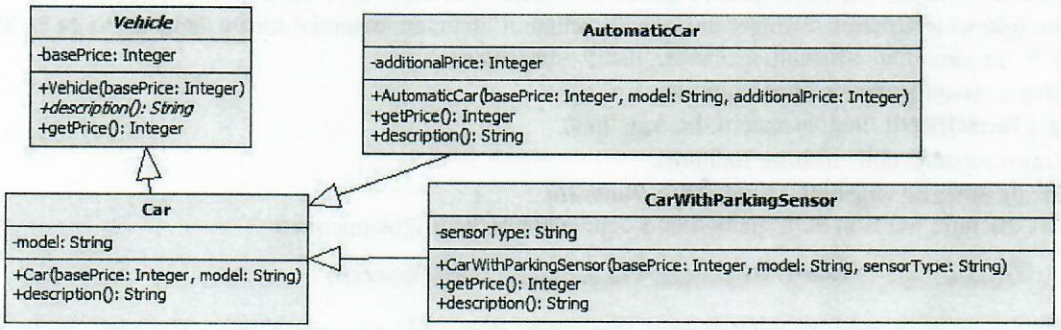
BABEȘ-BOLYAI UNIVERSITY
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Bachelor Degree Written Exam, July 2, 2018
Computer Science – English
VARIANT 1

SUBJECT 1. Algorithms and Programming

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

a) **Defines** the classes ***Vehicle*** (abstract), **Car**, **AutomaticCar** and **CarWithParkingSensor** according to the following UML class diagram:



- *basePrice* and *additionalPrice* must be strictly positive; *sensorType* and *model* must be non-null and nonempty. The constructors will enforce the constraints.
- The method **getPrice**() from the class ***Vehicle*** returns the vehicle's base price; the method **getPrice**() from the class **AutomaticCar** returns the additional price added to the car's base price and the method getPrice() from the class **CarWithParkingSensor** returns the car's base price, to which 2500 is added.
- The method **description**() from the class **Car** returns the car's model; the method **description**() from the class **AutomaticCar** returns the text "Automatic car " concatenated with the car's model and the method **description**() from the class **CarWithParkingSensor** returns the text "Car with parking sensor " concatenated with the sensor type, " " and the car's model.

b) **Defines a function** that has as parameter a list L of objects of type ***Vehicle***, and returns a list with pairs of type <*model*, *numberOfCars*>, which contains for each car *model* in the list L the number of cars *numberOfCars* having that model. In the list returned as result, each car model from the input list L will occur only once.

c) **Defines a function** that has as parameter a list of objects of type ***Vehicle*** and rearranges the list such that all cars having the price in the interval [1000, 2000] will appear in the list before the cars having the price less than 1000 or greater than 4000. This must be achieved by keeping the order of the items in the original list. Using lists or other auxiliary data structures is forbidden.

d) **Defines a function** that receives as parameter a list of objects of type ***Vehicle*** and prints the descriptions of all cars in the list.

e) The **main function** of the program creates a list with the following cars (please choose any values for their unspecified properties): one Audi, one automatic Audi, one Toyota, one automatic Mercedes and one Opel with parking sensor. For the previously defined list, call the function at b) and print the returned list. Call the function at c) and then print the list resulted after calling the function at d).

f) For the **List** data type used in the program, write the specifications of the used operations.

Remarks

- **Please indicate the used programming language.**
- **Do not use sorted containers and predefined sorting operations.**
- **Do not define other methods than those required in the subject.**

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

SUBJECT 2. Databases

Consider a database that stores the outcomes of all the matches played by national football teams (a team can play one match on a given date). The database has the following structure:
- table ***Teams*** with fields: **TId, Country, Continent;**
- table ***Players*** with fields: **PId, TeamId, Name, Age, NumberOfMatches;**
- table ***Matches*** with fields: **Team1Id, Team1Goals, Team2Id, Team2Goals, Date, Stadium, NumberOfSeats, City, Country**.

1. Determine the primary keys and the foreign keys for each of the above tables.
2. Determine at least 4 functional dependencies that refer to fields that don't represent codes.
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 countries.
   b. Using a **DateOfBirth** field instead of the **Age** field.
   c. Creating a separate table to store stadiums.
   d. Adding the integrity constraint ***Team1Id > Team2Id.***
4. On the initial structure, write an SQL query that is equivalent to the following query:

$$\Pi_{Name}\ (\ \sigma_{Continent\ =\ 'Asia'}\ (\ \textit{Teams}\ \otimes_{TId\ =\ TeamId}\ \sigma_{NumberOfMatches\ >100}\ (\textit{Players})))$$

5. Write an SQL query that returns the total number of matches played by each national team (***Country***, ***NoMatches***).

SUBJECT 3. Operating Systems

3.1 The code fragment below is compiled successfully into executable `pr`. Argument `w` of function `f` specifies which FIFO should be used for writing: 0 for `a` and 1 for `b`. Considering that all the instructions are executed successfully, that all necessary FIFOs are deleted and created again before each execution, and that `pr` is the only program accessing those FIFOs, answer the following questions.

```
void f(char* a, char* b, int w, char* s) {
  int f[2], r=1-w; char c;
  if(fork() == 0) {
    f[0] = open(a, w==0 ? O_WRONLY : O_RDONLY);
    f[1] = open(b, w==1 ? O_WRONLY : O_RDONLY);
    write(f[w], s, 1);
    read(f[r], &c, 1);
    printf("%c\n", c);
    close(f[0]); close(f[1]);
    exit(0);
  }
}
int main(int n, char** a) {
  int i;
  for(i=1; i<n; i+=4) {
    f(a[i], a[i+1], a[i+2][0]-'0', a[i+3]);
  }
  for(i=1; i<n; i+=4) {wait(0);}
  return 0;
}
```

a) Draw the process hierarchy diagram for an execution where `pr` receives `4*K` command line arguments.

b) What will the following execution print? Justify your answer.

```
./pr p q 1 x
```

c) What will the following execution print? Justify your answer.

```
./pr p q 1 x p q 0 y
```

d) Draw a diagram depicting the child processes and their read/write operations on the FIFOs, for the execution in (c).

e) What will the following execution print? Justify your answer.

```
./pr p q 1 x q p 1 y
```

3.2 Command `sed "s/A/B/"` replaces the first occurrence of regular expression `A` on each line with string `B`, substituting in `B` any reference `\N` with the content of the Nth expression between parentheses in `A`. What are the displayed results of the UNIX Shell script below, when executed in a directory containing C/C++ sources and headers? Explain line 3 in detail: commands, arguments, and pipe.

```
1 for F in *.c *.cpp *.h; do
2   if [ -f $F ]; then
3     grep "#include.*<" $F | sed "s/^.*<\(.*\)>.*$/\1/"
4   fi
5 done | sort
```

REMARKS

- All subjects are compulsory and full solutions are requested.

BAREM INFORMATICĂ
VARIANTA 1

Subiect 1 (Algoritmică și Programare):

Oficiu – 1p
Definirea clasei abstracte Vehicle – 0.3p din care
- atribut – 0.1
- metode - 0.2

Definirea clasei Car – 0.3p din care
- relația de moștenire – 0.1
- atribut – 0.1
- metode - 0.1

Definirea clasei AutomaticCar– 1p din care
- relația de moștenire – 0.15
- atribut – 0.15
- constructor – 0.35
- metode – 0.35

Definirea clasei CarWithParkingSensor – 1p din care
- relația de moștenire – 0.15
- atribut – 0.15
- constructor – 0.35
- metode – 0.35

Funcția de la punctul b) – 2.25p din care
- signatura corectă - 0.1p
- construire listă perechi conținând modele distincte- 2p
- returnare listă rezultat – 0.15

Funcția de la punctul c) – 2.25p din care
- signatura corectă - 0.1p
- rearanjare listă – 2.15p

Funcția de la punctul d) – 0.4p din care
- signatura corectă - 0.1p
- afișare descrierei mașini din listă – 0.3p

Funcția principală e) – 0.5p
f) Specificațiile operațiilor folosite pentru tipul de dată Listă– 1p

Subiect 2 (Baze de date)

1. 0.5p (chei primare) + 0.5p (chei externe) = 1p
2. 0.25p x 4 = 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 3 (Sisteme de operare):

3.1.a - diagrama cu un părinte și mulți fii - 0.5p
- K procese fiu - 0.5p

3.1.b - nimic - 0.5p
- se blochează open la deschiderea FIFO-ului - 0.5p

3.1.c - x si y - 0.5p
- în ordine nedeterminabilă - 0.5p

3.1.d - diagrama circulară cu două procese și două FIFO-uri - 1p

3.1.e - nimic - 0.5p
- se blochează open-urile și creează deadlock - 0.5p

3.2 Rezultate - lista sortată a fișierelor header sistem incluse în surse - 1p

3.2 Linia 3 - grep: extrage liniile care includ fișiere header sistem - 0.5p
- grep: detaliere expresie regulară - 0.5p
- pipe: transmite output-ul lui grep ca input lui sed - 0.5p
- sed: pastrează pe linie doar numele fișierului header - 1p
- sed: detaliere comandă și expresie regulară – 0.5p