Proceedings of the 9^{th} International Conference on Applied Informatics
Eger, Hungary, January 29–February 1, 2014. Vol. 1. pp. 361–368
doi: 10.14794/ICAI.9.2014.1.361

Analysis of algorithms and data modelling tasks of the final exam in Hungary

Viktória Lakó

Faculty of Informatics, University of Eötvös Loránd
viktoria.lako@gmail.com

Abstract

The aim of this paper is to study the algorithms and data modelling parts of the Hungarian final exam. In 2005, the final exams were reformed in Hungary. The research includes the final exams from May 2005 to May 2012, focusing on methodical and algorithmic orientated methods of the teaching of programming. The paper examines the presence of programming theorems, data types, and data structures in the tasks, and collects extra ideas which can make the tasks easier to solve. Following this examination, the results are evaluated on the aspects of the proportion of programming theorems, whether it is appropriate or not, and how the tasks of this exam could be improved. Moreover, this article can be useful for Informatics teachers as well, as they get a comprehensive picture of the programming task of the final exam, so they can prepare their students better.

Keywords: programming, computer science education, secondary education, programming theorems

MSC: 68-W

1. Introduction

In 2005, the final exams were reformed in Hungary. Since then final exams can be taken on 2 levels: medium and advanced level. Students have to take exams of at least 5 subjects. Hungarian Literature and Grammar, Maths, History and a chosen Foreign Language are compulsory. The fifth subject can be any other subject, for example Informatics. [1]

Informatics final exams consist of one practical part, with 4 tasks, which have to be solved on a computer, and an oral part, where students get a topic about which they have to speak 15-20 minutes. On both levels, the practical part is worth 120 points and the oral part 30 points. The 2 levels are different in the recommended time of the practical exam. On the medium level, the maximum time of the practical part is 180 minutes, this is 240 minutes on the advanced level. [2]

In the present exam system, the advanced level exam has to contain medium level task types, and can be completed with extra types of task. Exams can not consist of more than 2 parts. The same knowledge can be expected just in one of the parts. Algorithms and data modelling skills are expected just in the advanced level practical part. [3]

2. Analysis of algorithms and data modelling tasks

The most dominant part of the advanced level exam is the algorithms and data modelling task. The recommended time for this task is 90 minutes of the available 240 minutes. 45 points can be obtained for it of the maximum 120 points of the practical part. [2]

2.1. Algorithms and data modelling tasks

In this task, a problem of a practical situation has to be understood, and it has to be solved by writing a computer program. This program has to be implemented in a chosen programming language,the available languages being Pascal, C#, C++, Visual Basic, Python, Java [4]. The main problem is divided into 5 to 9 subproblems. The input data are always in text files, and the output usually has to be written in text files, as well.

2.2. Tasks implementation

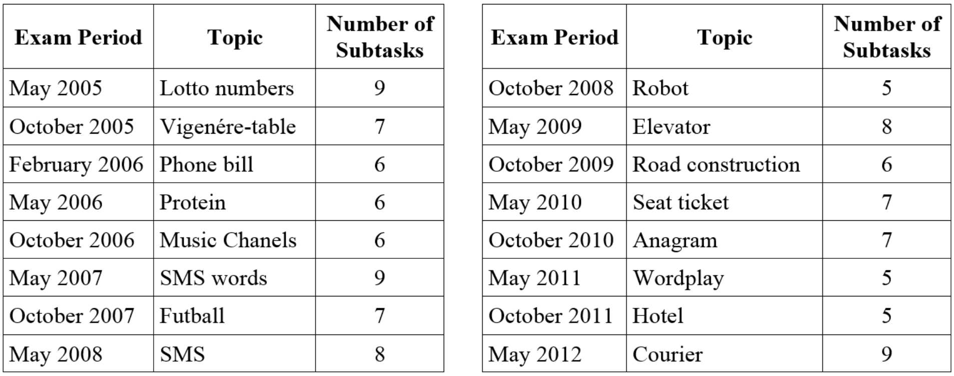
This paper shows the examination of the 16 final exams between May 2005 and May 2012. During the examination, the algorithms and data modelling problems were implemented in Python 3. The applied development environment was IDLE 3.

Python was chosen due to the following reasons: Python is a professional programming language, its syntax is simple and clear, so pupils can easily understand and learn it. Moreover, according to a recent study of the Royal Society of UK, Python is the most appropriate programming language in secondary education.[5]

The implementations of the tasks are available at the following website:
`http://pastebin.com/u/lviktoria`

2.3. Topic of tasks

Table 1 shows the topics of algorithms and data modelling tasks between 2005 and 2012. The topics can be classified into four main groups. In the first group, there are the tasks connected to phones (Phone bill, SMS words, SMS). The tasks of the second group are related to a company (Music Channels, Football, Elevator, Road construction, Seat ticket, Hotel, Courier). The tasks of the third one are connected to a discipline (Vigenére-table, Protein, Robot), the other tasks are assigned to the fourth group.



Exam Period	Topic	Number of Subtasks
May 2005	Lotto numbers	9
October 2005	Vigenére-table	7
February 2006	Phone bill	6
May 2006	Protein	6
October 2006	Music Chanels	6
May 2007	SMS words	9
October 2007	Futball	7
May 2008	SMS	8
October 2008	Robot	5
May 2009	Elevator	8
October 2009	Road construction	6
May 2010	Seat ticket	7
October 2010	Anagram	7
May 2011	Wordplay	5
October 2011	Hotel	5
May 2012	Courier	9

2.4. An example of the examination method

The tasks were examined in several aspects, based on questions such as Which data structure is the most appropriate for the input data and the solution of the problem? Which programming patterns could be used for the solutions of the subproblems? Are there any extra ideas which can make the problem easier to solve? Additionally, the main idea was to create a well "teachable" implementation.

The aspects of the examination were the following:

- **Programming Patterns:** Decision, Searching, Selection, Counting, Sequential Computing, Maximum Selection, Copying, Assortment, Classification, Sorting [6]
- **Data Types:** Integer, Float, String, Boolean
- **Data Structures:** List (1D), List (2D), Dictionary, Set
- **Other Skills:** Write to File, Read from File, Join Programming Patterns, Random numbers, Length of String or List, Digits, Sub-strings, Functions

An example: May 2011: Wordplay[1]

Each programming task starts with a long statement about a common situation:

> *„Ladderwords" is a game when you get the middle of a word and have to add one letter to the beginning and one letter to the end of this string to create a real word.*

[1] The task is available at:
http://www.oktatas.hu/kozneveles/erettsegi/feladatsorok/emelt_szint_2011tavasz

> *For example: OUN → C**OUN**T, R**OUN**D, F**OUN**D, M**OUN**T*
>
> *The supplied input file contains a unique word on each line.. Each word consists of only lower case Latin letters, and their length is shorter than 20 characters.*
>
> *Create a program code which solves the following problems!*

This long statement is followed by the subtasks:

> *#1 Read a text input, and decide whether it contains vowels or not.*

In the implementation of this problem a string variable is needed to store the text. For the storage of the vowels a list or set date structure could be used. To get the answer for this question, the decision programming pattern has to be applied. The answer will be a boolean value.

> *#2 Read the input file and write out the longest word and its length .*

To store the words of the input data a list is needed. Students have to know how to read data from file. To determine the longest word, the maximum selection pattern has to be used, and one has to obtain the length of a string.

> *#3 Hungarian words usually contain less vowels than consonants. Output the words from the file which contain more vowels than consonants Separate the words with a space. After listing the words, provide the following data as well: number of words found, number of words in the text file, the percentage of the found words in relation to all the words.*
>
> *Output format has to be the following:*
>
> *number of found words / number of all words : percentage of found words (with 2 decimals)*

To solve this problem there a function is needed which can count the number of vowels in a string (counting program pattern). After that, the words which contain more vowels than the half of the length of the word have to be assorted (assortment programming pattern). To output the answer in the required format the number of floats digits has to be handled.

> *#4 Collect the five-letter words into an array or a list. Read a three-letter word, and then write out those five-letter words from the list which can be made into a ladderword with the three-letter word.*

The five-letter words can be assorted to a list. Then from these five-letter words those words have to be assorted whose three middle letters coincides with the input word. In this subtask, the assortment programming pattern has to be used. To get the middle of the words, sub-string manipulation is required.

> *#5 Classify five-letter words, words are in the same group if they are ladderwords of the same three-letter string. You have to write out just those classes which contain more than one element. Each word has to be on a new line, and between the groups there has to be an extra line.*

To answer this question, the five-letter words have to be classified into groups. To handle these classes, the appropriate data structure is the dictionary. The keys of this dictionary are the different three-letter words from the middle of the five-letter words. The items of the dictionary are the list of the five-letter words which are in the same group. To get the middle of the words, sub-string manipulation is needed, and the outcome has to be written into a file.

2.5. The analysis

During the examination each exam task was implemented. This implementation made it possible toidentify the appropriate data types, data structures, programming patterns, and other pieces of knowledge which have to be known to solve the problems.

Presence of types

The research shows that the most common data types are string and integer (Figure 1) in the exams. In some cases, float and boolean variables had to be used, as well.

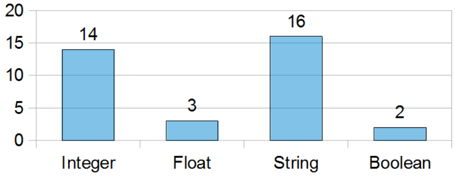


Figure 1

Presence of data structures

The basic data structure of the problems was usually a list. All tasks could be solved with using only the list data structure. In some cases, it was useful to apply sets, as you can see in the previous example. In the case of more complicated problems where one object has more properties that has to be manipulated, it could be useful to operate with a dictionary. Dictionaries are most useful when dealing with the classification of elements of a list (as in the 5^{th} task of the previous example). (Figure 2)

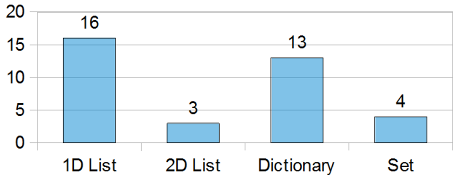


Figure 2

Presence of other skills

There are some pieces of knowledge which are expected at every exam, such as handling files and combining programming patterns. Often, one has to determine the length of a string or a list. This is useful knowledge as well. Problems could usually be solved without using any functions, but sometimes using a function makes the task easier to solve. (Figure 3)

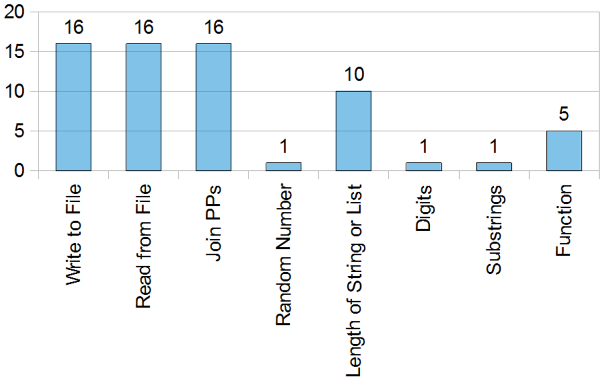


Figure 3

Presence of the programming theorems

The programming patterns occurred several times (Figure 4). Half of the listed programming patterns were used in every exam task, sometimes more than once. The other half of the patterns appeared in just a few cases. Some programming patterns had to be usually used joint with others.

Comparison of tasks

Figure 5 shows the tendencies of the 16 final exams. The columns are composed of five parts: the number of the different programming theorems, the number of subproblems needing the joining of programming patterns, the number of the different variable types and the number of other knowledge items used.

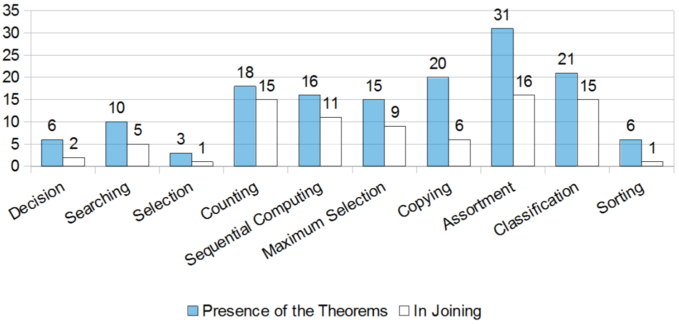


Figure 4

The sum of these 4 numbers might indicate the difficulty of the tasks. Similarly, we can define the difficulty of subtasks as well, these difficulty numbers are correlated with the points of the sub-problems. As my next research, the results of students could be compared to these difficulty numbers.

The difficulty of the tasks were similar during the examined period, but it can be noticed that tasks became a bit more complicated during the years.

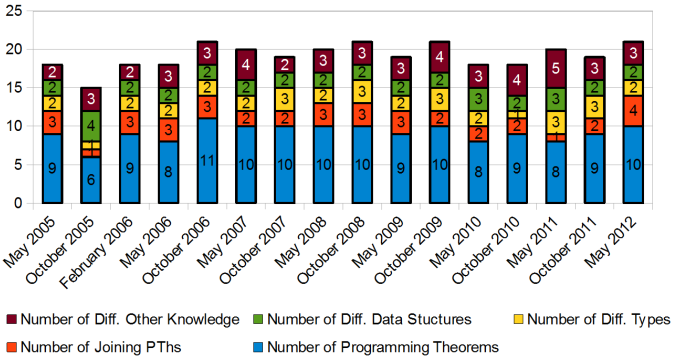


Figure 5

3. Conclusion

The goal of algorithms and data modelling tasks should be to measure the problem solving, data modelling and algorithm creating skills of the students. The research shows that this goal was mainly achieved between 2005 and 2012. Students have to understand a common situation, choose appropriate data structures and types, recognise the applicable programming pattern and to transform it to fit the task. Each exam contained both simple and complex sub-problems, so tasks could differentiate between students based on their programming skills.

The range of topics are wide, but the topics of the tasks are sometimes outdated, in the future it could be a goal to find topics which are closer to the interest of students.

The solution of the problem statements are usually very similar. If students solve four or five of these tasks, they won't get any surprise at the exam. It could be a development opportunity to form such questions which require wider usage of the programming patterns in one problem.

References

[1] 100/1997. (VI. 13.) Korm. rendelet az érettségi vizsga vizsgaszabályzatának kiadásáról (100/ 1997 Decree about the Rules of Final Exams), (2012)Available at: `http://net.jogtar.hu/jr/gen/hjegy\_doc.cgi?docid=99700100.KOR` (13 February 2014)

[2] INFORMATIKA II. A VIZSGA LEÍRÁSA (Description of School Leaving Exam in Informatics), (2012)Avaiable at: `http://www.oktatas.hu/pub\_bin/dload/kozoktatas/erettsegi/vizsgakovetelmenyek2012/informatika\_vl.pdf` (10 January 2014)

[3] INFORMATIKA I. RÉSZLETES ÉRETTSÉGIVIZSGA-KÖVETELMÉNY (Detailed Requirement of School Leaving Exam in Informatics), (2012)Avaiable at: `http://www.oktatas.hu/pub\_bin/dload/kozoktatas/erettsegi/vizsgakovetelmenyek2012/informatika\_vk.pdf` (10 January 2014)

[4] Szoftverlista az emelt szintű érettségi vizsgákhoz (Software List of Advanced Level School Leaving Exam in Informatics), (2012)Avaiable at: `http://www.oktatas.hu/bin/content/dload/erettsegi/nyilvanos\_anyagok\_2013majus/info\_emelt\_szoftverlista\_2013maj.pdf` (10 January 2014)

[5] The Royal Society: Shut down or restart? The way forward for computing in UK schools. The Royal Society (2012)

[6] Szlávi P. and Zsakó L.: Módszeres programozás: Programozási tételek (Systematic programming: Programming patterns), ELTE TTK Department of General Computer Science, 1993.