BABEȘ-BOLYAI UNIVERSITY
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Bachelor Degree Written Exam, July 1, 2019
Computer Science – English

VARIANT 2

REMARKS

- All subjects are compulsory and full solutions are requested.
- The minimum passing grade is 5,00.
- The working time is 3 hours.

SUBJECT Operating systems

1 Answer the following questions, considering that all the instructions of the C functions below are executed successfully

```
1   void f1(){
2     int i;
3     for(i=0; i<3; i++) {
4       if(fork() == 0) {}
5       wait(0);
6     }
7   }
8   void f2(){
9     int i, p = 0;
10    for(i=0; i<3; i++) {
11      if(p == 0) {
12        p = fork();
13      }
14      wait(0);
15    }
16  }
```

a) Draw the hierarchy diagram of the processes created by executing function `f1` in the parent process.

b) Draw the hierarchy diagram of the processes created by executing function `f2` in the parent process.

c) Rewrite function `f1` so that it creates the same number of processes as function `f2`, but having a different hierarchy. Explain and draw the process hierarchy diagram.

d) Explain the role of the system call `wait`.

2 Answer the following questions about the UNIX Shell script below, considering that it is executed in a directory with the structure shown on the right, with the following command line arguments: `a d1 apples b d2 pears`.

```
1   #!/bin/bash
2   while [ -n "$1" ]; do
3     if [ -d $1 ]; then
4       for F in `find $1 -type f`; do
5         if file $F | grep -q -v text; then
6           echo "Error: $F"
7           continue
8         fi
9         sed "s/^$2//" $F > $F.text
10      done
11      shift
12      shift
13    else
14      shift
15    fi
16  done
```

```
.
├─ d1        - directory
│ ├─ a       - text file
│ ├─ b       - text file
│ └─ x.txt   - PDF file renamed .txt
└─ d2        - directory
  └─ b       - text file
```

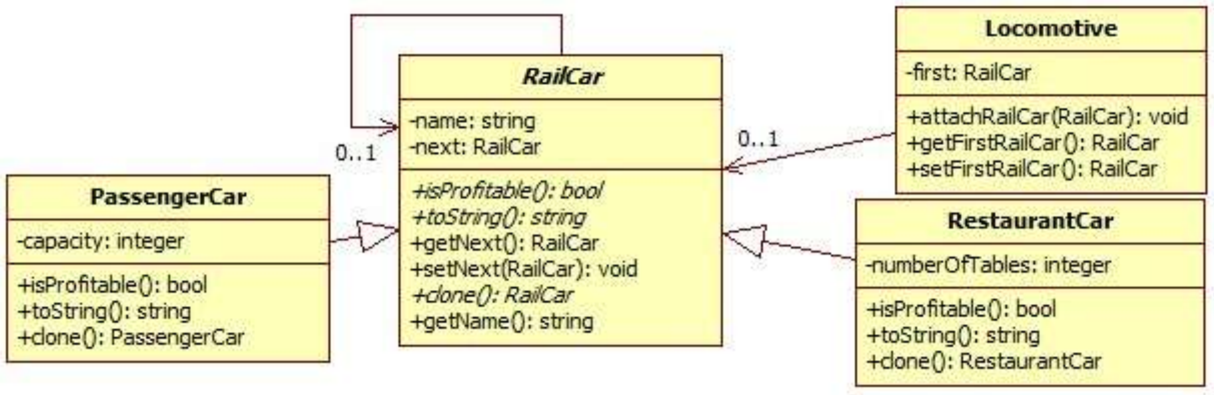
a) What will the script execution print and what are its effects on the structure of files and directories above?
b) Explain line 5 in detail.
c) How are the script execution results affected when the quotes on line 9 are replaced with apostrophes?
d) Explain what happens if lines 13 and 14 are removed.

VARIANT 2

SUBJECT Algorithms and Programming

Write a program in one of the programming languages Python, C++, Java, C#, with the following requirements:

a) Define classes **RailCar**, **PassengerCar**, **RestaurantCar**, **Locomotive** (stores rail cars attached to it), according to the following UML diagram (constructors are not shown on the diagram). Only the methods indicated at b) must be implemented, the others should only be declared.



- The attribute *name* in class **RailCar** (the railcar's name) must contain at least 2 characters; the attributes *capacity* in class **PassengerCar** and *numberOfTables* in class **RestaurantCar** must be strictly positive. The constructors will enforce the constraints.
- The abstract class **RailCar** has three abstract methods: **isProfitable**, **toString** and **clone**. The methods **clone** in **PassengerCar** and **RestaurantCar** create a copy of the current object**.**
- The method **attachRailCar** in class **Locomotive** attaches the railcar given as parameter in the sequence of railcars already attached to the locomotive such that the railcars are arranged in alphabetical order, by the string returned by the method **toString.**
- A passenger car is profitable (**isProfitable**) if its *capacity* is more than 40 seats. A restaurant car is profitable if the number of tables in the car (*numberOfTables*) is at least 20.
- For passenger cars the method **toString** returns the *name* concatenated to the *capacity*. For restaurant cars the method **toString** returns the *name* concatenated to the number of tables (*numberOfTables*).

b) Implement the following methods from the diagram at a): constructors for classes **RailCar, PassengerCar**, **RestaurantCar**; methods **toString** from classes **PassengerCar** and **RestaurantCar**; method **attachRailCar** from class **Locomotive**; methods **isProfitable** from classes **PassengerCar** and **RestaurantCar**; method **clone** from **PassengerCar** and method **setNext** from **RailCar.**

c) Define a function that has as parameter an object *l* of type **Locomotive** and eliminates from *l* those railcars that are not profitable.

d) Define a function that has as parameter an object *l* of type **Locomotive** and returns another object of type **Locomotive** that contains copies of *l*'s railcars, in reverse order.

e) The main function of the program creates an object *l* of type **Locomotive**, to which the following railcars are added (choose values for the unspecified attributes): three objects of type **PassengerCar** having 50, 32 and 40 seats and two objects of type **RestaurantCar** having 25 and 20 tables. Eliminate unprofitable railcars from *l*, using the function at c) and then call the function at d) to obtain a new object *l'* of type **Locomotive**. In the end, print all railcars in *l'*.

f) Write the specifications for method **attachRailCar** in class **Locomotive**.

- **Please indicate the used programming language.**
- **Do not define other methods than those shown in the diagram (except for the constructors).**
- **Do not use sorted containers and predefined sorting operations.**

You may use existing libraries for ***data structures*** *(Python, C++, Java, C#).*

VARIANT 2

SUBJECT Databases

Consider a database that stores data about the medical history of pacients in a hospital. No two doctors in the hospital have the same first name and last name. The database has the following structure:

- table ***Pacients*** with fields **PId, LastName, FirstName, Age, DateOfBirth**;
- table ***Procedures*** with fields **ProcId**, **Name, Description, Price**;
- table ***PacientsMedicalRecords*** with fields **PacientId**, **ProcedureId**, **Date, DoctorLastName, DoctorFirstName, DoctorSpecialty, DoctorSeniority**.

1. Determine the primary keys and the foreign keys for each of the above tables.
2. Determine at least 3 functional dependencies that refer to fields that don't represent ids.
3. Consider the following structure changes and specify which of them are essential for the database to be in **3NF** (the 3rd normal form). In the case of an affirmative answer, justify your choice.
   a. Creating a separate table to store doctors.
   b. Adding the integrity constraint **Age = current date – DateOfBirth** in the ***Pacients*** table.
   c. Eliminating the **Age** field from the ***Pacients*** table.
   d. Creating a separate table to store the pacients' age.
   e. Adding the integrity constraint **Date** < **current date** in the ***PacientsMedicalRecords*** table**.**
4. On the given structure, write an SQL query that is equivalent to the following query:

$$\Pi_{\text{LastName, FirstName}} (\sigma_{\text{DoctorSpecialty = 'radiology'}} (\textbf{\textit{PacientsMedicalRecords}}) \otimes_{\text{PacientId = PId}} \textbf{\textit{Pacients}}) \cap$$
$$\Pi_{\text{LastName, FirstName}} (\sigma_{\text{DoctorSpecialty = 'cardiology'}} (\textbf{\textit{PacientsMedicalRecords}}) \otimes_{\text{PacientId = PId}} \textbf{\textit{Pacients}})$$

5. On the given structure, write an SQL query that returns, for the specialties with the largest amount of money collected from procedures, the specialty name, the total number of procedures performed within the specialty and the amount of money collected (***Specialty, NumberOfProcedures, AmountOfMoney)***. Display all specialties that satisfy this condition.

BAREM INFORMATICĂ
VARIANTA 2

Subiect Algoritmică şi Programare

Oficiu – 1p
Definirea clasei abstracte **RailCar** – 1 p din care
  atribut – 0.2
  constructor si metode **-** 0.8
Definirea clasei **PassengerCar**– 1.1 p din care
  relația de moștenire – 0.2
  atribut – 0.1
  constructor si metode – 0.8
Definirea clasei **RestaurantCar** – 1.1 p din care
  relația de moștenire – 0.2
  atribut – 0.1
  constructor si metode – 0.8
Definirea clasei **Locomotive** – 1.3p din care
  atribute– 0.1
  constructor - 0.1
  metoda attachRailCar - 1.1p
Funcția de la punctul c) – 1.8p din care
  signatura corectă - 0.1p
  eliminare vagoane neprofitabile- 1.7p
Funcția de la punctul d) – 1.7p din care
  signatura corectă - 0.1p
  creare lista vagoane - 1.6p
Funcția principală e) – 0.5p
f) Specificaţia metodei **attachRailCar** din clasa **Locomotive**– 0.5p

Notă. Datorită faptului că signatura metodei setFirstRailCar() este incompletă pe diagrama UML, se va acorda punctaj maxim (1.7p) pentru punctul d).

Subiect Baze de date
1. 0.5p (chei primare) + 0.5p (chei externe) = 1p
2. 1p
3. a, c
  2 x (0.5p răspuns + 0.5p justificare) = 2p
4. rezolvarea completă a interogării = 2p
5. rezolvarea completă a interogării = 3p

1p of

Subiect Sisteme de operare

Oficiu – 1p

1.a Diagramă cu proces părinte, fii, nepoți și strănepoți, totalizând 8 procese – 1p
1.b Diagramă liniară cu 4 procese: părinte, fiu, nepot, strănepot – 1.5p
1.c Implementare corectă - 0.5p
  Explicație – 0.5p
  Diagramă – 0.5p
1.d Așteaptă terminarea unui proces fiu – 1p

2.a Afișează eroare pentru x.txt și creează trei fișiere cu extensia .text din care se șterg cuvintele date – 1.5p
2.b Comenzile file și grep – 0.5p
  Argumentele –q și –v – 0.5p
  Expresia regulară – 0.5p
2.c Caută literalmente $2, fără a substitui valoarea argumentului – 0.5p
2.d Ciclu infinit – 0.5p