ÉRETTSÉGI VIZSGA ● 2013. október 18.

INFORMATIKA ANGOL NYELVEN

EMELT SZINTŰ GYAKORLATI VIZSGA

2013. október 18. 8:00

A gyakorlati vizsga időtartama: 240 perc

Beadott dokumentumok	
Piszkozati pótlapok száma	
Beadott fájlok száma	

A beadott fájlok neve

EMBERI ERŐFORRÁSOK MINISZTÉRIUMA

Important information

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

Devices allowed for the exam: computer assigned to the candidate, 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 with your identifier**. Check that the directory that corresponds with the code 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 can 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 giving dimensions in cm, you can use the conversion 1 cm = 40 px.

It is suggested that you **read through** the exercises first, 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 correcting teacher should take the description of the case in the record into consideration. (The system administrator can not help the candidate with the solution of the exercises.)

At the end of the exam you should indicate **the number and 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 you work with and the programming environment you use.

Operating system: ○ Windows ○ Linux ○ MacOS X

Programming environment:

○ FreePascal	○ GCC	○ Visual Studio 2008 Professional
○ Lazarus 0.9	○ Perl 5	○ Visual C# 2010 Express
○ JAVA SE	○ Python	○ Visual Basic 2010 Express
○ ________________	○ ________________	○ ____________________________

1. Hungarian poems

A publication is created from the selected poems written by Ady Endre, Arany János, Babits Mihály, Berzsenyi Dániel and Vörösmarty Mihály. Your task is to create this publication based on the given description and the example. The selected poems are in file *poems.rtf*. The pictures of the poets are in files *ady.jpg*, *arany.jpg*, *babits.jpg*, *berzsenyi.jpg* and *vorosmarty.jpg*, respectively. During your work pay attention to being able to insert an automatically generated table of contents in the document.

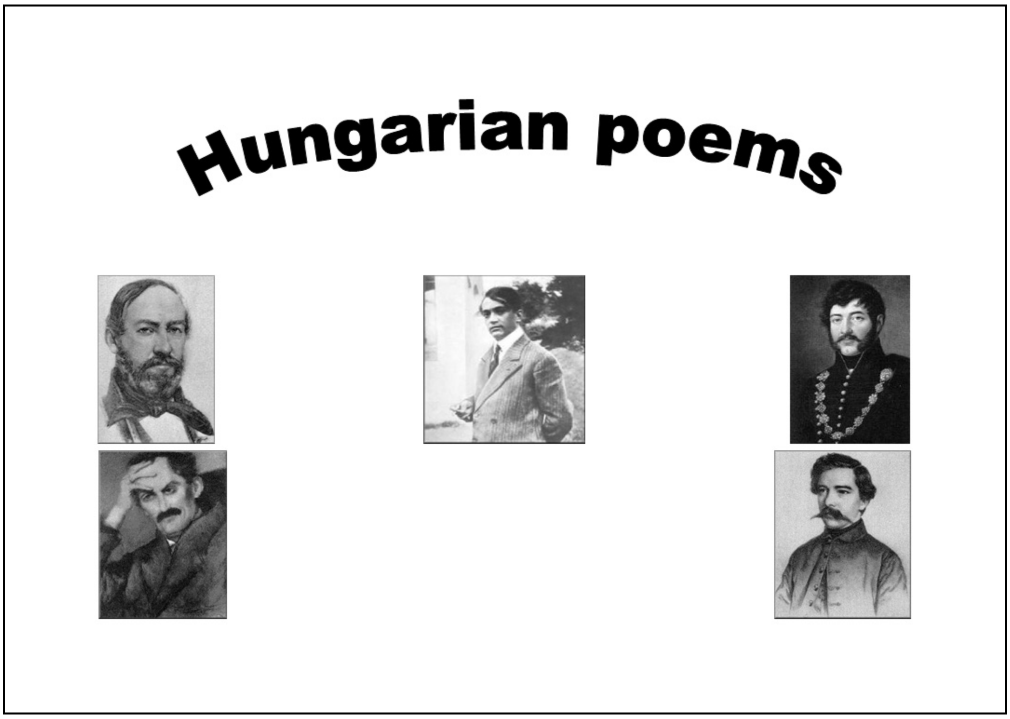
1. Create a cover page for the publication in a separate file named *covpage* in the default format of the word processor.
2. The orientation of the cover page is landscape, the page size is A4. The size of the margin is 2.8 cm on each side.
3. Insert the pictures of the poets based on the example. Set the height of the pictures to 5 cm keeping the aspect ratio. Ady Endre's picture should be centered both horizontally and vertically in the text area. Babits Mihály's and Arany János' pictures should be aligned to the bottom of the text area. Vörösmarty Mihály's and Berzsenyi Dániel's pictures should be aligned centered vertically in the text area. Vörösmarty's and Babits' pictures should be aligned to the left margin while Berzsenyi's and Arany's picture should be aligned to the right margin of the text area. Pay attention to keeping these settings even if the size of the margins is reduced.
4. Insert title "Hungarian Poems" above the pictures in the curved form and size shown in the example and align it centered horizontally. You may choose an arbitrary font type but pay attention to readability.
5. Create file *hungarianpoems* in a word processor using file *poems.rtf* in the default format of the program.
6. The orientation of the document is portrait, the page size is A4. Set the left and right margins to 4 cm and the top and bottom margins to 2.8 cm.
7. The font size of the header, the footer and the text of the poems is 13 points.
8. Create the header of the document according to the example, display text "Selected Hungarian Poems" in it.
9. Use page numbering aligned right in the footer of the document.
10. Presently each poem title is preceded by two line break characters and followed by three line break characters in the source file. Each verse is followed by two line breaks. Your task is to place every poet's name, poem title and verse in a separate paragraph. After formatting the text should not contain unnecessary spaces, line breaks and empty paragraphs.
11. The text should be displayed using font type Times New Roman (Nimbus Roman) throughout the document. The paragraphs should be preceded by a spacing of 0 points and followed by a spacing of 12 points. Use single line spacing for the whole text.

The exercise continues on the next page.

12. Format the lines containing the poets' name, their date of birth and death according to the example. The font size is 20 points and the font style is bold, the years should be displayed in italics. Insert the poets' pictures, set the height of the pictures to 3 cm keeping the aspect ratio. Place the pictures according to the example.
13. Set that the lines containing the poets' names are definitely the first lines on a given page.
14. Set the poem titles to a bold font of size 16 points. Pay attention to the poem titles and the first paragraphs of the poems being displayed on the same page throughout the document.
15. Set that no verse gets on two pages.
16. Insert an automatically created table of contents on a separate page according to the style of the sample. (If the font type of the table of contents is not the same as the font type of the text, then you do not have to change it, you can use default setting.) The table of contents should be separated from the poems, on the last page; if the word processor used by you can insert it only at the beginning of the document, insert it there.

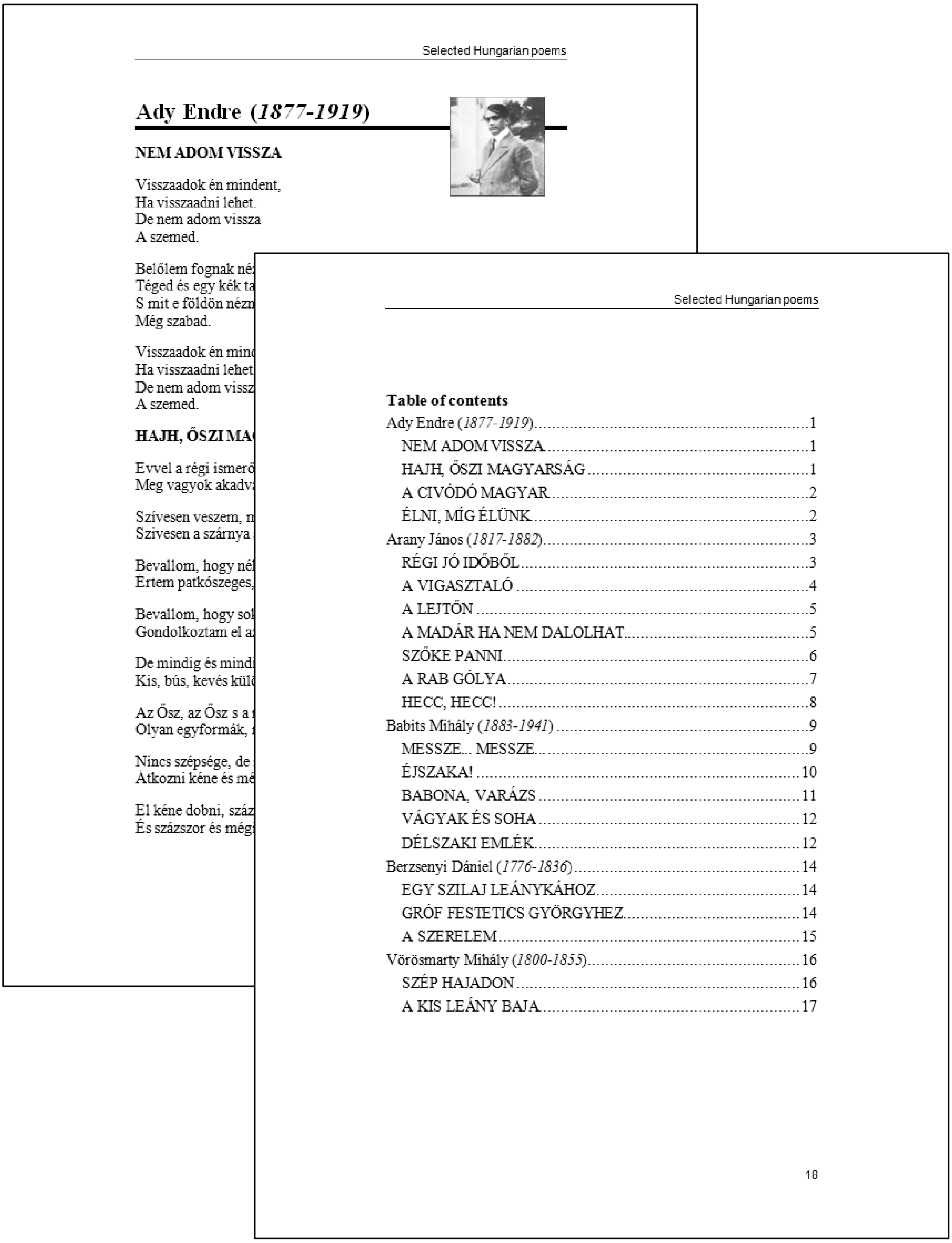
30 marks

Example:



Hungarian poems

Example for the Hungarian poems exercise:



Selected Hungarian poems

Ady Endre (*1877-1919*)

NEM ADOM VISSZA

Visszaadok én mindent,
Ha visszaadni lehet.
De nem adom vissza
A szemed.

Belőlem fognak né
Téged és egy kék ta
S mit e földön nézn
Még szabad.

Visszaadok én min
Ha visszaadni lehet
De nem adom vissz
A szemed.

HAJH, ŐSZI MA

Evvel a régi ismerő
Meg vagyok akadv

Szívesen veszem, n
Szivesen a szárnya

Bevallom, hogy né
Értem patkószeges,

Bevallom, hogy so
Gondolkoztam el a

De mindig és mind
Kis, bús, kevés kül

Az Ősz, az Ősz s a
Olyan egyformák,

Nincs szépsége, de
Átkozni kéne és mé

El kéne dobni, szá
És százszor és még

Selected Hungarian poems

Table of contents

18

Source:

http://mek.oszk.hu/

2. Test evaluation

In a survey twenty people filled in a test page. The asked questions were classified into three groups and they were investigated from aspects A and B within the group. The aggregation and the evaluation of the tests have already started. In the process the answers were aggregated per group and were classified into low, average and high categories. In the following exercises you have to work with these data. The data are in file `testdata.txt`, which is UTF-8 encoded and tagged by tabs.

Solve the following exercises using a spreadsheet processor. Perform the formatting operations not mentioned in the worded description based on the example.

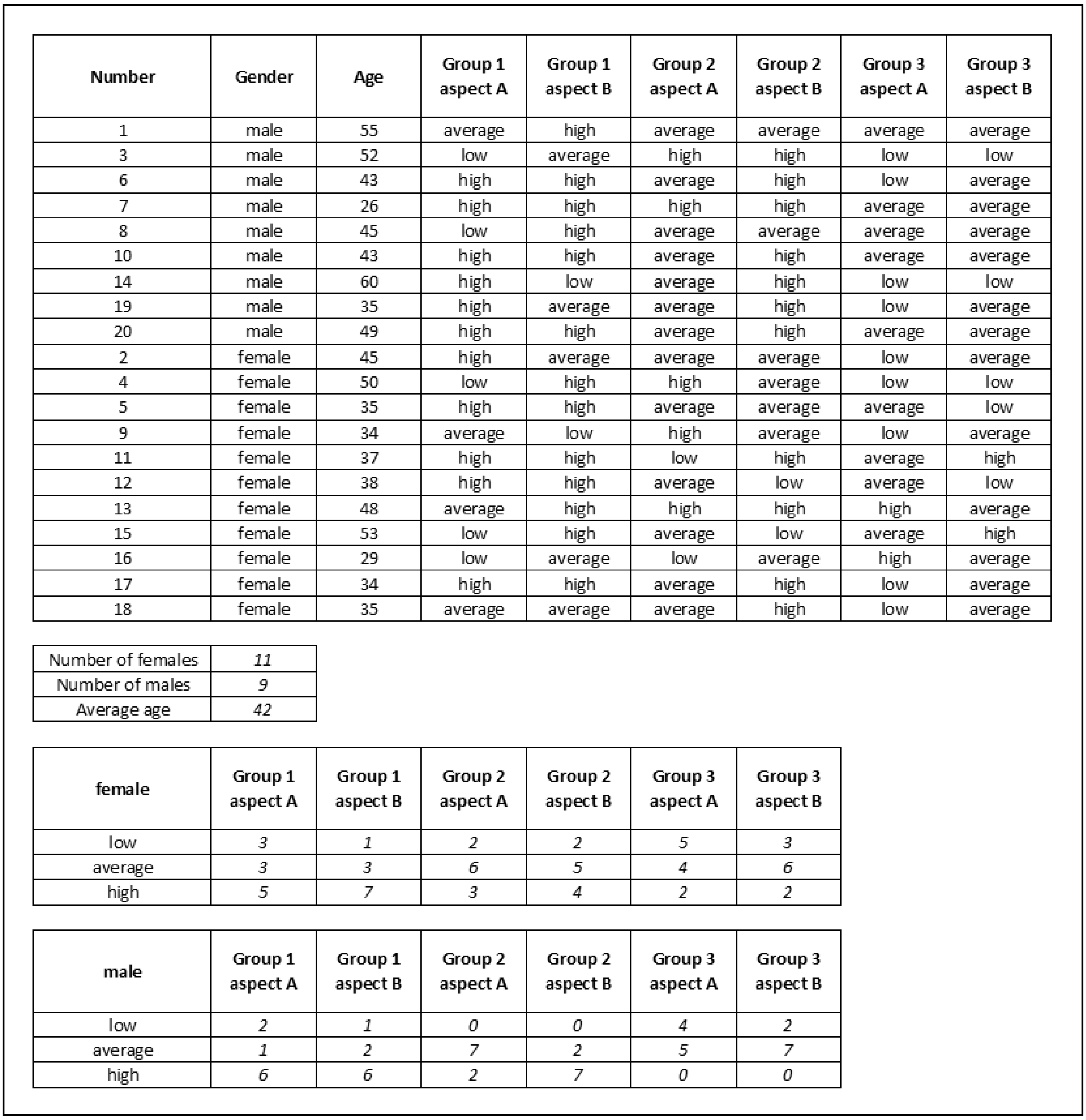
During the solution take the followings into consideration.

- *Whenever possible, use a formula, function or reference in the solution.*
- *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 instead of a formula resulting in a number enter integers between "1" and "7" and work on with these values. This way you can receive marks for these exercise parts as well.*

1. Import file `testdata.txt` into the spreadsheet processor starting from cell *A1* and save it as `processed` in the default format of the spreadsheet processor.
2. In cells *B23* and *B24* determine the number of males and females filling in the test using a function.
3. In cell *B25* determine the average age of those who filled in the test rounded to an integer using a function.
4. Sort the test data so that the data of males are at the top and the data of females are below them. In the following parts of the exercise you can make use of the ranges where males' and females' data are located.
5. In range *B28:G30* determine the number of answers given by females that belong into the low, average and high categories in each group of the test questions using a function. Use a formula that can be copied flawlessly within the given range in your solution.
6. In range *B33:G35* process the answers given by males in a similar way.
7. Create a column chart about the grouped and counted values of females so that the horizontal axis contains the names of the individual groups. The chart title should be "Females" and the chart should contain a legend. Mark columns belonging to the "***low***" value using green, the columns belonging to the "***average***" value using orange and the columns belonging to the "***high***" value using red colour. Place the chart next to the table part that contains the test data; the height of the chart should not be greater than the height of the table part.
8. In the table the width of columns *B:I* is the same and the data fit into them as shown in the example. Set the width of column *A* so that it does not hide any part of the texts in it.
9. Align all data in the table centered. Show the calculated data using italics. Format each cell in ranges *A1:I1, A27:G27* and *A32:G32* in the same way according to the example. The headings of the individual tables should be highlighted by bold style. Border the tables according to the example.

15 marks

Example for the Test evaluation exercise:



Number	Gender	Age	Group 1 aspect A	Group 1 aspect B	Group 2 aspect A	Group 2 aspect B	Group 3 aspect A	Group 3 aspect B
1	male	55	average	high	average	average	average	average
3	male	52	low	average	high	high	low	low
6	male	43	high	high	average	high	low	average
7	male	26	high	high	high	high	average	average
8	male	45	low	high	average	average	average	average
10	male	43	high	high	average	high	average	average
14	male	60	high	low	average	high	low	low
19	male	35	high	average	average	high	low	average
20	male	49	high	high	average	high	average	average
2	female	45	high	average	average	average	low	average
4	female	50	low	high	high	average	low	low
5	female	35	high	high	average	average	average	low
9	female	34	average	low	high	average	low	average
11	female	37	high	high	low	high	average	high
12	female	38	high	high	average	low	average	low
13	female	48	average	high	high	high	high	average
15	female	53	low	high	average	low	average	high
16	female	29	low	average	low	average	high	average
17	female	34	high	high	average	high	low	average
18	female	35	average	average	average	high	low	average

Number of females	*11*
Number of males	*9*
Average age	*42*

female	Group 1 aspect A	Group 1 aspect B	Group 2 aspect A	Group 2 aspect B	Group 3 aspect A	Group 3 aspect B
low	*3*	*1*	*2*	*2*	*5*	*3*
average	*3*	*3*	*6*	*5*	*4*	*6*
high	*5*	*7*	*3*	*4*	*2*	*2*

male	Group 1 aspect A	Group 1 aspect B	Group 2 aspect A	Group 2 aspect B	Group 3 aspect A	Group 3 aspect B
low	*2*	*1*	*0*	*0*	*4*	*2*
average	*1*	*2*	*7*	*2*	*5*	*7*
high	*6*	*6*	*2*	*7*	*0*	*0*

3. Conference

In the spring of 2008 a large conference was held at Springfield. The topic of each lecture was connected to the use of open source software at schools. The speakers were selected from the participants. Each lecture was 30 minutes long, the lunch break started at 12 o'clock. After the selection of the speakers and the formation of the final program the registrants had to register for each lecture they wished to attend. It is possible that several people with the same name participated in the conference.

1. Create a new database named `conference`. Import the three text files (`registrants.txt`, `registrations.txt`, `lectures.txt`), which are all UTF-8 encoded and tagged by tabs, into the database with the same name as the file name (***registrants***, ***registrations***, ***lectures***). The first line of each file contains the field names. Upon creation set the suitable types and keys.

Tables:

lectures (*id*, *speakerid*, *title*, *category*, *room*, *time*)

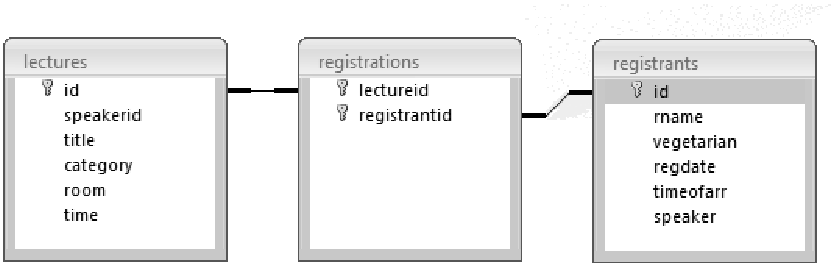
id The identifier of the lecture (number), key
speakerid The speaker's identifier (number) – the speaker's identifier is the same as the identifier received upon registration
title The title of the lecture (text)
category The category of the lecture (text)
room The name of the room where the lecture is held (text)
time The time when the lecture starts (time)

registrations (*lectureid*, *registrantid*)

lectureid The identifier of the lecture (number), key
registrantid The registrant's identifier (number), key

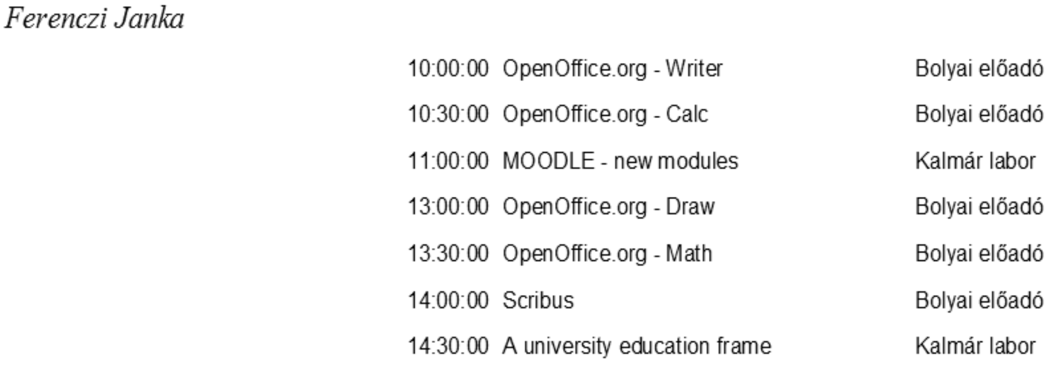
registrants (*id*, *rname*, *vegetarian*, *regdate*, *timeofarr*)

id The registrant's identifier (number), key
rname The registrant's name (text)
vegetarian Its value is true if the registrant is a vegetarian (Boolean)
regdate The date of the registration (date)
timeofarr The time the registrant arrived at the conference (time) – it is blank if he/she did not arrive



In the solution of the following exercises save the queries with the names given in brackets. Pay attention to displaying only the required values and no other data.

2. Create a query that lists the names of those who were not present when lunch started based on the time of arrival. (***2lunch***)
3. Create a query that lists the names and identifiers of those who participated in any of the lectures about Drupal in alphabetical order. Display each person only once. (***3drupal***)
4. Create a query that determines the most popular lecture based on the number of preliminary registrations. Give the title of the lecture (***4top***)
5. Ferenczi Janka and Berger Georgina know each other from the university. Then their fields of interest were similar. To determine whether it has changed create a query that displays the time, title and place of the lectures both of them registered to. (***5common***)
6. Create a Boolean type *speaker* field in table ***registrants***. Create a query that sets the value of the *speaker* field to true if the registrant's identifier is also among the speakers' identifiers. (***6speaker***)
7. Based on the preliminary registration each participant of the conference receives a personal program. Create a report that displays the start, the title and the place of the lectures selected by the participant in the order of the starting time grouped by participants. If required, you can prepare the solution of the exercise using a query. Upon creating the report use the following example as layout. (***7program***)



Ferenczi Janka

10:00:00	OpenOffice.org - Writer	Bolyai előadó
10:30:00	OpenOffice.org - Calc	Bolyai előadó
11:00:00	MOODLE - new modules	Kalmár labor
13:00:00	OpenOffice.org - Draw	Bolyai előadó
13:30:00	OpenOffice.org - Math	Bolyai előadó
14:00:00	Scribus	Bolyai előadó
14:30:00	A university education frame	Kalmár labor

8. Create a query that lists the names of the participants who arrived but missed at least one of the lectures selected by them. Display the name of each person only once. You may assume that those who arrived on time attended the lectures they selected. (***8latecomers***)

30 marks

4. Traffic check

Certainly everybody has seen a police patrol who watched the traffic from a roadside checkpoint during his/her service. The patrol's task is to screen for those who break the rules of road traffic as well as to check the passing vehicles at random or systematically. In certain cases a technical checking station capable of testing the technical condition of the selected vehicles is also installed.

You have to process the data of a day when both police and technical checks were performed on a one-way road. The machine attached to the traffic registration camera of the police car stored the data of at least 50 but at most 1000 vehicles passing on the street in file *`vehicles.txt`*. The lines of the file have the same structure, they contain the time and the registration number in the order of passing. The registration number is always 7 characters long, it contains the letters of the English alphabet, a hyphen and numerals in this order. Registration numbers different from the ones given in the example are also possible.

For example:

```
11 12 05 TI-2342
11 12 09 BU-5523
11 12 41 AAAA-99
11 13 12 DM-5632
…
```

The second line shows that the vehicle whose registration number is BU-5523 passed the checkpoint at 11 hours 12 minutes 9 seconds.

Create a program that answers the following questions. Save the source code of the program as *`patrol`*. (When writing the program you do not have to check the correctness or the validity of the data given by the user.)

Before displaying the results of exercise parts requiring writing on the screen display the number of the exercise part (for example `Exercise 3:`). If you request data from the user, display the kind of the value to be entered on the screen. Displays with or without accents are both accepted.

1. Read the data from file *`vehicles.txt`* and solve the following exercises using these.
2. Determine the minimum duration of the daily shift of the patrol if its work starts on the hour sharp and ends on the hour sharp (Each hour starts at 0 minutes 0 seconds and ends at 59 minutes 59 seconds). Display the result on the screen.
3. One vehicle is selected for technical check every hour. It is the vehicle that passes first in the given hour. Display the hour of the check and the registration number of the checked vehicle on the screen in the following format: `9 hour: AB-1234`. Every hour's data should be displayed in a separate line. Display only the data of the hours when there was a vehicle to be checked.
4. The first character of the registration number has a special meaning. From the possible letters "***B***" stands for bus, "***K***" for truck and "***M***" for motorbike, the other letters belong to cars. Display the number of vehicles that belong in the different categories and pass the checkpoint on the screen.
5. Find the start and the end of the longest period without traffic. Display the answer on the screen in the following format: `9:9:13 - 9:15:3`.

6. The patrol searches for a vehicle that was seen near an accident based on the registration number. The witnesses recalled only certain characters from the registration number so when the vehicle is searched for in the registry, the unknown characters of the registration number are replaced by character *. Request such a number from the user and then display the registration numbers that match it on the screen.

7. A road check lasts for exactly 5 minutes. While the check is in progress, vehicles may pass freely, the next vehicle is stopped only when the previous check is finished. If the patrol checked the first vehicle, then which vehicles were checked until the end of the shift? Write the time of the passing and the registration number of the checked vehicles into file `checked.txt` in the order of passing, in the same format as the input. Pay attention to the fact that the numbers belonging to temporal data contain leading zeros as well.

45 marks

Azonosító jel:

	maximum mark	achieved mark	examiner's signature
Word processing, presentation, graphics, web page creation **1. Hungarian poems**	30		
Spreadsheet processing **2. Test evaluation**	15		
Database management **3. Conference**	30		
Algorithmisation, data modelling **4. Traffic check**	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: ……………………….