BABEŞ-BOLYAI UNIVERSITÄT CLUJ-NAPOCA
FAKULTÄT FÜR MATEMATHIK UND INFORMATIK

Schriftliche Abschlussprüfung, 1. JULI 2019
Fachgebiet Informatik in deutscher Sprache

VARIANTE 2

BEMERKUNG.

- Alle Prüfungsthemen sind verpflichtend. Bei allen Prüfungsthemen müssen vollständige Lösungen angegeben werden.
- Die Mindestnote für das Bestehen der Abschlussprüfung ist 5,00.
- Die Arbeitszeit beträgt 3 Stunden.

TEIL Betriebssysteme

1 Beantworten Sie folgende Fragen, berücksichtigend, dass alle Befehle aus den untenstehenden C-Funktionen erfolgreich durchgeführt werden.

```
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) Zeichnen Sie die Hierarchie der Prozesse, die durch die Ausführung der Funktion `f1` im Vater Prozess erstellt wurden.

b) Zeichnen Sie die Hierarchie der Prozesse, die durch die Ausführen der Funktion `f2` im Vater Prozess erstellt wurden.

c) Schreiben Sie die Funktion `f1` so um, dass sie genau so viele Prozesse erstellt wie die Funktion `f2`, aber eine andere Hierarchie hat. Erklären und zeichnen Sie die Prozesshierarchie.

d) Erklären Sie die Rolle des Systemaufrufs `wait`.

2 Beantworten Sie die folgenden Fragen zum Shell UNIX-Skript, berücksichtigend, dass es in einem Verzeichnis mit der folgenden Struktur, mit den Argumenten in der Kommandozeile ausgeführt wird: `a d1 Apfel b d2 Birnen`

```
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 "Fehler: $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        - Verzeichnis
│  ├─ a      - Textdatei
│  ├─ b      - Textdatei
│  └─ x.txt  - PDF-Datei in .txt umbenannt
└─ d2        - Verzeichnis
   └─ b      - Textdatei
```

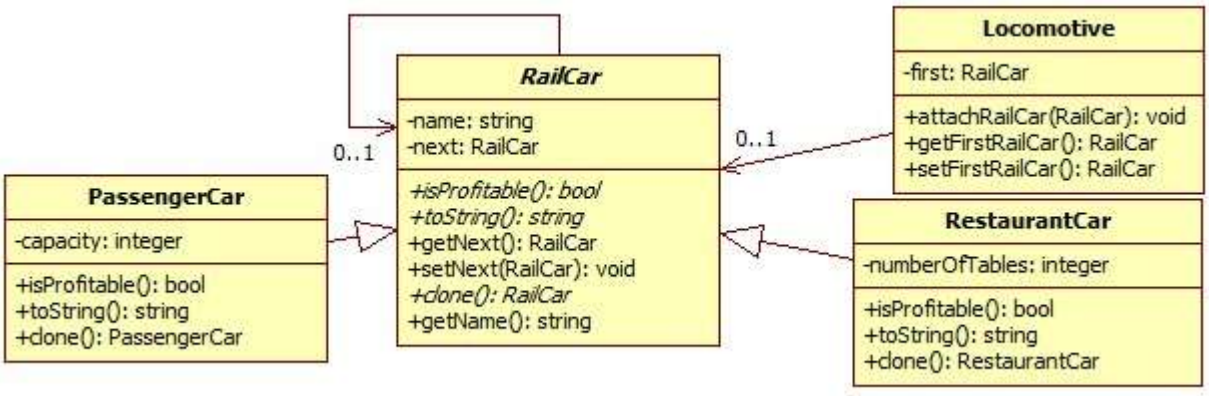
a) Was wird bei der Skriptausführung dargestellt und welche Auswirkungen hat dies auf die oben angegebene Verzeichnis- und Dateistruktur?
b) Erklären Sie im Detail Zeile 5
c) Wie ändern sich die Ergebnisse des Skriptausführungs, wenn die Anführungszeichen in Zeile 9 durch Apostrophe ersetzen wird?
d) Erklären Sie, was passiert, wenn die Zeilen 13 und 14 löschen werden.

VARIANTE 2

TEIL Algorithmen und Programmierung

Schreiben Sie ein Programm in Python, C++, Java oder C#, mit den folgenden Bedingungen:

a) Definiert die Klassen **RailCar** (Wagen), **PassengerCar** (Personenwagen), **RestaurantCar** (Restaurantwagen) **Locomotive** (Lokomotive, speichert die beiliegende Wagen) aufgrund des folgenden UML-Diagramms (die Konstruktoren sind nicht im Diagramm enthalten). Nur die Methoden von b) sollen implementiert werden, der Rest sollen nur deklariert werden.



- das Attribut *name* der Klasse **RailCar** (der Name des Wagens) soll mindestens zwei Zeichnen haben und die Attribute *capacity* (die Kapazität) der Klasse **PassengerCar** und *numberOfTables* (die Anzahl der Tische) der Klasse **RestaurantCar** sollen echt positiv sein. Die Konstruktoren sollen diese Bedingungen durchsetzen.
- Die abstrakte Klasse **RailCar** stellt drei abstrakte Methoden bereit: **isProfitable**, **toString** und **clone**. Die Methode **clone** der Klassen **PassengerCar** und **RestaurantCar** erzeugt eine Kopie der aktuellen Objektes.
- Die Methode **attachRailCar** der Klasse **Locomotive** fügt in der Liste der beiliegenden Wagen den Wagen, so dass die Wagen in einer alphabetischen Reihenfolge sind, nach dem Rückgabewert der Methode **toString.**
- Ein Personenwagen ist profitable (**isProfitable**) wenn seine Kapazität größer als 40 ist. Ein Restaurantwagen ist profitable, wenn die Anzahl der Tische (*numberOfTables*) größer als 20 ist.
- Für Personenwagen die Methode **toString** gibt *name* erweitert mit der Kapazität (*capacity*) zurück. Für Restaurantwagen die Methode **toString** gibt *name* erweitert mit *numberOfTables* zurück.

b) Implementiert die folgenden Methoden aus dem Diagramm von a): die Konstruktoren der Klassen **RailCar, PassengerCar**, **RestaurantCar**; die Methoden **toString** der Klassen **PassengerCar** und **RestaurantCar**; die Methode **attachRailCar** der Klasse **Locomotive**; die Methoden **isProfitable** der Klasse **PassengerCar** und **RestaurantCar**; die Methode **clone** der Klasse **PassengerCar** und die Methode **setNext** der Klasse **RailCar.**

c) Definiert eine Funktion, welche ein Objekt *l* vom Typ-**Locomotive** bekommt. Die Funktion entfernt von der Liste der beiliegenden Wagen diejenige, die nicht profitable sind.

d) Definiert eine Funktion, welche ein Objekt *l* vom Typ-**Locomotive** bekommt. Die Funktion gibt eine Liste zurück, welche die Kopien der Wagen von *l* in einer umgekehrten Reihenfolge enthält.

e) Erzeugt in der Main-Funktion ein Objekt vom Typ-**Locomotive**, in dem die folgenden Wagen hinzugefügt werden (wählen Sie die Werte für die unspezifizierte Eigenschaften der Objekte aus): drei Objekte vom Typ-**PassangerCar** mit 50, 32 und 40 Plätze und zwei Objekte vom Typ-**RestaurantCar** mit 25 und 20 Tische. Mit der Funktion von c) entfernen Sie die nicht profitablen Wagen und rufen Sie die Funktion von c) auf, um ein neues Objekt *l'* vom Typ-**Locomotive** zu erzeugen. Am Ende, geben Sie die Wagen von *l'* auf dem Bildschirm aus.

f) Schreiben Sie die Spezifikation der Methode **attachRailCar** der Klasse **Locomotive**.

Bemerkung

- **Geben Sie die Programmiersprache an.**
- **Sie dürfen keine weiteren Methoden definieren, außer Konstruktoren und den Methoden, die im UML-Diagramm dargestellt sind.**
- **Sie dürfen nicht Sorted-Containers oder vordefinierte Sortierfunktionen benutzen.**

Sie dürfen Datenstrukturen von Standard-Bibliotheken benutzen (Python, C++, Java, C, C#).

VARIANTE 2

TEIL Datenbanken

Gegeben sei eine Datenbank, die Informationen über Patienten aus einem Krankenhaus speichert. Man weiß, dass es in dem Krankenhaus keine zwei Ärzte mit demselben Namen und Vornamen gibt. Die Datenbank hat die folgende Struktur:

- Tabelle ***Pacients*** mit den Feldern: **PId, LastName, FirstName, Age, DateOfBirth**;
- Tabelle ***Procedures*** mit den Feldern: **ProcId**, **Name, Description, Price**;
- Tabelle ***PacientsMedicalRecords*** mit den Feldern: **PacientId**, **ProcedureId**, **Date, DoctorLastName, DoctorFirstName, DoctorSpecialty, DoctorSeniority**.

1. Bestimmen Sie die Kandidat- und die Fremdschlüssel für jede der obigen Tabellen.
2. Bestimmen Sie mindestens drei funktionale Abhängigkeiten auf Felder, die nicht Ids repräsentieren.
3. Gegeben seien die folgenden Strukturveränderungen. Bestimmen Sie welche von diesen notwendig sind, um die Datenbank in 3NF (die dritte Normalform) zu sein. Für bejahende Antworten begründen Sie die Auswahl.
    a. Die Erstellung einer weiteren Tabelle, um Ärzte zu speichern.
    b. Das Einfügen der Einschränkung **Age = current date – DateOfBirth** in der Tabelle ***Patients*****.**
    c. Die Beseitigung des Feldes **Age** aus der Tabelle ***Patients*****.**
    d. Die Erstellung einer weiteren Tabelle, um das Alter des Patienten zu speichern.
    e. Das Einfügen der Einschränkung **Date** < **current date** in der Tabelle ***PacientsMedicalRecords*****.**
4. Für die gegebene Struktur, schreiben Sie eine SQL Abfrage, die zu der folgenden Abfrage äquivalent ist:

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

5. Für die gegebene Struktur, schreiben Sie eine SQL Abfrage, die jene Abteilungen (identifiziert durch **DoctorSpecialty**) angibt, in denen das meiste Geld aus den Behandlungen (**Procedure**) einkassiert wurde:
    - die Namen der Abteilungen,
    - die gesamte Anzahl der Behandlungen, die in jeder dieser Abteilungen durchgeführt wurden,
    - und die gesamte Geldsumme, die in jeder dieser Abteilungen einkassiert wurde (**Specialty, NumberOfProcedures, AmountOfMoney**).

Man gebe alle Abteilungen an, welche diese Bedingung erfüllen.

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