next i

Ps = "(" + st(mP(Xopt, 1)) + ";" + st(mP(Xopt, 2)) + ")"
Qs = without_spaces(Cells(33, 13))
If Ps = Qs Then mark = mark + 1 Else r = krest(33, 13)

Z = Cells(33, 17)
If Zopt = Z Then mark = mark + 1 Else r = krest(33, 17)
```

Fig. 7. Checking corner points and optimal solution.

In the future, we will not dwell on a detailed description of the tasks in subsequent modules, but will only give a brief description of them.

The module "Jordan-Gauss Method" (Fig. 8) is devoted to solving systems of linear equations using the Jordan-Gauss algorithm. By performing Jordan-Gauss transformations, the student determines whether the generated system is compatible or incompatible, and in the case of compatibility, it is definite or indefinite. At each step of the transformations, the system coefficients are entered into tables, the basic variables are noted, and the checksums are calculated for each calculated line.

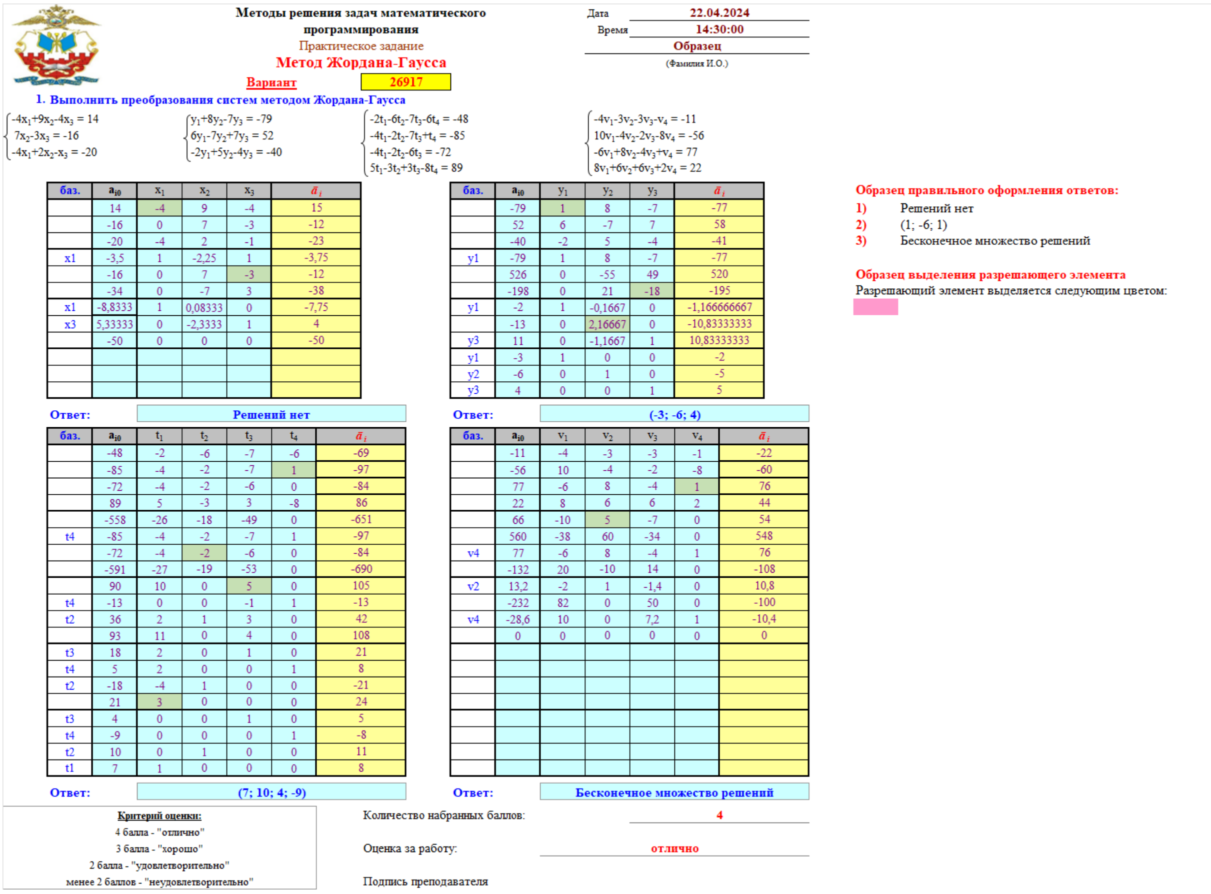


Fig. 8.Module "Jordan-Gauss Method".

In working with the module "Single Substitution Transformations" (Fig. 9), students also use the algorithm of the Jordan-Gauss method. By solving a system of m linear equations with n unknowns, they reduce the system to a single basis. By introducing free variables into the basis, the basic variables are transformed into free ones and new basic solutions are obtained. Such transformations are called single substitutions. The task of the students is to find all the basic solutions of the generated systems using single substitution transformations. The results of all transformations are entered into tables, and then the basic solutions are written out.

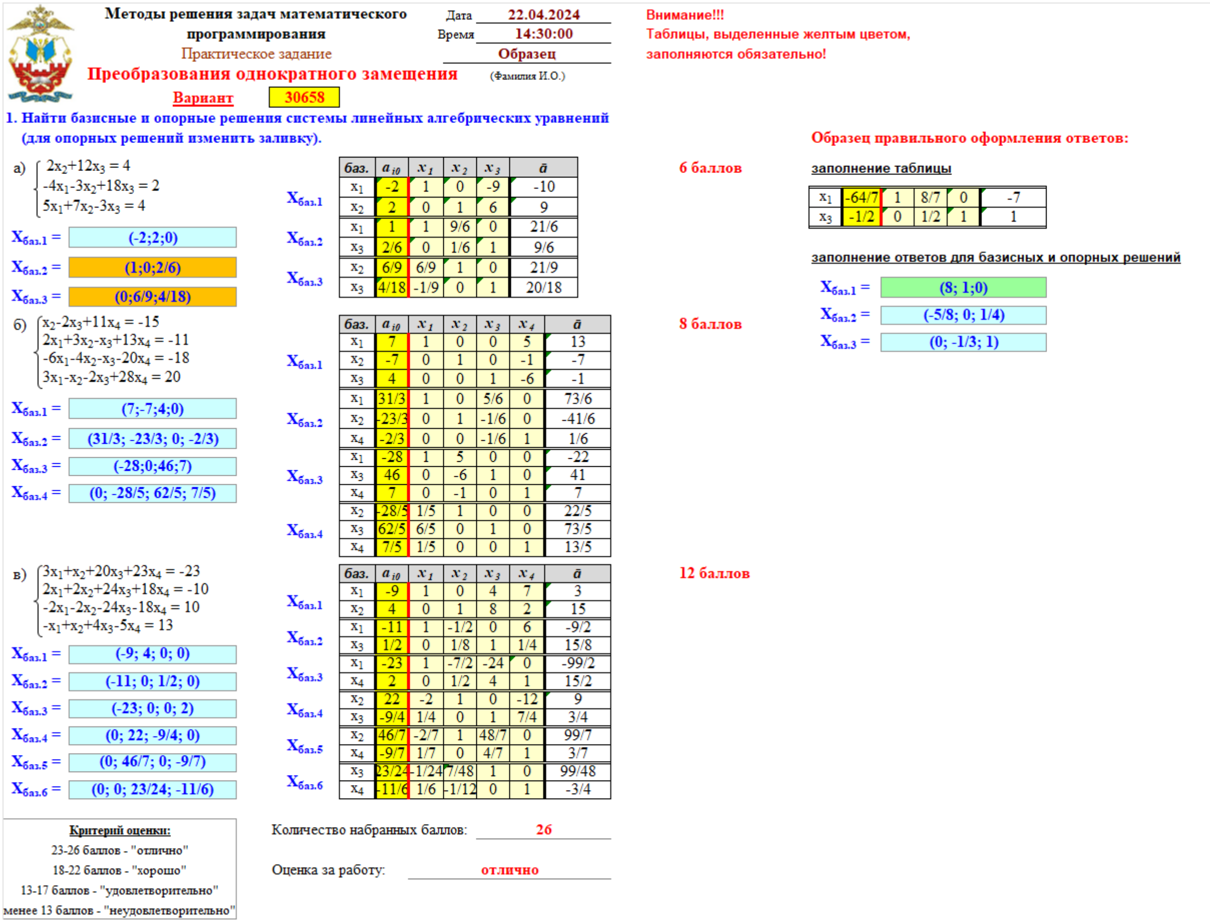


Fig. 9.Module "Single Substitution Transformations".

In the Simplex Transformations module (Fig. 10), students also perform single-substitution transformations. However, a number of restrictions are imposed on the process:

1) The resolving element is selected only in the column where there are positive elements;

2) If there is only one positive element, then it is taken as the resolving element;

3) If there are several positive elements, then the one for which the ratio of the free term to it is the smallest is taken as the resolving element [4].

As a result of executing this algorithm, all reference solutions are determined.

In the first task of the work, it is necessary to perform simplex transformations of already prepared reference solutions, and in the second, it is necessary to find any reference solution and, using simplex transformations, determine all the others.

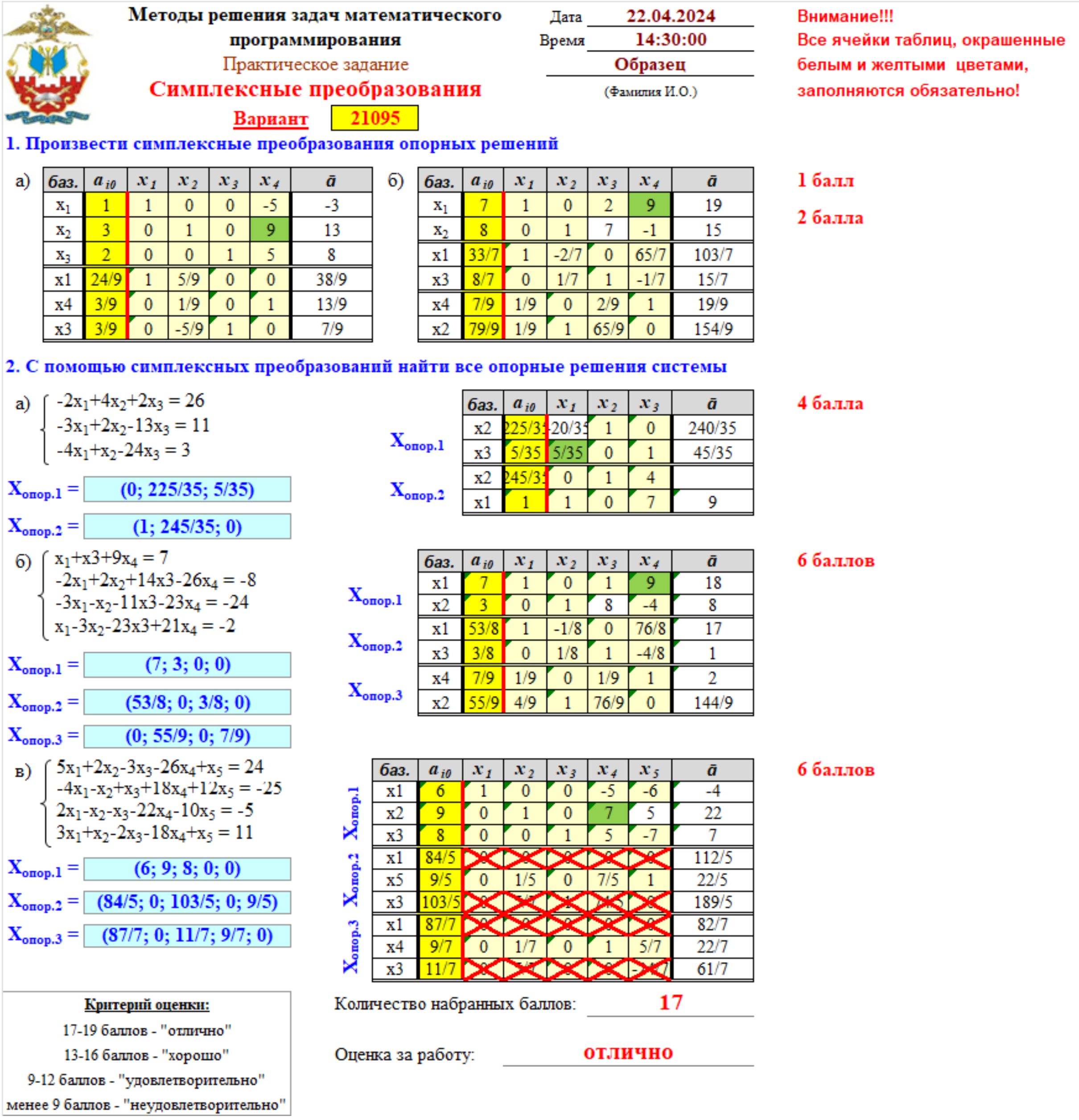


Fig. 10.Module "Simplex Transformations".

Perhaps the most labor-intensive module to complete is the “Simplex method for solving linear programming problems” (Fig. 11).

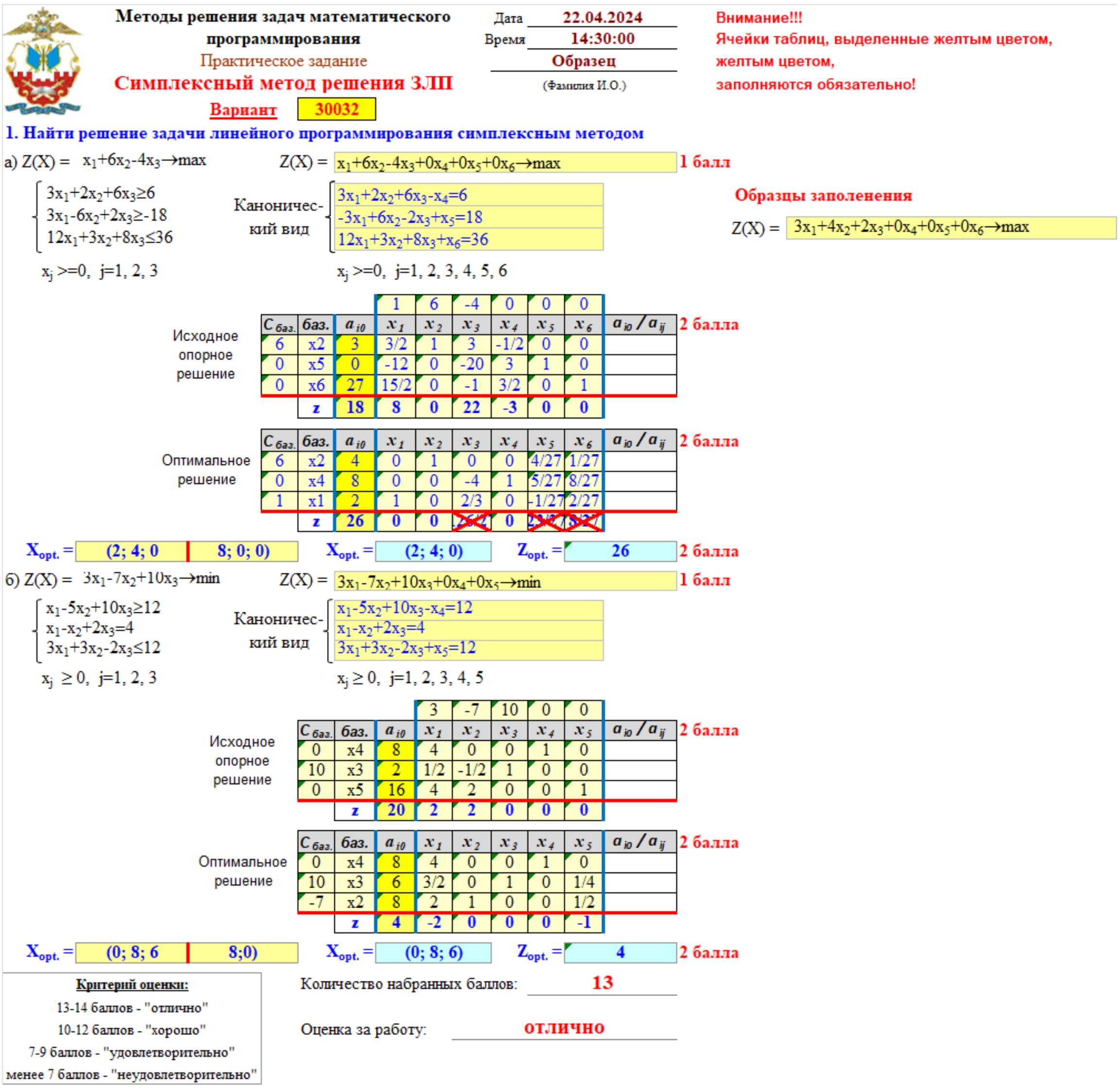


Fig. 11.Module "Simplex method for solving LLP".

Before solving a problem using the simplex method, students transform the generated system so that the right-hand sides of the constraint system are non-negative. The resulting system is then written in canonical form.

Next, the system of equations is reduced to the initial reference solution using simplex transformations, and the resulting solution is checked for optimality. In the case of non-optimality of the reference solution, a resolving element is selected using the simplex method, and a transition to another basic solution is made [4].

Solving each problem will require several steps. The results of finding the initial reference and optimal solutions must be entered into tables.

The program implements the problem of linear programming for maximum and minimum, leading to systems in canonical form of four and five variables, respectively.

The Dual Problems module (Fig. 12) is designed to solve linear programming problems using duality theorems.

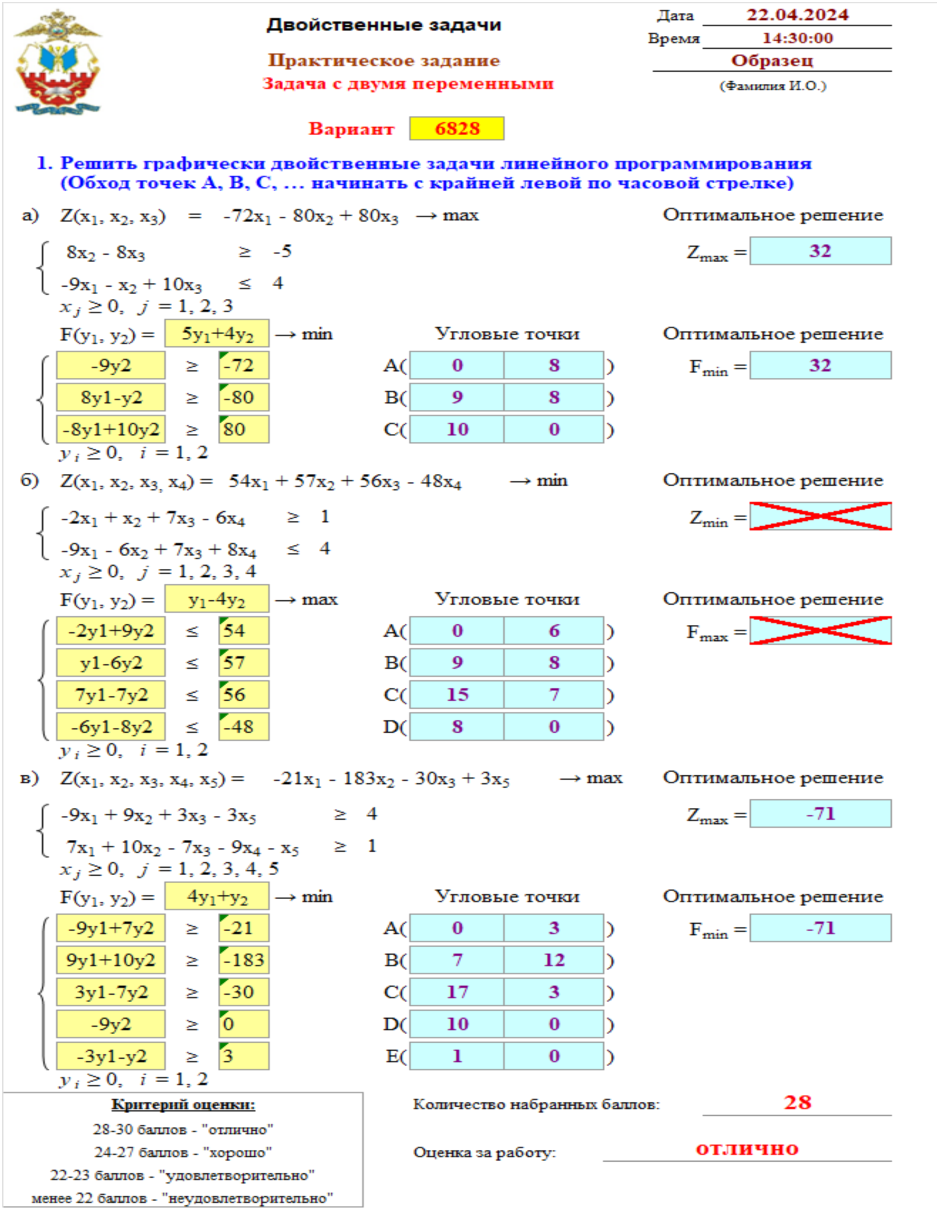


Fig. 12.Module "Dual problems".

Students are asked to create a dual problem with two arguments for a generated linear programming problem with more than two variables and solve it graphically.

In the program, students must enter the parameters of a new objective function and the corresponding system of constraints, write out the coordinates of the corner points of the region of admissible values, as well as the optimal solutions to a pair of dual problems.

The presented complex of programs has been implemented in the educational activities of the Department of Informatics and Mathematics of the Krasnodar University of the Ministry of Internal Affairs of Russia. The experience of using it in the educational process has yielded positive results. Currently, work is underway to create a number of similar programs to support the disciplines "Mathematics", "Mathematical Foundations of Information Processing" and "Econometrics".

List of references

1. Arshinov G.A. On the methods and technologies for developing computer programs for generating and checking assignments in mathematical disciplines / G.A. Arshinov, V.N. Laptev, E.V. Mikhailenko // Polythematic network electronic scientific journal of the Kuban State Agrarian University (Scientific journal of KubSAU) [Electronic resource]. - Krasnodar: KubSAU, 2023. - No. 09 (193). P. 50 - 81. - IDA [article ID]: 1932309008. - Access mode: http://ej.kubagro.ru/2023/09/pdf/08.pdf, 2 u.p.l.

2. Laptev V.N. On technologies for developing software applications for generating and checking practical assignments in mathematical disciplines / V.N. Laptev, E.V. Mikhailenko // Polythematic network electronic scientific journal of the Kuban State Agrarian University (Scientific journal of KubSAU) [Electronic resource]. - Krasnodar: KubSAU, 2020. - No. 01 (155). Pp. 164 - 177. - IDA [article ID]: 1552001013. - Access mode: http://ej.kubagro.ru/2020/01/pdf/13.pdf, 0.875 u.p.l.

3. Laptev V.N. Some aspects of using the Visual Basic for Application environment to create educational applications in mathematical disciplines / V.N. Laptev, E.V. Mikhailenko // Polythematic network electronic scientific journal of the Kuban State Agrarian University (Scientific journal of KubSAU) [Electronic resource]. - Krasnodar: KubSAU, 2014. - No. 09 (103). P. 222 - 233. - IDA [article ID]: 1031409014. - Access mode: http://ej.kubagro.ru/2014/09/pdf/14.pdf, 0.75 u.p.l.

4.Mikhailenko E.V.Higher Mathematics. Part 2: textbook / E. V. Mikhailenko. - Krasnodar: Krasnodar University of the Ministry of Internal Affairs of Russia, 2023. - 335 p.

5.Mikhailenko E.V. Methods for creating computer programs for automation of preparation and checking of practical assignments based on office applications. / E.V. Mikhailenko // Mathematical methods and information technology tools [Electronic resource]: Proc. XIX Int. scientific-practical. conf. (June 22, 2023) / editorial board: I.N. Starostenko, E.V. Mikhailenko, M.V. Sharpan et al. - Electronic data - Krasnodar: Krasnodar University of the Ministry of Internal Affairs of Russia, 2023. - 1 electronic optical disc. Pp. 199 - 211.