DEVELOPING A VBA PROGRAM TO GENERATE DOCUMENTS CONTAINING EXERCISES

Zoltán FABULYA

Abstract: *In online education, remote accountability has required the use of new techniques to minimize the potential of cheating due to limited control. With different test worksheets, we can solve this problem, so the students taking the tests at the same time cannot get better results by sharing their solutions with each other. However, in the case of a large number of candidates, it is not possible to create individual test worksheets manually. There is a need for a VBA program that generates the required number of worksheets from an exercise repository as a Word document. To do this, it was necessary to assess how many tasks the repository should contain and what its structure should be. The document generation program allows each candidate to receive a unique test worksheet, which, may include common exercises, but this can be kept at a sufficiently low level depending on the size of the exercise repository.*
Keywords: *Microsoft Word, online examination, programming, VBA*

1. INTRODUCTION

Education faces new challenges due to the COVID-19 virus situation. This has not only changed the technique of education, but also the method of accountability. Previously, the examination was conducted in a personal presence under controlled conditions. During the online education, it had to be solved for the examinee to be able to present his or her knowledge in his or her home by taking advantage of the possibilities provided by digital technology and the Internet. In individual, oral exams, the easiest way to prevent cheating is to use a webcam. However, in the case of a written test, the use of camera surveillance is limited, especially when it is required in large numbers at the same time. The ingenuity of the students taking the tests can surpass all our imaginations when they want to meet the requirements without learning and knowledge. If the same test worksheet containing the same exercises is solved by the participants at the same time, they can easily pass the solution to each other. For this reason,

it is of primary importance to create unique worksheets for each student. However, these can only be made manually in limited quantities. Therefore, it became necessary to automate the creation of the documents containing the exercises, which we can do by writing a program after well-planned preparations. The Microsoft Office suite provides programming with Visual Basic for Applications (VBA). With the completed program we can generate Word documents in the required amount. Thus, each candidate gets a Word document containing only his or her series of exercises. Naturally, this method cannot ensure that the system is circumvented, but we can take a serious step to limit cheating with the unique worksheets.

2. MATERIAL AND METHOD

The test worksheets containing the exercises generated as a result of the work are created as Word documents. Microsoft Office provides all the necessary tools:

- Create an exercise repository as a Word document.
- VBA program development.

The VBA programming environment ensures that the execution of monotonically repetitive tasks can be accomplished by writing a program (Zimmerman, 1996; McGrath, 2019). VBA programs are most commonly used in Microsoft Excel spreadsheets (Hampel, 2017; Hampel, 2018), but it can also be used in Word as well, as in this case.

With simple calculations, we can determine how many exercises we need in the repository to have a sufficiently low number of identical exercises in two worksheets. To do this, we needed to determine the distribution (1) and the expected value (2) of the binomially distributed probability variables (Obádovics, 2020) from the field of probability calculation.

$$P(X = k) = \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k}, (k = 0, \dots, n) \quad (1)$$

$$E(X) = n \cdot p \quad (2)$$

where:

X: probability variable, the number of identical exercises in two worksheets

$P(X = k)$: the probability of the event

p: the parameter of the distribution, the probability of identical exercises

n: the parameter of the distribution, the number of exercises in a worksheet

3. RESULTS AND EVALUATION

3.1 PLANNING

First, to generate the worksheet with the exercises, we had to plan every detail: first of all, how many exercises should one worksheet contain. This is affected by:

- the number of exercise-types;
- the length of time allowed to solve the test.

Based on the above, by including eight exercises, it is possible for at least half of the exercise types to appear in a worksheet, while the timeframe of the solution is eighty minutes. The determination of the timeframe stems from several years of experience as an instructor and examiner, according to which, for the student, it is sufficient to ensure twice the duration of the time required for the tutor. Thus, an average of ten minutes per exercise is sufficient for the candidates.

The next issue is the size of the exercise repository. This is important because it is less likely to have two worksheets with identical exercises that is randomly generated from a repository with multiple exercises. For one exercise, four different options proved to be sufficient. Using the tools of probability calculation, a binomially distributed probability variable (X) denotes the number of identical exercises in two worksheets. The value of the parameters of the distribution is:

$$n = 8$$
$$p = \frac{1}{4}$$

The first parameter is the number of exercises in a worksheet (number of independent selections), while the second parameter is the probability of a selected exercise identical with an exercise within another worksheet. The following results were obtained:

- The probability of a complete match of two test worksheets: (3)

$$P(X = 8) = \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k} = \binom{8}{8} \cdot \left(\frac{1}{4}\right)^8 \cdot \left(1 - \frac{1}{4}\right)^{8-8} = \frac{1}{65{,}536} \quad (3)$$

- The expected value of the number of identical exercises in worksheets: (4)

$$E(X) = n \cdot p = 8 \cdot \frac{1}{4} = 2 \quad (4)$$

So, on average, two exercises would be the same in worksheets containing eight exercises. It can also be read from the above result that 65,536 different test worksheets can be compiled, which differ from each other in at least one exercise.

Based on the above, the test repository contains 8 × 4 = 32 exercises. However, if the 4-4 alternative exercises were of the same type, only eight exercise types could have appeared in the exam. Since the goal is to account for more than that (15 types), the importance of each type had to be taken into account. To make sure the most important exercise types are definitely on a worksheet, 4-4 of them are needed in the repository, while fewer are required of the less important ones.

3.2 THE STRUCTURE OF THE TEST REPOSITORY

The repository contains four exercises in groups of eight. One exercise is added to a test worksheet from each group. Since the repository is a Word document, we also need to solve the that the program that generates the worksheets, need to interpret the location of an exercise in the document. The easiest way to do this is to form a single paragraph for each exercise, as the paragraphs can be identified by their serial number in the VBA programming language. However, when writing the exercise, it is often impossible to use a one-paragraph layout. For this reason, each exercise can be specified in a text box. Then we get a repository with a structure that consists of 32 paragraphs, and each paragraph contains a text box. The exercise in the text box can contain any number of paragraphs.

3.3 THE TEST WORKSHEET GENERATOR PROGRAM

The task of the generator program is to create test worksheets in the desired number of cases by randomly selecting one exercise from all four groups in the repository.

The VBA program is as follows:

```
Public Sub program()
db = Inputbox("Number of test worksheets needed")
  For i = 1 To db
    Documents.Add
    For j = 1 To 8
      p = (j - 1) * 4 + Int(Rnd * 4) + 1
      Documents("F32.docx").Paragraphs(p).Range.Copy
      Documents(1).Paragraphs(j).Range.Paste
```

```
        Documents(1).Paragraphs.Add
      Next j
      fn = "Test worksheet" & i
      Documents(1).SaveAs2 "h:\Exam\" & fn
    Next i
  End Sub
```

At first, the program asks for the number of test worksheets to be generated in the db variable. The For i… Next i loop then ensures that the instructions that create the document in a task queue are executed again in the required number (pcs).

Steps to create a worksheet:

- Create a new document (Documents.Add).
- Copy an exercise from all eight exercise groups to the new document.
    - Select one of the four exercises randomly from the group and define which paragraph is this the repository.
    - Copy the selected paragraph to the clipboard (Copy).
    - Paste the contents of the clipboard into the generated new document (Paste).
    - Add another paragraph to the generated document for the next exercise (Paragraphs.Add).
- Generate filename for saving (fn).
- Save the generated document (SaveAs2).

The end of the document filenames will contain their serial number (*i*) and the document files will be saved in the Exam folder.

4. CONCLUSIONS

With Visual Basic for Applications, we can easily create any number of test worksheets based on an exercise repository with little programming knowledge. In this way, we can ensure that each student is given a unique set of exercises during the online training and remote examination caused by the virus situation, so that we can keep low the possibility of students helping one another.

References

Hampel, György. 2017. “Excel VBA alkalmazása egy biometriai esettanulmány példáján bemutatva. [Using Excel VBA in a biometric case study.]”, Jelenkori társadalmi és gazdasági folyamatok [Contemporary Social and Economic Processes] 12 (4): 35–40.

Hampel, György. 2018. “Egymintás t-próba programozható kialakítása Excel VBA környezetben. [Designing programmable one-sample t-test in Excel VBA environment.]” Jelenkori társadalmi és gazdasági folyamatok [Contemporary Social and Economic Processes] 13 (3–4): 169–175.

McGrath, Mike. 2019. Excel VBA in easy steps. 3rd edition. Warwick: In Easy Steps Limited.

Obádovics, J. Gyula. 2020. Valószínűségszámítás és matematikai statisztika [Probability and mathematical statistics]. Budapest: Scolar Kiadó.

Zimmerman, M. W. 1996. Microsoft Office 97 Visual Basic Programmer’s Guide. Washington: MicrosoftPress.

NOTES ON THE AUTHORS

Zoltan FABULYA (Dr. Ph.D.) associate professor; University of Szeged Faculty of Engineering, Institute of Engineering Management and Economics, 7 Mars tér, H-6724 Szeged, Hungary; phone number: (+36) 62 546 000; email: fabulya@mk.u-szeged.hu. Research areas: mathematical modelling, Excel and Access application development in VBA. Main courses: Qualifications and university degree: Mechanical engineer in food industry (BSc), Computer programmer mathematician (BSc), Software designer mathematician (MSc), Management and organizational science (PhD). Teaching activity: Mathematics, Mathematical statistics, Biometry, Informatics, Decision support systems, Programming, Web developing.