

O. Zelensky*,
orcid.org/0000-0001-8780-587X,
V. Lysenko,
orcid.org/0000-0002-5200-1211

State University of Economics and Technology, Kryvyi Rih,
Ukraine

* Corresponding author e-mail: zelensky@kneu.dp.ua

AUTOMATED STUDENT KNOWLEDGE TESTING AND CONTROL SYSTEM ZELIS

Purpose. To substantiate the methodology and develop the software of the automated system of testing and monitoring the knowledge of ZELIS students.

Methodology. Examination papers contain text, tables, pictures, formulas, etc., that is, all the possibilities of the MS WORD editor. Questions and answer options are generated randomly. As the operation showed, the most effective number is 30 questions and 5 answers to them. When working with forms, a large number of people are tested. Each form is scanned to receive a raster file in *.jpg format. To recognize these files, a mathematical apparatus has been developed that accurately determines the number of the correct answer based on the form. In online mode, students receive input data through the Internet, which is relevant in the conditions of martial law, as well as when testing students' knowledge during the educational process without additional use of laboratory time.

Findings. The ZELIS software is updated in the Visual Studio 2019 environment in the C# programming language using the MS ACCESS or SQL SERVER DBMS, as well as the FireBase RealTime DataBase cloud DBMS.

Originality. Input information for creating tests comes in a single RTF format that allows you to use tables, figures, formulas, etc. The system also provides the possibility of simultaneous testing of a large number of persons. This process uses proprietary recognition algorithms and takes only a few minutes to process a large number of forms.

Practical value. The article describes the automated system of testing and monitoring the knowledge of ZELIS students, which was developed by the authors. The system was developed at the State University of Economics and Technology and has been operating since 2015. Approximately 250 exams are conducted in one online paper-based testing session, with approximately 1,200 students participating.

Keywords: *knowledge testing, ZELIS, form, algorithm, RealTime DataBase, DBMS*

Introduction. The ZELIS system is designed to assess students' knowledge in two modes: forms and dialogue.

1. Knowledge assessment in the forms mode.

1.1. Knowledge assessment via offline forms.

1.2. Knowledge assessment via online forms.

2. Knowledge assessment in dialogue mode (in a computer lab – local network).

Developers: Oleksandr Semenovych Zelenskyi, Doctor of Technical Sciences, Professor; Volodymyr Serhiiiovych Lysenko, Candidate of Economic Sciences, Associate Professor.

The system was developed at the State University of Economics and Technology (Kryvyi Rih) and has been in operation since 2015. In one session alone, around 250 exams are conducted online using forms, with about 1,200 students participating.

Literature review. In recent years, automated testing and student knowledge assessment systems have been the focus of numerous scientific studies. Significant attention is given to the integration of information technology into the educational process, which enhances the efficiency and objectivity of knowledge evaluation [1, 2]. The use of automated systems not only saves time for instructors and students but also boosts motivation for learning through the immediate availability of results. Well-known systems such as Moodle, Google Classroom [3], and others [4, 5] are widely used in higher education institutions and show positive outcomes in student knowledge assessment.

Unsolved aspects of the problem. However, standard systems such as Moodle and Google Classroom have several drawbacks:

- input information is limited to text and images – many systems do not support MS Equation objects and tables [6, 7];
- there is no option to categorize questions by difficulty levels [8, 9];

- exams for each student and the questions within them are not randomized based on probability theory;

- there is no option to assess knowledge using different grading scales [10, 11];

- limited system capabilities: either offline or online modes only [12, 13].

Thus, there is a need to create a universal, automated knowledge assessment system, ZELIS, which will handle any information compatible with MS Word and operate in both offline and online modes.

Purpose. To justify the methodology and develop software for the automated testing and knowledge assessment system, ZELIS, for students.

Methodology. The ZELIS software has been updated in the Visual Studio 2019 environment using the C# programming language and supports databases such as MS ACCESS, SQL SERVER, and the cloud-based FireBase RealTime Database.

In offline mode, each question on the form is accompanied by five squares. The student marks the square with the correct answer using a checkmark or cross. The primary task is to recognize the form and identify the correct answer square. Each form is scanned to produce a raster image file in *.jpg format. A mathematical framework has been developed to accurately detect the correct answer on the form. First, the image is cleaned to ensure that each pixel is either black or white. Then, the square with the highest concentration of black pixels is selected.

In online mode, students receive input data over the Internet, which is essential during wartime conditions and facilitates ongoing knowledge testing throughout the academic process without requiring additional lab time.

Findings. Input Information. For all operating modes, the instructor prepares questions and answers in MS Word. The document is saved in an RTF file format. This format, while large in size due to the absence of styles and links, includes functions that operate with text, allowing the extraction of questions and answers.

Questions and answers can include tables, images, formulas, and other MS Word elements, utilizing the full range of the editor's capabilities. The instructor organizes the document as shown in Fig. 1.

Let us consider the main requirements for the document:

1. The document includes questions and their answers. The recommended number of questions is between 200 and

Structure of Table_B

No.	Title	Field Type	Description
1	nom_bilet	Long Integer	Ticket number
2	nom_vopr_bd	Long Integer	Question number from the "Table_A"
3	nom_pr_otv_bd	Long Integer	Correct answer number from "Table_A"
4	stoim_pr_otv	Double Floating Point	Weight coefficient for the question
5	per_otv	Text	Shuffling of answer options
6	nom_pr_otv	Long Integer	Number of the correct answer considering shuffling
7	nom_otv	Long Integer	Student's answer number

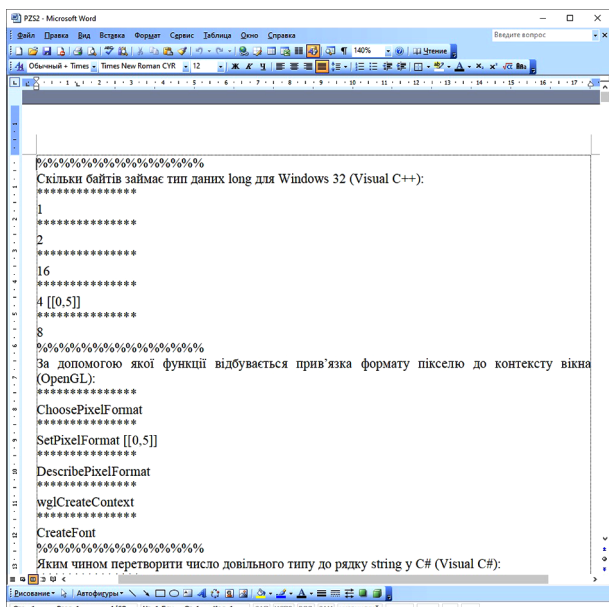


Fig. 1. Document for formulating questions and answers

300. Each question contains 5 answer options. Questions vary in difficulty levels, with weight coefficients of 0.5, 0.8 and 1. The number of questions at each difficulty level should be balanced.

2. Each question is separated by the line “%%%%%%%%%”, and each answer by the line “*****”.

3. To the right of the correct answer, a weight coefficient indicating the question's difficulty level is added within double square brackets “[]”, for example, “[0.8]” for the second difficulty level.

Based on the document, a database (DB) is created with questions and answers for the selected discipline. The database contains two tables: Table_A – for storing questions and answers, and Table_B – for generating examination papers from questions in Table_A.

Table 1 shows the structure of Table_A.

Table 2 shows the structure of Table_B.

The tables are related relationally through the field “nom_

vopr_bd”. Table “Table_B” is intended for the knowledge assessment mode using forms in both offline and online modes and contains only references to the “Table_A” (Fig. 2).

For the dialogue knowledge assessment mode, it is sufficient to use “Table_A”, as the student responds only to randomly generated questions from this table.

Table “Table_B” is used exclusively for the forms mode since the student answers questions on the generated forms, which contain a specified number of questions. In both the dialogue and forms modes, questions do not repeat.

Experience has shown that the optimal sample size for the number of questions on the form is 30. This number is also recommended for the dialogue mode. Of these, 10 questions will be generated for each of the three difficulty levels according to the weight coefficients of 0.5, 0.8 and 1. This is achieved using SQL queries with the random number generator function RND.

```
SELECT TOP 10 * FROM Table_A where stoim_pr_otv=0.5
ORDER BY Rnd(-1267666048*TimeValue(Now())*[nom_vopr_bd])
SELECT TOP 10 * FROM Table_A where stoim_pr_otv=0.8
ORDER BY Rnd(1083600640*TimeValue(Now())*[nom_vopr_bd])
SELECT TOP 10 * FROM Table_A where stoim_pr_otv=1
ORDER BY Rnd(716721536*TimeValue(Now())*[nom_vopr_bd])
```

Each instructor prepares an *.rtf document. This document is submitted to the administrator for database formation, or the database can be created by the instructor using a small program. The database file is then forwarded to the administrator.

After forming “Table_A” in dialogue mode and generating both tables (“Table_A” and “Table_B”) in forms mode, the preparation of output data for knowledge assessment is complete.

In the forms knowledge assessment mode, examination papers are generated from the database, containing a specified number of questions. The selection of questions and their corresponding answers is done randomly. Operational experience has shown that the most effective number of questions in an examination paper is 30, each with 5 answer options. Each student receives the questions and answers from the examination paper, along with a form to fill out one of the five answers (the correct answer is marked with a checkmark).

There are two methods for knowledge assessment using forms: offline and online.

In the offline method, the questions and answers, as well as the printed forms, are distributed to the students. This is typically done when there is a large number of students in the classroom. The completed forms are then submitted to the knowledge assessment administrator. In this case, a large number of computers are not required; it is sufficient to hand

Table 1

Structure of Table_A

No.	Title	Field Type	Description
1	nom_vopr_bd	Counter	Question number (primary key)
2	vopros	MEMO Field	Question in RTF format
3	nom_otv_1	MEMO Field	First answer in RTF format
4	nom_otv_2	MEMO Field	Second answer in RTF format
5	nom_otv_3	MEMO Field	Third answer in RTF format
6	nom_otv_4	MEMO Field	Fourth answer in RTF format
7	Nom_otv_5	MEMO Field	Fifth answer in RTF format
8	stoim_pr_otv	Double Floating Point	Weight coefficient for the question
9	nom_pr_otv	Long Integer	Number of the correct answer (from 1 to 5)

nom_bilet	nom_vopr_bd	nom_pr_otv_bd	stoin_pr_otv	per_otv	nom_pr_otv	nom_otv
1	22	3	0,5	3 2 1 4 5	1	0
1	138	2	0,5	5 1 4 3 2	5	0
1	52	3	0,5	4 3 2 5 1	2	4
1	13	5	0,5	5 3 2 1 4	1	0
1	129	1	0,5	1 3 5 2 4	1	0
1	111	5	0,5	5 1 2 3 4	1	0
1	21	3	0,5	5 2 1 4 3	5	0
1	50	2	0,5	5 3 1 4 2	5	0
1	125	1	0,5	4 1 5 2 3	2	0
1	29	3	0,5	4 5 2 3 1	4	0
1	144	3	0,8	5 2 4 1 3	5	0
1	106	1	0,8	5 1 2 3 4	2	0

Fig. 2. Generated table of examination papers "Table_B"

over the forms to the administrator. Each form is scanned to produce a raster image file in *.jpg format. A mathematical framework has been developed to accurately determine the correct answer based on the scanned forms. Over an extended period, there have been no failures. Subsequently, calculations are performed for all students, and the results are forwarded to the institution's administration.

Algorithm for recognizing forms in offline mode. In the offline knowledge assessment mode, each student receives an examination paper with questions and a form to fill in the correct answers. The template for the printed form is shown in Fig. 3.

On the form, 5 squares are allocated for each question. The student marks the square corresponding to the correct answer with a checkmark or a cross. After filling out the form, it is scanned to obtain a raster image file in *.jpg format.

The main task is to determine the square corresponding to the correct answer for each question in the given raster image file. To achieve this, the image is first cleaned so that each pixel is characterized by either black or white color. After this, the square with the maximum number of black pixels is selected.

Initially, let us consider how to classify each pixel in the image as black or white. The core idea of the method lies in comparing the intensity value of a pixel with the average value of its neighboring pixels. This adaptive thresholding method is

a simple extension of the Welner method, whose main concept is to compare each pixel with the average of the surrounding pixels [14]. The authors of this idea are Derek Bradley from Carleton University (Canada) and Gerhard Ross from the National Research Council of Canada [15].

To compute the intensity of each pixel, the following formula is used [16, 17]

$$I = 0.2125R + 0.7154G + 0.0721B, \quad (1)$$

where R , G , B are the intensities of the red, green, and blue colors, respectively.

These coefficients are selected based on the characteristics of how the human eye perceives images (sensitivity to green and blue colors) and are used in the high-definition television (HDTV) model [18].

This method operates using two passes over the image by employing integral matrices. The first pass is necessary for constructing the integral matrix, which contains the sum of pixel intensities I across rows and columns.

To calculate the integral matrix of the image, we the sum of all $P(x, y)$ elements to the left and above the pixel (x, y) store in each of its nodes $I(x, y)$. This is done using the following formula for the element-wise construction of the integral matrix

$$I(x, y) = I(x-1, y) + \sum_{j=0}^{y-1} f(x, j). \quad (2)$$

In Fig. 4 (left and center), an example of calculating the integral matrix is demonstrated. The input data consists of a 4 by 4 image matrix that contains pixel intensities, as shown in the left image.

For instance, to calculate the element of the matrix with indices $x = 1$ and $y = 2$, which equals 13 in the figure, we need to add the element located to the left of the number 13 (which is 7) to three elements located above, namely 1, 4 and 1.

As a result, we obtain

$$I(1, 2) = I(0, 2) + \sum_{j=0}^{y-1} f(x, j) = 7 + 1 + 4 + 1 = 13. \quad (3)$$

After obtaining the integral matrix, the sum function for any rectangle with the coordinates of the upper left corner

Fig. 3. Template of the form

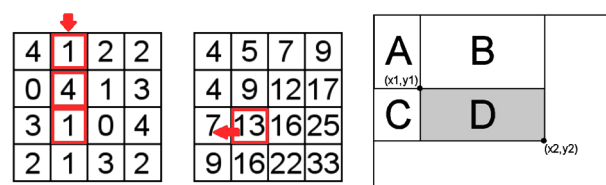


Fig. 4. Example of Using the Integral Matrix

(x_1, y_1) and the lower right corner (x_2, y_2) can be calculated using the following formula

$$\sum_{x=x_1}^{x_2} \sum_{y=y_1}^{y_2} f(x, y) = I(x_2, y_2) - I(x_1 - 1, y_2) - I(x_2, y_1 - 1) + I(x_1 - 1, y_1 - 1). \quad (4)$$

In Fig. 4, on the right, it is shown that calculating the sum $P(x, y)$ for rectangle D using formula (3) is equivalent to calculating the sums over the rectangles $(A + B + C + D) - (A + B) - (A + C) + A$.

Based on the obtained integral matrix, we will calculate the sum of pixels in the highlighted rectangle using the already defined integral matrix (Fig. 5).

For example, to determine the sum of pixels in a rectangle with the coordinates of the upper left corner $(x_1 = 1, y_1 = 1)$ and the lower right corner $(x_2 = 2, y_2 = 2)$, we proceed as follows

$$\sum_{x=1}^2 \sum_{y=1}^2 f(x, y) = I(2, 2) - I(0, 2) - I(2, 0) + I(0, 0) = 16 - 7 - 7 + 4 = 6. \quad (5)$$

Thus, using the obtained integral matrix and formula (4), we can immediately obtain the sum of the intensity values of the pixels in any rectangular area. This method works by using two passes over the image.

In the first pass, the integral matrix of the image is calculated using formula (2).

Let us present a fragment of the program code for calculating the integral matrix.

```
BitmapData bitmapData = bitmap.LockBits(new Rectangle(0, 0,
    bitmap.Width, bitmap.Height), ImageLockMode.ReadWrite,
    bitmap.PixelFormat);
byte* ptrFirstPixel = (byte*)bitmapData.Scan0; // BGR
intIImg = new long[bitmapData.Width, bitmapData.Height]; //
    Integral image, summed area table
long sum;
for (int x = 0; x < bitmapData.Width; x++)
{
    int xBytes = x * bytesPerPixel;
    sum = 0;
    for (int y = 0; y < bitmapData.Height; y++)
    {
        byte* currentLine = ptrFirstPixel + y * bitmapData.Stride;
        sum += (long)(0.2125 * currentLine[xBytes + 2] + 0.7154 *
            currentLine[xBytes + 1] + 0.0721 * currentLine[xBytes]);
        if (x == 0)
            intIImg[x, y] = sum;
        else
            intIImg[x, y] = intIImg[x - 1, y] + sum;
    }
}
```

Next, a square with a side length of pixels equal to 1/8 of the image width is selected, and the value t is set to 0.85 [19, 20]. In the second pass, it is necessary to calculate the average value of the pixel in a square of size $s \times s$ using the integral matrix. The square is constructed relative to the current pixel. Then, the intensity of the current pixel is compared with the average intensity of the pixels in that square. If the

4	1	2	2	4	5	7	9
0	4	1	3	4	9	12	17
3	1	0	4	7	13	16	25
2	1	3	2	9	16	22	33

Fig. 5. Example of determining the sum of pixels using the integral matrix

intensity value of the current pixel is less than the average value in the specified square multiplied by the value t , it is set to black; otherwise, it is set to white.

Let us present a fragment of the program code for the second pass over the image.

```
int s = bitmapData.Width / 8;
float t = 0.85f;
int hs = s / 2;
for(int x=0;x<bitmapData.Width;x++)
{
    int xBytes = x * bytesPerPixel;
    int x1, y1, x2, y2, count;
    for (int y = 0; y <
        bitmapData.Height; y++)
    {
        byte* currentLine = ptrFirstPixel +
            (y * bitmapData.Stride);
        x1 = Math.Max(1, x - hs);
        y1 = Math.Max(1, y - hs);
        x2 = Math.Min(bitmapData.Width - 1,
            x + hs);
        y2 = Math.Min(bitmapData.Height - 1,
            y + hs);

        count = (x2 - x1) * (y2 - y1);

        sum = intIImg[x2, y2] -
            intIImg[x1-1, y2] - intIImg[x2, y1-1] +
            intIImg[x1-1, y1-1];

        if ((0.2125 * currentLine[xBytes + 2] + 0.7154 *
            currentLine[xBytes + 1] + 0.0721 * currentLine[xBytes])
            * count < sum * t)
        {
            currentLine[xBytes] = 0;
            currentLine[xBytes + 1] = 0;
            currentLine[xBytes + 2] = 0;
        }
        else
        {
            currentLine[xBytes] = 255;
            currentLine[xBytes + 1] = 255;
            currentLine[xBytes + 2] = 255;
        }
    }
}
```

From the fragment, it can be seen that for each pixel, only the available information is taken to form the specified area relative to its center. This was not accounted for in the classical algorithm and is accomplished using the following lines to determine the top-left and bottom-right coordinates of the square, specifically x_1, y_1, x_2, y_2 .

```
x1 = Math.Max(1, x - hs);
y1 = Math.Max(1, y - hs);
x2 = Math.Min(bitmapData.Width - 1, x + hs);
y2 = Math.Min(bitmapData.Height - 1, y + hs).
```

In Fig. 6, the left side shows the initial input image, while the right side displays the result of processing using this algorithm. As seen in the figure, the image was not only converted to a black-and-white format but also cleaned of various "noise".

Thus, processing the image using this method allows for obtaining a clear two-color black-and-white image, separating useful information from the background, and eliminating the effects of lighting and "noise".

After obtaining a black-and-white image, it is necessary to determine the coordinates of the cell areas. To define all the necessary characteristics of the two-dimensional coordinate system for the cells, reference points – markers – are needed on the scanned file. When printing the form on a sheet, the markers are placed in the same coordinate system as the cells.

Knowing their geometric arrangement allows us to find their actual pixel coordinates on the raster image of the scanned

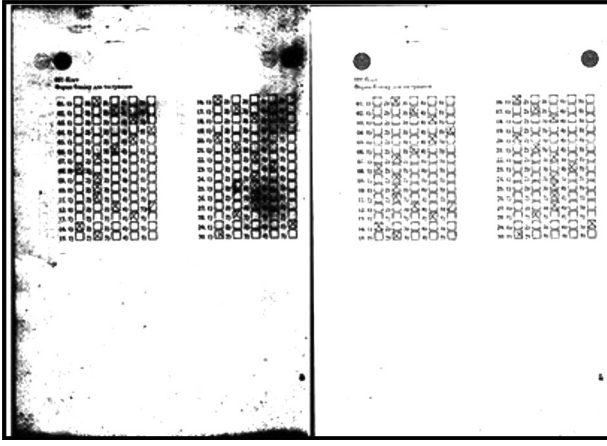


Fig. 6. Result of image cleaning and conversion to black-and-white format

file. There should be two such markers, as it is necessary to assess the distance between them to find the coordinates of the cells. The distance between them always correlates with all distances on the sheet and provides a characteristic scale – the ratio of distances between two pixels and the actual physical dimensions of the sheet. Knowing this data, we can calculate the real distance between two pixels in physical units. The angle between the two markers represents the angle of rotation of the sheet during scanning, and the coordinate system must be rotated accordingly, with the initial point considered the coordinates of one of the two markers, to align the scanned image.

To achieve the highest accuracy for the found rotation angle, the markers should be positioned at maximum distance apart. For the convenience of implementing the algorithm, the markers were arranged parallel to the X-axis (abscissa). If the form is printed accurately without distortions and the positions of the markers are correctly identified, all cell areas (and any point in the coordinate system) can be calculated with maximum precision. However, in practice, there is always some margin of error; thus, it makes sense to reduce the required area of the cell to ensure it fits within the actual cell and does not overlap its boundaries.

Fig. 7 shows the marker used for creating the form.

This marker has a circular shape, as its detection will not be affected by the rotation of the image.

The search for markers can be performed using the integral matrix, just as the average pixel value was computed during the image data preparation. It is necessary to find two of the most intensely square areas in the image, so these markers should be dense enough concerning the second graphic content of the form. The search area is square and will be sized to fit within the bounding circle of the marker, as the intensity sum in this area is the highest. The side length of this square is unknown, so it can be assumed to depend on the image size, for example, its width.

The dependency of marker sizes on the dimensions of the scanned file is determined after the form is created. To minimize these errors, it is essential to print and scan the forms as accurately as possible.

In this case, two markers are placed on the form: the left and the right. The left marker should be searched for as the most intense area in the left half of the image, while the right



Fig. 7. Marker

marker should be found in the right half; there is also no sense in searching in the lower half of the sheet.

After finding the coordinates of the markers, it is necessary to calculate other characteristics of the coordinate system in which they are located. The division value is calculated based on the distance between the markers. The rotation angle of the system is the angle between the markers. The left marker serves as the reference point, in this case, the upper left. Working with the system, we will place the coordinate axis with its center at the found center of the left marker. The x-axis is directed to the right, and the y-axis is directed down, just as in the data structure of System.Drawing.Bitmap.

In this case, it is convenient to work with coordinates that are relative to the distance between the markers because this distance reflects the actual physical size of the scanned image.

Next, we calculate the coordinate placement of the cell areas in the image, after which we count the sum of black pixels for each cell. Thus, we will obtain the intensities of all the cells. The more intense the cells, the blacker pixels they contain. Empty cells will have the lowest intensity, while filled cells will have the highest.

The intensity of each cell is compared with the recognition threshold. The recognition threshold for responses is set at just 10 percent and can be adjusted. This means that a correct answer will be considered if the filled area of the square exceeds 10 percent (Fig. 8).

As a result of recognizing the “paper” form, the data is transferred to an electronic form (Fig. 9).

In the second method, an online mode is implemented using Google FireBase cloud technologies, which is relevant under martial law and during student knowledge testing throughout the educational process without requiring additional laboratory time. For each examination paper, questions and answers are transmitted to the “cloud”. Students have an application that reads the questions and answers for their examination papers and fills out an electronic form, marking their correct answers. The results of their answers are sent to Google FireBase, where they are retrieved by the administrator. Further calculations are performed just as in the first case. The work is conducted within the time frame specified in the schedule.

In the knowledge control mode, examination papers are not formed in a dialogue, and students answer a specified number of questions. Questions and answers are generated randomly. This mode is used in a computer lab – a local network. This mode also includes student training, where the student sees the number of the correct answer and the answer they selected, thus learning in the process when dealing with a large number of questions.

Parameters for Student Evaluation. Grading Scale. At this stage, the evaluation system is chosen: 5-point, 12-point, 30-point, 40-point, 50-point, 100-point, or 200-point.

Let us present the formulas for calculating the grade.

To calculate the proportion of correct answers, the following formula is used

$$P_{cor} = \frac{3 \cdot (0.5 \cdot N_{cor0.5} + 0.8 \cdot N_{cor0.8} + N_{cor1})}{2.3 \cdot N_{tot}}, \quad (6)$$

where $N_{cor0.5}$ is the number of correct answers at the 1st level of difficulty with a weight coefficient of 0.5; $N_{cor0.8}$ is the number of correct answers at the 2nd level of difficulty with a weight coefficient of 0.8; N_{cor1} is the number of correct answers at the 3rd level of difficulty with a weight coefficient of 1; N_{tot} is the total number of questions in the examination paper.

The grade is calculated as follows.

$$Gr = Gr_{min} + (Gr_{max} - Gr_{min}) \cdot P_{cor}, \quad (7)$$

where Gr_{min} is the lower boundary of the grade; Gr_{max} is the upper boundary of the grade; P_{cor} is the proportion of correct answers.

Optimization of grading. When recalculating grades for the optimization mode, there is a reduction in the grade because,

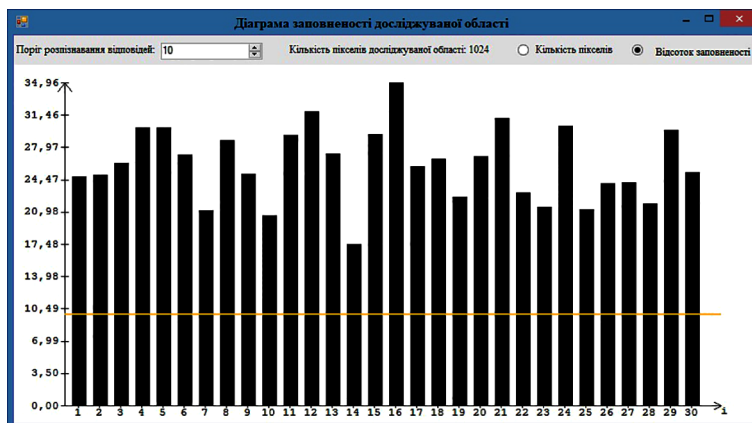


Fig. 8. Diagram of the fill level of the examined area

Fig. 9. Electronic form for testing

according to probability theory, with 5 answer options, the student already has a 20 % chance of answering questions correctly.

Let us present the following dependencies. The calculation of the maximum error for the selected grading system is determined by the following formula.

$$\Delta \max = \frac{(Gr_{\max} - Gr_{\min})}{5}. \quad (8)$$

The reduction in the grade is calculated as follows.

$$\Delta Gr = \Delta \max \frac{(Gr_{\max} - Gr)}{(Gr_{\max} - (Gr_{\min} + \Delta \max))}. \quad (9)$$

Next, the obtained grade Gr is reduced by the calculated decrease ΔGr . If the grade falls below the minimum threshold, it remains at that lower level.

The purpose of this optimization (the reduction of the grade) is to eliminate the random component that occurs during student testing.

For instance, for a grade in the 5-point evaluation system equal to 2.6, the maximum reduction would be 0.6, and the student would receive a grade of 2.

Let us consider this case.

Output Data is

$$Gr = 2.6. \quad Gr_{\min} = 2. \quad Gr_{\max} = 5.$$

Calculation is as follows

$$\Delta \max = \frac{(5-2)}{5} = 0.6;$$

$$\Delta Gr = 0.6 \cdot \frac{(5-2.6)}{(5-(2+0.6))} = 0.6 \cdot \frac{2.4}{2.4} = 0.6;$$

$$Gr_o = Gr - \Delta Gr = 2.6 - 0.6 = 2.$$

Thus, all 20 % of the correct answers that the student scored were deducted.

Conclusions. The article describes the automated testing and knowledge assessment system ZELIS, developed by the authors. The system was created at the State University of Economics and Technology and has been operational since 2015. During one testing session in online form, approximately 250 exams are conducted with about 1,200 students participating.

The input information for creating tests is provided in a unified RTF format, which allows the use of tables, images, formulas, etc. The system also enables simultaneous testing of a large number of individuals. During this process, proprietary algorithms for recognition are utilized, processing a significant number of answer sheets in just a few minutes.

Additionally, the system supports working with electronic answer sheets using the Firebase RealTime Database cloud database. This enables students to take exams and study online, which is particularly relevant during wartime.

The ZELIS software has been updated in the Visual Studio 2019 environment, written in the C# programming language, and uses the MS ACCESS or SQL SERVER DBMS, as well as the Firebase RealTime Database cloud DBMS.

References.

1. Drissi, S., & Amirat, A. (2016). An adaptive E-learning system based on student's learning styles: An empirical study. *International*

Journal of Distance Education Technologies, 14, 34-51. <https://doi.org/10.4018/IJDET.2016070103>.

2. Ethink (2018). *Three Benefits of Adaptive Learning in Your LMS*. Retrieved from <https://ethinkeducation.com/blog/3-benefits-utilizing-adaptive-learning-lms>.

3. *Moodle vs Google Classroom: Key Differences You Should Know*. (n.d.). Retrieved from <https://www.teachfloor.com/blog/moodle-vs-google-classroom>.

4. Harati, H., Yen, C. J., Tu, C. T., Cruickshank, B., & Armfield, S. W. (2020). Online adaptive learning: A study of score validity of the adaptive self-regulated learning model. *International Journal of Web-Based Learning and Teaching Technologies*, 15, 18-35. <https://doi.org/10.4018/IJWLTT.2020100102>.

5. Bergey, B. W., Ketelhut, D. J., Liang, S., Natarajan, U., & Karakus, M. (2015). Scientific inquiry self-efficacy and computer game self-efficacy as predictors and outcomes of middle school boys' and girls' performance in a science assessment in a virtual environment. *Journal of Science Education and Technology*, 24, 696-708. <https://doi.org/10.1007/s10956-015-9558-4>.

6. Chiranjeevi, K., & Jena, U. (2018). SAR image compression using adaptive differential evolution and pattern search based K-means vector quantization. *Image Analysis and Stereology*, 37, 35-54. <https://doi.org/10.5566/ias.1611>.

7. Hesterman, D. (2017). *Report on Intensive Mode Delivery in Engineering, Computer Science, and Mathematics*. Retrieved from http://www.ecm.uwa.edu.au/_data/assets/pdf_file/0009/2700846/Hesterman-2015-UWA-ECM-Report-on-intensive-mode-delivery.pdf.

8. Hussain, A. J., Al Fayadh, A., & Radi, N. (2018). Image compression techniques: a survey in lossless and lossy algorithms. *Neurocomputing*, 300, 44-69. <https://doi.org/10.1016/j.neucom.2018.02.094>.

9. Jarno, M., & Bormin, H. (2012). Lossless compression of hyperspectral images using clustered linear prediction with adaptive prediction length. *IEEE Geoscience and Remote Sensing Letters*, 9, 1118-1121. <https://doi.org/10.1109/LGRS.2012.2191531>.

10. Latham, G., Seijts, G., & Slocum, J. (2016). The Goal-setting and goal orientation labyrinth. *Organizational Dynamics*, 45, 271-277. <https://doi.org/10.1016/j.orgdyn.2016.10.001>.

11. Li, J., & Liu, Z. (2019). Multispectral transforms using convolution neural networks for remote sensing multispectral image compression. *Remote Sensing*, 11, 759-779. <https://doi.org/10.3390/rs11070759>.

12. Murray, M. C., & Pérez, J. (2015). Informing and performing: A study comparing adaptive learning to traditional learning. *Informing Science: the International Journal of an Emerging Transdiscipline*, 18, 111-125. Retrieved from <http://www.inform.nu/Articles/Vol18/IS-Jv18p111-125Murray1572.pdf>.

13. Nussbaumer, A., Hillemann, E., Gütl, C., & Albert, D. (2015). A Competence-based service for supporting self-regulated learning in virtual environments. *J. Learn. Anal.*, 2, 101-133. <https://doi.org/10.18608/jla.2015.21.6>.

14. Panadero, E. (2017). A Review of Self-Regulated Learning: Six Models and Four Directions for Research. *Frontiers in Psychology*, 8, 422. <https://doi.org/10.3389/fpsyg.2017.00422>.

15. Sabourin, J., Mott, B., & Lester, J. (2013). Discovering behavior patterns of self-regulated learners in an inquiry-based learning environment. In Lane, H. C., Yacef, K., Mostow, J., Pavlik, P. (Eds.). *Lecture Notes in Computer Science: Artificial Intelligence in Education*, (pp. 209-218). Berlin/Heidelberg: Springer.

16. Shi, C., Zhang, J., & Zhang, Y. (2016). Content-based onboard compression for remote sensing images. *Neurocomputing* 191, 330-340. <https://doi.org/10.1016/j.neucom.2016.01.048>.

17. Villegas-Ch, W., Roman-Cañizares, M., Jaramillo-Alcázar, A., & Palacios-Pacheco, X. (2020). Data Analysis as a Tool for the Application of Adaptive Learning in a University Environment. *Applied Sciences*, 10, 7016. <https://doi.org/10.3390/app10207016>.

18. Lefei, Z., Liangpei, Z., & Dacheng, T. (2015). Compression of hyperspectral remote sensing images by tensor approach. *Neurocomputing*, 147, 358-363. <https://doi.org/10.1016/j.neucom.2014.06.052>.

19. Zemliachenko, A. N., Abramov, S. K., Lukin, V. V., Vozel, B., & Chehdi, K. (2015). Lossy compression of noisy remote sensing images with prediction of optimal operation point existence and parameters. *Journal of Applied Remote Sensing*, 9, 095066. <https://doi.org/10.1117/1.JRS.9.095066>.

20. Zhan, X., Zhang, R., Yin, D., & Huo, C. (2013). SAR image compression using multiscale dictionary learning and sparse representation. *IEEE Geoscience and Remote Sensing Letters*, 10, 1090-1094. <https://doi.org/10.1109/LGRS.2012.2230394>.

Автоматизована система тестування й контролю знань студентів ZELIS

О. С. Зеленський*, В. С. Лисенко

Державний університет економіки і технологій, м. Кривий Ріг, Україна

* Автор-кореспондент е-mail: zelensky@kneu.dp.ua

Мета. Обґрунтувати методику й розробити програмне забезпечення автоматизованої системи тестування та контролю знань студентів ZELIS.

Методика. Білети з питаннями містять текст, таблиці, малюнки, формули тощо, тобто всі можливості редактора MS WORD. Питання й варіанти відповідей формуються випадковим чином. Як показала перевірка, найбільш ефективною кількістю є 30 питань і 5 відповідей до них. При роботі із бланками тестується велика кількість людей. Кожен бланк сканується для отримання растрового файлу у форматі *.jpg. Для розпізнавання цих файлів розроблено математичний апарат, що достатньо точно визначає по бланку номер правильної відповіді. В онлайн-режимі студенти отримують вхідні дані через мережу Інтернет, що актуально в умовах воєнного стану, а також при тестуванні знань студентів протягом навчального процесу без додаткового використання лабораторного часу.

Результати. Програмне забезпечення ZELIS оновлено в середовищі Visual Studio 2019 мовою програмування C# із використанням СУБД MS ACCESS або SQL SERVER, а також хмарної СУБД Firebase RealTime DataBase.

Наукова новизна. Вхідна інформація для створення тестів надходить в єдиному форматі RTF, що дозволяє використовувати таблиці, малюнки, формули тощо. Система також надає можливість одночасного тестування великої кількості осіб. Під час цього процесу використовуються власні алгоритми для розпізнавання, і це займає лише декілька хвилин для обробки значної кількості бланків.

Практична значимість. У роботі наводиться опис автоматизованої системи тестування й контролю знань студентів ZELIS, що розроблена авторами. Система розроблена у Державному університеті економіки і технологій та діє із 2015 року. Під час одного сеансу тестування в режимі бланків онлайн проводиться приблизно 250 іспитів, в яких беруть участь близько 1200 студентів.

Ключові слова: тестування знань, ZELIS, бланк, алгоритм, RealTime DataBase, СУБД

The manuscript was submitted 02.09.24.