

INFORMATIKA ANGOL NYELVEN

EMELT SZINTŰ GYAKORLATI VIZSGA

2013. május 13. 8:00

A gyakorlati vizsga időtartama: 240 perc

Beadott dokumentumok	
Piszkozati pótlapok száma	
Beadott fájlok száma	

A beadott fájlok neve

**EMBERI ERŐFORRÁSOK
MINISZTERIUMA**

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Important information

You have **240 minutes** to solve the practical exercises.

Devices allowed for the exam: computer assigned to the student, paper, pen, pencil, ruler, sealed notepaper.

You can take **notes** on the internal sides of the exercise sheet and the notepaper, these should be submitted at the end of the exam but their content will not be evaluated.

The exercises **can be solved in any order**.

Please pay attention to **frequent saving** (every 10 minutes); it is suggested that you save your work every time you start a new exercise.

You should save your exam work in the **exam directory** that **corresponds to the identification number** found on the exercise sheet. Check that the directory that corresponds to the number found on the exercise sheet is accessible; if it is not accessible, notify the supervising teacher at the beginning of the exam.

Save your works **in the exam directory** and at the end of the exam **check** that every solution is in the given directory because only those solutions can be evaluated. Check that the files to be submitted are readable, because files that can not be opened will not be evaluated.

If you solve the database management exercise with LibreOffice Base, you should submit the SQL commands describing update queries either as part of the LibreOffice Base database file or as a separate text file. If you submit them as a text file, the name of the text file should refer to its contents clearly (e.g. *SQL_commands.txt*) and the required query name should be displayed next to the command.

The submitted program can be evaluated only if the candidate created the source file(s) that correspond to the chosen programming environment in the exam directory and contain(s) the source codes that belong to the solution of the exercise parts.

The **source files** can be found in the exam directory.

In the case of programs that do not support setting dimensions in cm, use the conversion 1 cm = 40 px.

It is suggested that you **read through** the exercises first and then solve the individual exercise parts one by one.

If your computer has **technical problems**, indicate it to the supervising teacher. The fact of indication and the observed problem will be recorded. The lost time will be added to the duration of the exam. If the problem is not of computer nature, the examiner should take the description of the case in the record into consideration. (The system administrator cannot help the candidate with the solution of the exercises.)

At the end of the exam you should indicate **the number and the name of files created and submitted by you and located in the exam directory and its subdirectories** on the first page of the exam document. When finishing the exam, do not leave the room until you have done so and have shown it to the supervising teacher.

Please indicate the operating system and the programming environment you work with.

Operating system: ☐ Windows ☐ Linux ☐ MacOS X

Programming environment:

- | | | |
|-----------------------------------|------------------------------|---|
| <input type="radio"/> FreePascal | <input type="radio"/> GCC | <input type="radio"/> Visual Studio 2008 Professional |
| <input type="radio"/> Lazarus 0.9 | <input type="radio"/> Perl 5 | <input type="radio"/> Visual C# 2010 Express |
| <input type="radio"/> JAVA SE | <input type="radio"/> Python | <input type="radio"/> Visual Basic 2010 Express |
| <input type="radio"/> _____ | <input type="radio"/> _____ | <input type="radio"/> _____ |

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

1. Menu

In restaurants the majority of the offers is constant, which is supplemented with local specialties from time to time. In the Fisherman Restaurant fish meals are changed every week.

File *dishsource.txt*, which is UTF-8 encoded, contains the names and the prices of the steadily available meals. Besides these the restaurant offers three fish meals that change from week to week. File *fishdishes.txt*, which is tagged by tabs, contains information on the fish meals of the menu of the week. The first line of the file contains the field names:

<i>fish1, price1</i>	the name and the price of the first fish dish
<i>fish2, price2</i>	the name and the price of the second fish dish
<i>fish3, price3</i>	the name and the price of the third fish dish

Create the menus five weeks in advance according to the given example and the description. Depending on settings, pages may not end at the same place as shown in the example. Do not use unnecessary spaces to lay out the text. The text should not contain empty paragraphs.


1. Create the menus of the week as a mail merge document using file *dishsource.txt* in the default format of the program. Save the source document as *menu* in the default format of the word processor.
2. In the document the page size is A4, the left and right margins are 2.2 cm, the top margin is 4 cm and the bottom margin is 3 cm.
3. Find and replace parentheses “(” together with the spaces before them with line breaks and words “*forint*” with abbreviation “HUF” in the whole document. Delete the “)” parentheses.
4. The font type of the text is Arial (Nimbus Sans) if not specified else. The font size is 11 points for the names and the prices of the dishes; 20 points for the dish categories and 26 points for the text in the header. The font size of the text in the footer and the dish descriptions according to the example is 8 points.
5. In the document the line spacing is 1.5 lines, the spacing before the paragraphs is 0 points and the spacing after them is 6 points if the description does not specify else. The first line of the paragraphs should start at the margin, and the other lines should have an indent of 0.5 cm.
6. Set the prices of the dishes to 16 cm with a right tab.
7. For the aesthetic appearance of the menus border the pages according to the example. The width of the border should be between 2.5 points and 3.5 points.
8. Arrange the header with a table that has one row, three columns and no border. The first and third cells of the table should be of equal width. The contents of the three cells should be aligned centered in vertical direction and right, centered and left respectively in horizontal direction. Type in words “Fisherman” and „Restaurant” into the left and right cells. Insert picture *fishlogo.png* into the middle cell and set its height to 2.2 cm keeping the aspect ratio.
9. Display picture *cornerpiece.png* and its reflected copy on each page according to the example; set their height to 2 cm. The lines on the edge of the pictures should overlap with the margins.


--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

10. The font type of the text in the header and the footer is Lucida Calligraphy (URW Chancery). Replace the text of the footer from the braces at the end of the source text and delete the braces. Align the text centered horizontally.
11. The font style of the dish category names shown in the example is small caps and their background is gray, there is a spacing of 6 points before and after them.
12. Under dish category name “***Fish dishes – offer of the week***” display the name and the price of the three fish dishes of the week by inserting the given data fields. Pay attention to their style corresponding to the style of the other dishes.
13. Save the completed mail merge document merged into a file as *menus_of_the_week* in the default format of the word processor. (If you use an OpenOffice.org program, you can save numbered copies as well.)

30 marks

Example:

	
STARTERS	
Smoked trout on a bed of saled	1340 HUF
Delicious green saled in balsamic vinegar, with roasted walnut	
Mangalitsa sausage coated in potatoes	
SOUPS	
Goose soup with goose liver dumplings	
Sour bean soup with sausage	
Cabbage soup with boletus mushroom and smoked meat	
Potato cream soup with leek	
FISH DISHES - OFFER OF THE WEEK	
Fish soup from carp Szeged style	
Catfish stew	
Zander fillets fried in bread crumbs	
MAIN DISHES	
Trout fillet in sour cream with bacon and potatoes	
Catfish trunk in bacon with vegetable ragout and parsley potato	
Fried goose leg with cracklings and mashed potatoes as your g	
Grilled chicken leg with potatoes fried in oven and peasant saled	
Chicken breast Palóc style with corn in cream and rice	
Skewed turkey breast autumn style with potato puree, morello	
stuffed with walnut, prune and chestnut	
Chicken breast in cheese sauce with rice	
Turkey breast fried in cheese with roasted potatoes	
Dear Guests! A service fee of 10% will be added to the	

	
Lamb knuckle with potatoes roasted with onion and garlic compote 4580 HUF	
Ranger stew with potato pancake	
spicy ragout made of mixed meat with mushroom 2580 HUF	
Mangalitsa roast peasant style with braised cabbage and knedli 2340 HUF	
Stag stew in red wine with halusky 3580 HUF	
Beefsteak with boletus mushroom ragout and roasted potatoes 3580 HUF	
Vegetarian steak	
grilled aubergine fried with vegetable and cheese 1980 HUF	
King's plate of abundance	
Becks cutlet, fried goose leg, skewed chicken breast, fried mushroom, mixed garnish and home-made pickles 5580 HUF	
PASTA	
Brinzove halusky, that is, potato noodles with curded ewe-cheese 1380 HUF	
Potato pancake with spiced curded ewe-cheese 1160 HUF	
Halusky with pickled cabbage 1380 HUF	
SALADS	
Red cabbage saled 540 HUF	
Cucumber saled with sour cream 580 HUF	
Tomato saled with basil 640 HUF	
Home-made mixed pickles 680 HUF	
DESSERTS	
Pancake filled with walnut with chocolate sauce 1080 HUF	
Pancake filled with poppyseed with plum sauce 960 HUF	
Quince in red wine with caramel ice cream and white chocolate with rum 1100 HUF	
Chestnut parfait with forest sauce 620 HUF	
Dear Guests! A service fee of 10% will be added to the prices in the menu.	

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

2. Packages

The eggs sold in retail outlets should be marked per piece. The mark on the egg or the package should contain (among others) the category of the egg based on its size.

File `qualsource.txt` contains the data of the quality control sample of an egg sorting and packing machine. The sample contains the data of 100 eggs: the mass, the colour and the intactness determined by inspection with light passed through the egg.

Solve the following exercises using a spreadsheet processor.

During the solution take the followings into consideration.

- *You may perform auxiliary calculations to the right of column J.*
 - *Whenever possible, use a formula, function or reference in the solution to get the required results even if the base data are changed.*
 - *Wherever possible, use a function that can be copied flawlessly.*
 - *Perform the calculation so that correct results are obtained even if columns A:D are filled with the data of another sample to Row 101 of the table.*
 - *There are parts in the exercise that use the results from a previous question. If you could not solve the previous part completely, use its solution as it is, or enter a reasonable result and work on with it. This way you can receive marks for these exercise parts as well.*
1. Import text file `qualsource.txt`, which is a UTF-8 encoded text file tagged by tabs, into the spreadsheet processor starting from cell `A1`. Save your work as `inspection` in the default format of the spreadsheet processor.
 2. In column *Category* of the worksheet determine the category of the individual eggs based on their mass using the category table in range `F1:I5`.
 3. Under heading “**Quantity**” determine the number of eggs per category in the sample.
 4. In the cells next to inscriptions “**Number of intact eggs:**” and “**Number of damaged eggs:**” determine the number of intact and damaged eggs.
 5. We wish to investigate the data per colour, intactness and category. For this purpose enter a value into each cell of range `F17:H17` (for example: “brown”, “yes”, “M”).
 6. In the cell next to inscription “**Number of eggs:**” determine the number of eggs that meet all three properties given in the previous row.
 7. In the cell next to inscription “**Number of 6-egg boxes:**” calculate the number of boxes that can be filled fully with the eggs determined in the previous row, if one box can hold 6 eggs.
 8. Set that the numbers in the cells containing the number of eggs and boxes are displayed without decimal digits, with “pcs” unit.
 9. Border all cells containing data with a thin line. Highlight the auxiliary table giving category limits and its heading with a thick border. Do not set a border for the other cells.

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

10. All text in the first row should have bold font style. Set the direction, the layout and the alignment of the text according to the example. You do not have to align the text according to the example in the other rows of the table.

15 marks

Example:

	A	B	C	D	E	F	G	H	I
1	Mass (gram)	Colour	Intact	Category		Size	Lower limit (gram)	Upper limit (gram)	Marking
2	71	brown	yes	L		small	0	53	S
3	78	brown	yes	XL		medium	54	63	M
4	61	brown	yes	M		large	64	73	L
5	54	brown	yes	M		extra large	74	100	XL
6	53	brown	yes	S					
7	48	brown	yes	S					
8	68	white	yes	L					
9	59	brown	yes	M					
10	56	brown	no	M					
11	67	brown	yes	L					
12	71	brown	yes	L					
13	61	brown	yes	M					
14	40	brown	yes	S					
15	67	brown	yes	L					
16	44	brown	yes	S					
17	73	white	yes	L					
18	77	brown	yes	XL					
19	53	brown	yes	S					
20	73	brown	yes	L					
21	84	brown	yes	XL					
22	40	brown	yes	S					

Category		Quantity
S		25 pcs
M		35 pcs
L		25 pcs
XL		25 pcs

Number of intact eggs:	25 pcs
Number of damaged eggs:	15 pcs

Colour	Intact	Category
brown	yes	M
Number of eggs:		25 pcs
Number of 6-egg boxes:		15 pcs

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

3. Compulsory

Compulsory Student Library, which is located in Big city, is open between 8 a.m. and 4 p.m. on every day, even on holidays. As a pilot project, a lending system electronic in every aspect was introduced in July 2009. A remotely readable chip was installed on each reader's card and in the books, so the borrower only has to pass through the appropriate gate. The administration of the lending process is performed by the computer in the background. In this exercise we use the simplified form of the database managed there.

1. Create a new database with name *compulsory*. Import the data tables into the database as ***works***, ***copies***, ***lendings*** and ***students***. These are UTF-8 encoded text files tagged by tabs, their first line contains the field names.
2. After importing set the suitable data formats and keys.

Tables:

works (*id*, *author*, *title*, *grade*)

<i>id</i>	The identifier of the work (number), this is the key
<i>author</i>	The author of the work (text)
<i>title</i>	The title of the work (text)
<i>grade</i>	The grade where the work is a compulsory reading (number)

copies (*id*, *workid*, *price*, *purchase*)

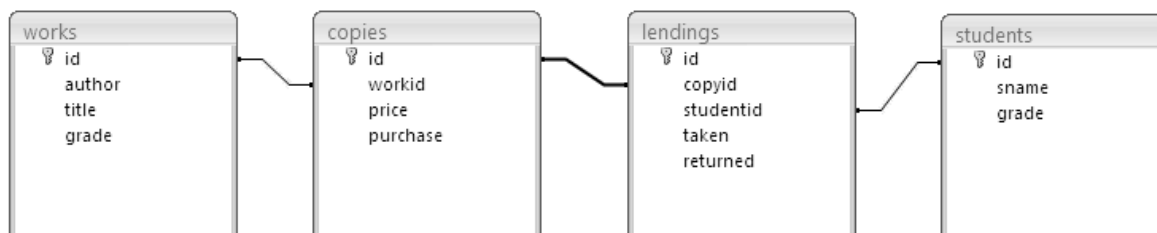
<i>id</i>	The identifier of the copy (number), this is the key
<i>workid</i>	The identifier of the work (number)
<i>price</i>	The purchase price of the copy (number)
<i>purchase</i>	The date of purchase of the copy (date)

lendings (*id*, *copyid*, *studentid*, *taken*, *back*)

<i>id</i>	The identifier of the lending (number), this is the key
<i>copyid</i>	The identifier of the borrowed copy (number)
<i>studentid</i>	The borrowing student's identifier (number)
<i>taken</i>	The first day of the lending (date)
<i>returned</i>	The last day of the lending (date), if the book was not returned, then it is empty

students (*id*, *sname*, *grade*)

<i>id</i>	The student's identifier (number), this is the key
<i>sname</i>	The student's name (text)
<i>grade</i>	The student's grade in the year under investigation (number)



--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Prepare the solution of the following exercises. Pay attention to displaying only the required fields and expressions in the solution, do not display additional fields. Save your solutions with the name given in brackets.

3. Create a query that lists the names of the students who have unreturned books. (**3not**)
4. List the titles of the works by “*Shakespeare*” compulsory in grades 10 and 11 using a query. (**4ws**)
5. Give the title and the price of the work by “*Voltaire*” that was purchased last using a query. (**5voltaire**)
6. Give the sum allocated to purchases and the number of copies purchased by the library per year using a query. (**6peryear**)
7. Create a query that lists the students and the works they have with them at the time of the opening on 2009.09.30. (**7whohasit**)
8. When somebody returns a book, the system sends an alarm immediately if the 28-calendar day lending time was exceeded. At the time of returning the book a fine equivalent to 3 percent of the purchase price of the book must be paid for each additional day beyond the day on which the lending time ends. (If a book was borrowed on the 1st and returned on the 29th then no fine is to be paid.) Create a query that indicates who paid a fine, when he/she paid it and how much fine was paid. You do not have to round the fine. (**8fine**)
9. Create a report that lists those who borrowed the work titled “*Az apostol*” and the date they took and returned the book grouped by copies. Group the data by the identifier of the copy. Display the first and the last days of the lending and the borrowing student’s name in the order of borrowing in the layout shown in the example. (You have to follow the example only with the text and the order of the fields but not with the format.) Prepare for the creation of the report with a query or a temporary table. (**9apostol**)

Az apostol

Copy	Taken on	Returned on	Borrower's name
420			
	2009.07.11.	2009.08.02.	Dudás Krisztián
	2009.08.09.	2009.08.12.	Kardos Ádám
	2009.08.14.	2009.09.09.	Hajas Attila

10. You have to list the names of the students who did not borrow the work titled “*Anna Karenina*” although it was recommended to their grade. In order to reach your aim, create two queries that – used as subqueries at the suitable place in the following SQL command – give the correct solution. (**10ak1, 10ak2**)

```
SELECT students.sname
FROM students
WHERE students.grade=(
    10ak1
)
AND students.id NOT IN (
    10ak2
);
```

30 marks

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

4. Numbers

In the Internet quiz “*Do you like numbers?*” the contestants get questions that should be answered with an integral number. The questions come from different topics (e.g. literature, mathematics, history, chemistry) and are worth 1 to 3 points depending on their difficulty. We know that the value of the answers for the questions is between 0 and 1 thousand million.

The organisers of the quiz store the questions in a data file. In the file each question is located in two lines. The first line contains the question and the second contains the correct answer, the points awarded for the correct answer and the naming of the topic separated by single spaces. The file does not contain letters with accents yet, e.g. instead of word “gyümölcsízű” word “gyumolcsizu” was entered into the file.

For example:

```
When was the Bane of Mohacs?  
1526 1 history
```

The question in the example is: When was the Bane of Mohacs? The correct answer is: 1526. The correct answer is worth 1 point and the question belongs into the topic of history.

The data file is only partially ready. Your task is to test this half-finished data file. The file contains at most 100 questions. It certainly contains questions from the topic of mathematics, history and geography, but there may be other topics as well.

Create a program that answers the following questions using the data in file *numberquest.txt*. Save the source code of the program as *numbers*. (You do not have to check the data of the read file and the validity of the user’s answers.)

Before displaying the result of exercises that require writing information on the screen display the number of the exercise part on the screen (for example: Exercise 3). If you request data from the user, also display the data type to be entered on the screen. Displays with or without accents are both accepted.

1. Read the data from file *numberquest.txt* and solve the following exercises using them.
2. How many questions does the data file contain? Display the answer on the screen.
3. Determine the number of mathematical questions in the data file and how many of these are worth 1, 2 and 3 points. Display the answer on the screen as a complete sentence.

For example:

```
The data file contains 20 mathematical questions, 10  
questions are worth 1 point(s), 6 questions are worth  
2 point(s), 4 questions are worth 3 point(s).
```

4. Find the range of the numerical values of the answers located in the file. Display the answer on the screen as a complete sentence.
5. Which topics are actually in the data file? Display the names of the topics on the screen so that each occurring topic is displayed exactly once.

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

6. Request the name of a topic from the user and then select a question from this topic at random. Upon selecting pay attention to each question belonging into the given topic having a chance. (You may assume that the user gave the name of an existing topic correctly.) Display the question, request an answer from the user and then give the points awarded for the answer. (An incorrect answer is worth 0 point.) If the answer was incorrect, give the correct answer as well. The dialogue should be displayed in the following format:

For example:

```
Which topic would you like to receive a question from?  
history  
When was the Bane of Mohacs? 1514  
The answer is worth 0 point(s).  
The correct answer is: 1526
```

7. Generate a test consisting of 10 questions at random so that no question appears more than once in it. (However, pay attention to each question read having a chance of being selected.) Write the test into file *testquests.txt* in the following format. (The first number is the point awarded for the correct answer; it is followed by the correct answer and the question, both separated by a single space.) At the end of the file write the total score that can be awarded for the test.

For example:

```
...  
1 1526 When was the Bane of Mohacs?  
...  
A total of 20 points can be awarded for the test.
```

45 marks

Sources:

1. Menu

http://g.virbcdn.com/_f/cdn_images/resize_640x640/3d/PageImage-477875-1676299-XL01_aff91cfbda74.jpg

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

	maximum mark	achieved mark	examiner's signature
Word processing, presentation, graphics, web page creation 1. Menu	30		
Spreadsheet processing 2. Packages	15		
Database management 3. Compulsory	30		
Algorithmisation, data modelling 4. Numbers	45		
Mark of the practical exam part	120		

Date:

	elért pontszám egész számra kerekítve/ achieved mark rounded to an integer	javító tanár aláírása/ examiner's signature	programba beírt egész pontszám/ mark written into program as an integer
Szövegszerkesztés, prezentáció, grafika, weblapkészítés/ Word processing, presentation, graphics, web page creation			
Táblázatkezelés/ Spreadsheet processing			
Adatbázis-kezelés/ Database management			
Algoritmizálás, adatmodellezés/ Algorithmisation, data modelling			

jegyző/registrar

Dátum/Date: