

**BUS455**  
**“Applied Programming and Data Analysis for Business”**

**FINAL GROUP-BASED PROJECT**

**SUBMISSION DEADLINE:**

**DATE: 01 December 2017**

**TIME: 14.00**

**I. REPORT PREPARATION:**

You should submit TWO separate files:

**File 1: One Report file** (only PDF format) that includes the following:

**1.1 Problems 1-2 (Python - part). For each problem:**

- a. Python code  
*Copy your code and past it into you report.*
- b. Explanation of your solution
  - *Describe the idea of your designed solution*
  - *How your code is solving the problem*
- c. Screenshot of the results from the PyCharm
- d. Required:
  - *Comment in your code explaining what you are doing. For example:*  

```
s = [5, 35, 26, 44, 4, 1] # list s is created
```
  - *Code should be implemented in Python 3 (i.e., not in Python 2)*

**1.2 Problem 3-4 (R - part). For each problem:**

- a. R code  
*Copy your code and past it into you report.*  
Explanation of your solution
  - *Describe the idea of your designed solution*
  - *How your code is solving the problem*
- b. Screenshots of the results from the RStudio
- c. Required:
  - *comment in your code explaining what you are doing. For example:*  

```
y <- c(1:10) # create a data vector with elements 1-10
```

**1.3 Problem 5: performed analysis**

#### 1.4. Problem 6 (SQL – part)

- Follow Problem 6-details to report your results.
- REMEMBER: you must report the overall ER-model (relational schema) of your database.

#### REQUIREMENTS:

- Report should be typed (not handwritten)
- Report should be written in English

#### **File 2: One compressed file (ZIP or RAR format) including the following:**

##### 2.1 Problems 1-2 (Python):

- a. 2 code files (.py format) - for problems 1-2, respectively
  - Names of the .py files should include problem's number
  - For example: problem\_1.py; problem\_2.py

##### 2.2 Problems 3-4 (R):

- a. 2 code files (.r format) - for problems 3-4, respectively
  - Names of the .r files should include problem's number
  - For example: problem\_3.r; problem\_4.r

##### 2.3 Problem 6 (SQL):

- a. Database file (.db format)
- b. The .txt file with the SQL queries' code
- c. The overall ER-model (relational schema) of your database (PDF format)

## II. SUBMISSION PROCEDURE:

You must submit your final group-based project via WISEflow.

Please, remember, that you must submit two files prepared according to “I. REPORT PREPARATION” procedure specified above:

**File 1: One Report file** (only PDF format)

**File 2: One compressed file** (ZIP or RAR format)

**YOUR SUBMISSION WILL BE CHECKED FOR PLAGIARISM!**

## PROBLEM 1

I. You learned the following Python concepts:

- how to create functions using **def** keyword
- **while** control flow statement
- **if-else** conditional statement

Based on this knowledge write the following Python code:

Create function **reverse(s)** that will return a reversed version of any sentence **s** (i.e., sentence with reversed order of words) entered by user in the interactive mode as a string. Your function should take an argument **s**, which is any sentence entered via interactive mode as a string (i.e., *str* type) and print its reversed version.


The idea is the following:

```
Type your sentence:
I      like to      travel
travel      to like  I
```

### REQUIREMENTS:

1. You must apply **while** control flow statement and **if-else** conditional statement in **reverse(s)** function in order to create your mechanism to reverse the sentence.
2. You must create your own mechanism to reverse the sentence. It implies that built-in functions (such as *split()*, *reverse()* or any other) and imported modules are NOT allowed in your code.
3. The number of spaces between words should also be reflected in the reversed version of the sentence. Your solution should process all spaces between words in a sentence **s**. For example, sentence **s** can have two spaces between the first and the second words, and ten spaces between the second and the third words that should be reflected in the reversed sentence:

```
Type your sentence:
Hello Python      World
World            Python Hello
```



original sentence  
reversed sentence

II. Apply your **reverse(s)** function to the following string:

*Bergen is so beautiful !*

The number of spaces in the given sentence is as follows:

The diagram illustrates the number of spaces between the words in the sentence "Bergen is so beautiful !". Blue curly braces are placed below the text to indicate the following counts:

- 10 spaces between "Bergen" and "is"
- 2 spaces between "is" and "so"
- 1 space between "so" and "beautiful"
- 7 spaces between "beautiful" and "!"
- 3 spaces after the exclamation mark

NOTES:

- Code must be implemented in Python 3 (i.e., not in Python 2)
- Provide the screenshot of your results from the PyCharm

## PROBLEM 2

Write a Python code for the following problem.

Consider a small social network  $G$  that consists of seven persons connected to each other in the following way (Figure 1):

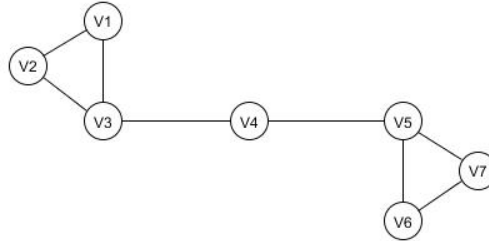


Figure 1. Social network  $G$

NOTATIONS:

1. Based on Figure 1, each person  $v$  has a number of its direct links (in the network  $G$ ) denoted by  $\deg_G(v)$ .

For example:

$\deg_G(v_1) = 2$ , because person  $v_1$  has two direct links (specifically, to persons  $v_2$  and  $v_3$ )

$\deg_G(v_3) = 3$ , because person  $v_3$  has three direct links (specifically, to persons  $v_1$ ,  $v_2$  and  $v_4$ )

2. List of persons (i.e., neighbors) that are reachable from a person  $v$  in at most one hop (i.e., via direct link) within network  $G$  is denoted by  $N_G(v)$ . Each person  $u$  from  $N_G(v)$  has a number of its direct links (in the network  $G$ ) denoted by  $\deg_G(u)$ .

Note: direct link is a link that directly connects two persons. For example:  $v_4$  and  $v_5$  are connected via direct link;  $v_4$  and  $v_3$  are connected via direct link.

For each person, we can measure its importance value  $I(v)$  based on the following algorithm (pseudo-code):

*Algorithm "Importance":*

```

1  FOR each person  $v$  from  $G$  do:
2     $I(v) = \frac{1}{1 + \deg_G(v)}$ 
3    FOR each person  $u$  from  $N_G(v)$  do:
4       $I(v) += \frac{1}{1 + \deg_G(u)}$ 
5    END
6  END
7  print:  $I(v)$  of each person
8  print:  $\sum I(v)$ , which is a sum of all  $I(v)$ -s in  $G$ 
  
```

Write a Python code for the *Algorithm “Importance”* and apply it to the social network presented in Figure 1.

NOTES:

- You must use **for** and/or **while** control flow statements in your code.
- Remember, according to lines 7-8 of *Algorithm “Importance”* your code should print the following:
  - importance value  $I(v)$  of each person
  - sum of the importance values of all persons in the network:  $\sum I(v)$

Your results should be printed in a similar format:

```
I ( person 1 ) = . . .
I ( person 2 ) = . . .
. . .
I ( person 7 ) = . . .
SUM: . . .
```

(Your computational results should be displayed in the field . . .)

- Code must be implemented in Python 3 (i.e., not in Python 2)
- Provide the screenshot of your results from the PyCharm applied to the network presented in Figure 1.

### PROBLEM 3

**I.** Write an R code that takes a temperature in any of the following degrees: Celsius, Fahrenheit, and Kelvin; and displays an equivalent temperature in any of the following degrees: Celsius, Fahrenheit, and Kelvin.

Your solution code should contain at least four functions. One of the functions should have the following format:

`Conversion (temp, from, to)`, where:

1. Argument `temp` takes a temperature (any numeric value) to process.
2. Argument `from` specifies the temperature scale of the temperature value `temp`. Specifically, argument `from` takes one of the following character-mode values:

"C" – corresponds to Celsius;  
 "F" – corresponds to Fahrenheit;  
 "K" – corresponds to Kelvin.

3. Argument `to` specifies the temperature scale of the resulting temperature value that should be returned by the `Conversion`-function. Argument `to` takes one of the following character-mode values:

"C" – corresponds to Celsius;  
 "F" – corresponds to Fahrenheit;  
 "K" – corresponds to Kelvin.

Thus, arguments `from` and `to` can take any of the following values: "C", "F", "K".

Function `Conversion` takes a value of `temp` in a temperature scale of `from` and returns the corresponding value in a temperature scale of `to`.

For example, function `Conversion(-41.5, "K", "F")` takes -41.5 degrees on the Kelvin scale and returns -534.37 degrees on the Fahrenheit scale.

Your code should display "ERROR"-message if parameters `from` and/or `to` take any character-mode value other than "C", "F", or "K".

For example, when the function `Conversion(-41.5, "D", "F")` is called it should return the following value in the R console:

```
[1] "ERROR"
```

Note:

Solving this problem, you are not allowed to have any packages loaded into your script by `library()`-function

HINT:

Convert Celsius to Fahrenheit:  $^{\circ}\text{F} = ^{\circ}\text{C} \times \frac{9}{5} + 32$

Convert Celsius to Kelvin:  $\text{K} = ^{\circ}\text{C} + 273.15$

Convert Fahrenheit to Kelvin:  $\text{K} = (^{\circ}\text{F} + 459.67) \times \frac{5}{9}$

**II.** Show your results for the following run:

```
Conversion(13.15, "K", "C")
```

```
Conversion(-30, "C", "F")
```

```
Conversion(-0.5, "H", "K")
```

Provide the screenshot of your results (from RStudio console)



### PROBLEM 4

Let's imagine a long hotel corridor with 100 doors all in a row. All doors are numbered 1 to 100 (in order). All doors are initially opened. We send 10 persons, one after another, to change the state of the doors:

to close the door (if it is opened) OR to open the door (if it is closed)

The first person sent to the corridor has to change the state of every 5-th door:

5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100

The second person sent to the corridor has to change the state of every 10-th door:

10, 20, 30, 40, 50, 60, 70, 80, 90, 100

The third person sent to the corridor has to change the state of every 15-th door:

15, 30, 45, 60, 75, 90

The fourth person sent to the corridor has to change the state of every 20-th door:

20, 40, 60, 80, 100

The fifth person sent to the corridor has to change the state of every 25-th door:

25, 50, 75, 100

The sixth person sent to the corridor has to change the state of every 30-th door:

30, 60, 90

The seventh person sent to the corridor has to change the state of every 35-th door:

35, 70

The eighth person sent to the corridor has to change the state of every 40-th door:

40, 80

The ninth person sent to the corridor has to change the state of every 45-th door:

45, 90

The tenth person sent to the corridor has to change the state of every 50-th door:

50, 100

You should write an R-code that will output the numbers of doors that are closed after all 10 persons went through the given hotel corridor.

NOTE: your R-code should not output the total number of closed doors. It should output the specific numbers of doors that are closed after all 10 persons went through the given hotel corridor.

Provide the screenshot of the resulted numbers of doors (from RStudio console)

Requirement: you have to use **for**-statement(s) in your code.

**PROBLEM 5**

Perform comparative analysis of R and Python in terms of data analysis:

Discuss advantages of Python over R for data analysis. Discuss advantages of R over Python for data analysis.

Note:

You are allowed to provide example(s), and/or relative case(s), and/or (hypothetical) real life situation(s).

*Format: technical essay*

*Maximum (for problem 5): 1500 words. Text above this limit will be ignored.*

## PROBLEM 6

Design and create your own database (based on the SQLite3).

Database should be logically consistent and should reflect the realistic entities, attributes and relations. It can reflect any industry such as telecoms, retail, health care or any other industry (based on your preferences and interests).

### ***PLAGIARISM CONTROL:***

- 1. You are not allowed to use any RDBMS database templates from the Internet.*
- 2. Make sure you create your own original database.*

### **1. DATABASE IDEA:**

Describe the idea: what does your database reflect in terms of the selected industry.

### **2. ER-MODEL:**

**Create the overall ER-model (relational schema) of your database.**

#### Entities requirements:

Your database should contain **10-15 entities** (i.e., tables) with corresponding attributes.

#### Relationships requirements:

Your database should have:

- at least one one-to-one (1:1) relationship
- at least three one-to-many (1:N) / many-to-one (N:1) relationships that are not a part of N:N relationships
- at least two many-to-many (N:N) relationships

*Please, see slide 31 (SQL - Lecture 1 "Databases" available via ITSLEARNING) to recall what is the overall database ER-model (relational schema). Your ER-model should contain all details regarding the entities, attributes, datatypes and relationships.*

#### **NOTE:**

You can submit the scanned hand-drawn version of the overall database ER-model (relational schema). In this case, make sure that your hand-drawn ER-model is readable (i.e., high-resolution).

### 3. DATABASE CREATION:

- The database schema and data records should be created using **only SQL queries**.
- Each table of the resulted database should be populated by at least five records using **INSERT INTO** statement.
- You must use **only SQLite DB Browser** (employed in the course) to create your database.

*You have to use “Execute SQL”-tab from the SQLite DB Browser for all manipulations to create the database and to populate it by data records. No other tools are allowed.*

#### 3.1 SQL code:

- Provide all SQL queries in the PDF-report.

*It should be possible to copy the overall SQL code from your PDF-report and execute it with no errors having the entire database created. Thus, all SQL queries should be provided in the correct order.*

- Attach the **.txt** file of the overall SQL code to the submission.  
*If you have difficulties to create and save .txt file you can use online tools such as:*  
<https://www.editpad.org/>
- Attach your **.db** database file to the submission.

#### 3.2. For each relationship (between tables)

a. Briefly describe the purpose of the relationship (its role in the database):

- Why the given relationship is created; which entities does it connect
- Specify the type of the relationship: 1:1, 1:N (or N:1), or N:N

b. For each table participating in the given relationship:

- Briefly describe the purpose of the table; what does it reflect
- For column(s), that is/are assigned to be primary and/or foreign key(s), explain briefly why the given specific column(s) is/are assigned to be primary key(s) and/or foreign key(s).