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

Schriftliche Abschlussprüfung, 5. September 2023
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 Algorithmen und Programmierung

Wichtiger Hinweis:

- Das Fehlen eines richtigen Programmierstils (sinvolle Namen für Variablen, Einrückung des Codes, ggf. Kommentare, Lesbarkeit des Codes) führt zum Verlust von 10 % der entsprechenden Punktzahl.
- Für Pseudocode sind nur die folgenden Anweisungen erlaubt: *Zuweisung*, *For-Schleife*, *Wenn*, *Do-while-Schleife*, *While-Schleife*, *Ausgeben* und *Zurückgeben*.
- Man darf keine weiteren Attribute, Methoden definieren, außer Konstruktoren, ggf. Destruktoren und was in der Aufgabenbeschreibung angegeben ist. Die Sichtbarkeit der in der Aufgabenbeschreibung angegebenen Attributen darf nicht verändert werden.

Aufgabe 1. (1.5 Punkte)

Gegeben seien eine natürliche Zahl x und der Vektor $a[1]$, $a[2]$....$a[n]$ ($n \geq 1$) bestehend aus absteigend sortierten eindeutigen natürlichen Zahlen. Schreiben Sie in Pseudocode einen Algorithmus mit der Zeitkomplexität $O(\log_2 n)$, der die Position *poz* zurückgibt, an der x im Vektor a liegt (falls x in dem Vektor überhaupt vorkommt), oder die Position an der x eingefügt werden soll, damit die absteigende Reihenfolge der Elemente weiterhin beibehalten wird. Begründen Sie die Komplexität. **Anmerkung.** Das Einfügen eines Elements an Position k in dem Vektor erfordert das Verschieben der Elemente an Positionen k, $k+1$, ..., n um eine Position nach rechts und das Erhöhen der Länge des Vektors.

Beispiel. Wenn $n=4$, der Vektor a ist (14, 12, 9, 5) dann:

- für $x=15$ wird $poz=1$ zurückgegeben, weil der Vektor durch das Einfügen von **15** (**15**, 14, 12, 9, 5) ist;
- für $x=11$ wird $poz=3$ zurückgegeben, weil der Vektor durch das Einfügen von **11** (14, 12, **11**, 9, 5) ist;
- für $x=12$ wird $poz=2$ zurückgegeben, weil das Element **12** schon an der zweiten Position liegt.

Aufgabe 2. (1.5 Punkte)

Schreiben Sie in Pseudocode einen Algorithmus, der als Eingabeparameter bekommt, eine Ganzzahl k, $0 < k \leq n$ und einen Text, der als ein Vektor $x[1]$, $x[2]$....$x[n]$ ($n \geq 1$) von Zeichen dargestellt wird. Der Algorithmus erstellt den Vektor y[1], y[2]....y[n], der die zirkuläre Permutation des Vektors x um k Positionen nach rechts darstellt. Geben Sie die Zeitkomplexität des Algorithmus an.

Beispiel. Wenn $n=6$ şi und der Text ist S, T, R, I, N, G, dann für $k=2$ wird der Vektor y N, G, S, T, R, I beinhalten.

Aufgabe 3. (1.5 Punkte)

Gegeben sei ein Vektor a bestehend aus n Elementen ($a[1]$, ..., $a[n]$), die eine Permutation von $\{1, ..., n\}$ ($n>0$) darstellen. Schreiben Sie in Pseudocode einen Algorithmus mit der im schlechtesten Fall Zeitkomplexität $\theta(n)$, der bestimmt, ob der Vektor a ein Inorder-Durchlauf eines binären Suchbaums mit n von 1 bis n indexiert Knoten repräsentieren kann. Wenn ja, geben Sie den Wert 0 aus, andernfalls wird eine Liste von Austauschen ausgegeben, die einen Vektor ergeben kann, der die Bedingung erfüllt. Jede Lösung, die nicht in die angegebene Komplexitätsklasse fällt, erhält eine Teil-Punktzahl.

Beispiel. Wenn $n = 3$, der Vektor a ist (3, 2, 1), dann wird der Austausch 1 3 ausgegeben (die Elemente an den Positionen 1 und 3 müssen vertauscht werden – also ein möglicher Baum, dessen Durchlauf (1, 2, 3) ist, hat Wurzel 2, linkes Kind 1 und rechtes Kind 3).

Aufgabe 4. (2.25 Punkte)

Gegeben sei der folgende unvollständige C++-Code. Ergänzen Sie die Implementierung der Klassen **Computer** und **SorterByTotalPrice**, damit die **Main**-Methode auf dem Bildschirm "HC85, 165" ausgibt. Die **Sort**-Methode der **SorterByTotalPrice**-Klasse bekommt eine Produktliste und wird sie nach Gesamtpreis absteigend sortieren. Das Sortieren erfolgt durch den Aufruf der in der C++-Standardbibliothek definierten Methode **std::sort**. Der Gesamtpreis eines Computers berechnet sich durch das Addieren einer Steuer zum Grundpreis.

```
#include <string>
#include <vector>
#include <iostream>
#include <algorithm>
class Product {
   private:
       std::string name;
       int baseprice;
   public:
       Product(const std::string& name, int baseprice):
                    name{name},
                    baseprice{baseprice} {}
       virtual int total_price() {
              return this->baseprice;
       }
       virtual std::string toString() {
              return this->name;
       }
      virtual ~Product() {}
};
class Computer: public Product {
   private:
       int tax;
       //A: missing code
};
class SorterByTotalPrice {
     //B: missing code
};
int main() {
       std::vector<Product*> computers{new Computer{"HC90", 140, 10},
                                       new Computer{"HC91", 100, 12},
                                       new Computer{"HC85", 150, 15}};
       SorterByTotalPrice::sort(computers);
       std::cout << computers[0]->toString();
       for (auto c: computers)
            delete c;
       return 0;
}
```

Aufgabe 5. (2.25 Punkte)

Implementieren Sie in C++ die Klassen für Aufgaben 1-3 und danach die Funktion für Aufgabe 4.

1. Die ***Contact***-Klasse ist abstrakt und besitzt ein privates Attribut *name* vom Typ-String, eine abstrakte Methode **sendMessage** (pur virtual), die einen Parameter *message* als String bekommt und eine Methode **getName**, die den Wert des Name-Attributs zurückgibt.
2. Die **Person**-Klasse erweitert die **Contact**-Klasse, besitzt ein zusätzliches privates Attribut *number* vom Typ-String und die Methode **sendMessage** bekommt eine Nachricht vom Typ-String als Parameter und gibt sowohl den Namen und die Nummer der Person als auch die Nachricht aus.
3. Die **Group**-Klasse erweitert die **Contact**-Klasse und kann null oder mehrere Kontakte beinhalten. Die Methode **addContact** bekommt einen Kontakt als Parameter und fügt ihn der Kontaktliste hinzu. Die Methode **sendMessage** gibt sowohl den Namen und die Nummern aller Kontakte der Kontaktliste als auch die als Parameter übergebene Nachricht aus.
4. Erzeugen Sie die folgenden Objekte (der Wert für das Attribut *number* kann beliebig sein): 2 Personen mit den Namen "Mother" und "Father"; 2 Personen mit den Namen "Jane", "John"; eine Gruppe "Parents", die "Mother" und "Father" beinhaltet; eine Gruppe "Family", die die Gruppe "Parents" und "Jane" beinhaltet. Mit der Verwendung der Funktion **sendMessage** senden Sie der Gruppe "Family" und der Person "John" die folgende Nachricht zu: "You are invited to my birthday party next week!". Der Speicher muss korrekt verwaltet werden.

TEIL Datenbanken

Aufgabe 1. (4 Punkte)

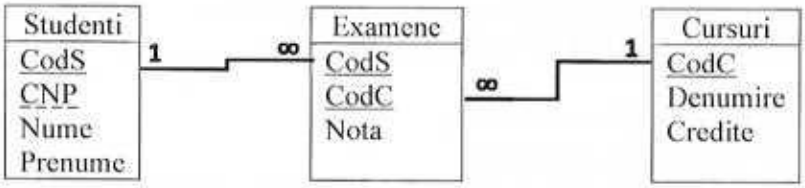
Ein Fahrradverleih speichert die folgenden Informationen in einer relationalen Datenbank:

- Ein Fahrradhersteller hat einen Code, einen Namen und ein Herkunftsland.
- Ein Fahrrad gehört zu einem Fahrradhersteller, hat einen Code, ein Modell, eine Größe (eine Zahl, z. B. 56) und einen Mietpreis pro Tag.
- Ein Kunde hat einen CNP, einen Namen, eine Telefonnummer und eine E-Mail-Adresse.
- Wenn ein Kunde ein Fahrrad mietet, wird ein Verleihformular erstellt. Ein Kunde kann mehr als ein Fahrrad auf einmal mieten. Ein Verleihformular gehört zu einem Kunden, es enthält ein Datum, an dem das Formular erstellt wurde, und eine Liste der ausgeliehenen Fahrräder, jeweils mit einem Start- und Enddatum für den Verleihzeitraum. Wenn ein Kunde am selben Tag zurückkommt, um weitere Fahrräder auszuleihen, werden diese zu dem bestehenden Verleihformular für diesen Tag hinzugefügt. Ein Kunde kann an einem Tag nicht mehr als ein Verleihformular haben. Wenn ein Kunde an einem anderen Tag wiederkommt, um Fahrräder auszuleihen, wird ein neues Verleihformular erstellt.

Erstellen Sie ein relationales Datenbankschema in BCNF, wobei Sie die Primärschlüssel, Kandidatenschlüssel und Fremdschlüssel sorgfältig hervorheben. Erstellen Sie das Schema auf eine der im folgenden Beispiel gezeigten Arten:

* Beispiel für die Tabellen Studenti, Cursuri und Examene:

V1. Diagramm mit Tabellen, Primärschlüssel durch eine durchgezogene Linie unterstrichen, Kandidatenschlüssel durch eine gestrichelte Linie unterstrichen, Beziehungen direkt zwischen den Fremdschlüsseln und den entsprechenden Primär-/Kandidatenschlüsseln (z. B. Beziehung zwischen der CodS-Spalte in Examene und der CodS-Spalte in Studenti).



V2.

Studenti[CodS, CNP, Nume, Prenume]
Cursuri[CodC, Denumire, Credite]
Examene[CodS, CodC, Nota]
Primärschlüssel sind durch eine durchgezogene Linie und Kandidatenschlüssel durch eine gestrichelte Linie unterstrichen.{CodS} ist Fremdschlüssel in Examene, der auf {CodS} in Studenti verweist. {CodC} ist Fremdschlüssel in Examene, der auf {CodC} in Cursuri verweist.

Aufgabe 2. (5 Punkte)

Gegeben seien folgende Relationenschemata:

Clienti[CodClient, NumeClient, Telefon]
Soferi[CodSofer, NumeSofer, Salariu]
Autoturisme[CodAuto, NumarInmatriculare, MarcaAuto, *CodSofer*]
Curse[CodCursa, *CodClient, CodAuto*, DataCursa, MomentPlecare, DurataCursa, LocPlecare, Destinatie, SumaDePlata]
Primärschlüssel sind unterstrichen. Fremdschlüssel sind kursiv geschrieben und haben denselben Namen wie die Spalten, auf die sie sich beziehen.

a. Schreiben Sie eine SQL-Abfrage, die den Code, den Namen und das Gehalt der Fahrer zurückgibt, die das höchste Gehalt unter den Fahrern haben, die mindestens eine Fahrt mit dem Zielort (Destinatie) 'Iulius Mall Cluj' mit einem Auto mit der Marke (MarcaAuto) 'Ford'gemacht haben. Entfernen Sie Duplikate.

b. Gegeben seien folgende Instanzen der Relationen Autoturisme und Curse:

Autoturisme:

CodAuto	NumarInmatriculare	MarcaAuto	CodSofer
1	CJ11AAA	Ford	1
2	CJ22BBB	Opel	2
3	CJ33CCC	Volkswagen	3

Curse:

Cod Cursa	Cod Client	Cod Auto	Data Cursa	MomentPlecare	DurataCursa	Loc Plecare	Destinatie	Suma DePlata
1	1	1	2023.06.25	07:30	15	LP1	D1	20
2	1	2	2023.06.26	08:30	20	LP2	D3	30
3	2	1	2023.06.26	12:30	10	LP4	D4	15
4	1	1	2023.06.30	10:15	5	LP1	D2	10
5	3	3	2023.07.01	15:05	40	LP1	D4	50
6	3	2	2023.07.01	20:00	12	LP2	D2	15

b1. Geben Sie das Ergebnis der Auswertung der folgenden Abfrage für die angegebenen Instanzen an. Geben Sie ausschließlich die Werte der Tupel und Spaltennamen im Ergebnis an, ohne alle Schritte der Abfrageauswertung zu nennen.

```
SELECT a.CodSofer, COUNT(DISTINCT c.CodClient) Nr
FROM  Autoturisme a INNER JOIN Curse c ON a.CodAuto = c.CodAuto
WHERE c.DataCursa BETWEEN '2023.01.01' AND '2023.06.30'
GROUP BY a.CodSofer
HAVING COUNT(DISTINCT c.CodClient) >= ALL
        (SELECT COUNT(DISTINCT c.CodClient)
         FROM Autoturisme a INNER JOIN Curse c ON a.CodAuto = c.CodAuto
         WHERE c.DataCursa BETWEEN '2023.01.01' AND '2023.06.30'
         GROUP BY a.CodSofer)
```

b2. Begründen Sie für jede der folgenden funktionalen Abhängigkeiten, ob diese für die Daten in der Instanz Curse gilt oder nicht:

- {CodCursa} → {CodAuto}
- {CodAuto} → {LocPlecare}.

THEMA Betriebssysteme

Aufgabe 1 (4p)

Beantworten Sie die folgenden Fragen zur Ausführung des unten stehenden Codefragments, wobei Sie wissen, dass die Funktion `mkfifo` eine FIFO-Datei erstellt und die Funktion `unlink` diese löscht.

```
 1  int main(int argc, char** argv) {
 2      int f; char c = 'a';
 3
 4      mkfifo("fifo", 0600);
 5      if(fork() == 0) {
 6          f = open("fifo", O_RDONLY);
 7          read(f, &c, 1);
 8          printf("%c\n", c);
 9          close(f);
10          exit(0);
11      }
12
13      f = open("fifo", O_WRONLY);
14      write(f, "xy", 2);
15      close(f);
16
17      wait(NULL);
18      unlink("fifo");
19      return 0;
20  }
```

a) Was wird auf der Konsole angezeigt?

b) Was wird in der Konsole angezeigt, wenn die Zeile 18 zwischen die Zeilen 9-10 verschoben wird?

c) Wie wird die Funktion des Programms beeinflusst, wenn die Zeile 4 zwischen die Zeilen 5-6 verschoben wird?

d) In welcher Situation wird 'a' angezeigt?

e) Erklären Sie die Rolle der Argumente für die Funktion `mkfifo` in Zeile 4.

Aufgabe 2 (5p)

Beantworten Sie die folgenden Fragen zur Ausführung des unten stehenden UNIX-Shell-Skripts.

```
 1  #!/bin/bash
 2
 3  for A in $*; do
 4      for F in x/*; do
 5          if ! grep -E -q "^$A(_+$A)*$" $F; then
 6              echo "$A" >> $F
 7          elif [ `grep -E "^$A(_+$A)*$" $F | wc -l` -gt 1 ]; then
 8              grep -E -v "^$A(_{1,}$A){0,}$" $F > $F.aux
 9              echo "$A" >> $F.aux
10              mv $F.aux $F
11          fi
12      done
13  done
```

a)	Welchen Inhalt wird die Datei `x/a.txt` in der rechten Spalte haben, nachdem das Skript mit dem Argument `abc` ausgeführt wurde?	`abc` `def`

b)	Welchen Inhalt wird die Datei `x/b.txt` in der rechten Spalte haben, nachdem das Skript mit dem Argument `abc` ausgeführt wurde?	`abc`
		`def`
		`abc`

c)	Welchen Inhalt wird die Datei `x/c.txt` in der rechten Spalte haben, nachdem das Skript mit den `abc def`-Argumenten ausgeführt wurde?	`def_def`
		`abc`
		`abc_def`
		`abc_abc`
		`def_def_def`

d)	Erklären Sie ausführlich die Bedingung in Zeile 7.	
e)	Erklären Sie ausführlich den regulären Ausdruck in Zeile 8.	

BAREM INFORMATICĂ
VARIANTA 2

Subiect Algoritmică şi Programare

Oficiu – 1p

Problema 1. Punctaj - 1.5p

- Subalgoritm corect: **1p**
  - signatura corectă: 0.1p
  - implementare căutare binară: 0.9p
- Justificarea complexității (căutare binară): **0.5p**

Problema 2. Punctaj - 1.5p

- Subalgoritm corect implementat: **1.25p**
  - signatura corectă: 0.1p
  - implementare: 1.15p
- Complexitate $\theta(n)$: **0.25p**

Problema 3. Punctaj - 1.5p

Soluție: *șirul trebuie transformat în permutarea identică*.

- Soluție având complexitatea timp în caz defavorabil $\theta(n)$ - 1.5p
- Soluție având complexitatea timp în caz defavorabil $\theta(n \cdot \log_2 n)$ - 1p
- Soluție având complexitatea timp în caz defavorabil $\theta(n^2)$ - 0.75p
- Alte soluții care nu se încadrează în clasele de complexități de mai sus - 0.25p

Problema 4. Punctaj - 2.25p

- Implementarea clasei **SorterByTotalPrice** cu metoda statică **sort()** și folosirea funcției **std::sort**, cu o funcție lambda (**0.75p)**
  - pentru folosire **std::sort**, cu operator <
    bool operator<(Product*, Product*) 0.6p
  - pentru folosire **std::sort**, cu o functie standalone
    bool nume_functie(Product*, Product*) 0.75p
  - pentru funcții de sortare implementate (fără **std::sort**) corect 0.25p
- Implementarea metodei total_price in clasa **Computer** (**0.5p**)
- Implementarea constructorului în clasa **Computer** (**0.5p**)
- Implementarea metodei **toString()** in clasa **Computer** (**0.5p**)

Problema 5. Punctaj - 2.25p

- clasa ***Contact*** **(0.4p)**:
  - câmp privat *name*, constructor (0.1), metoda **getName()** (0.05), metoda abstractă **sendMessage** (0.15), destructor virtual (0.1).
- clasa **Person (0.35p):**
  - moștenire (0.05), câmp privat **number**, constructor (0.15), metoda **sendMessage** (0.15)
- clasa **Group (0.85p)**:
  - moștenire (0.05), câmp privat ***lista de contacte*** (0.2), constructor (0.1), metoda **addContact** (0.2), metoda **sendMessage** (0.3), destructor (a se vedea punctajul de mai jos pentru distrugerea corecta a obiectelor)
- main **(0.65p)**:
  - construire corectă obiecte **Person** (0.15), construire corectă obiecte **Group** + adăugare de contacte (0.3), apel **sendMessage** la Group si la Person (0.1), distrugere corecta obiecte (0.1)

Subiect Baze de date

Oficiu – **1p**

Problema 1. Punctaj - 4p

- relații cu atribute corecte, chei primare, chei candidat: **3p**
- legături modelate corect (chei externe): **1p**

Problema 2. Punctaj - 5p

- **a -** rezolvarea completă a interogării: **2.5p**

- **b1** - rezultat evaluare interogare:

CodSofer	Nr
1	2

  - coloane – **0.5p**
  - valori tuplu – **1p**

- **b2** - {CodCursa}→{CodAuto} este satisfăcută – **0.25p**; **0.25p** explicație
  - {CodAuto}→{LocPlecare} nu este satisfăcută – **0.25p**; **0.25p** explicație

Notă: La specializările Informatică engleză și Informatică maghiară se iau în considerare versiunile traduse în limbile corespunzătoare.

Subiect Sisteme de operare

Oficiu – 1p

Problema 1. Punctaj – 4p

a) Se va tipări 'x' – 0.5p
b) Se va tipări 'x' – 1p
c) Dacă părintele execută open înainte ca fiul să creeze fifo, fiul se blochează la open şi părintele la wait – 1p
d) Niciodată – 1p
e) Primul este numele fifo-ului de creat şi al doilea sunt permisiunile de acces – 0.5p

Problema 2. Punctaj – 5p

a) Conţinutul va fi acelaşi – 1p
b) Conţinutul va fi `def` urmat de `abc` pe altă linie – 1p
c) Conţinutul va fi `abc_def` urmat de `abc` pe altă linie urmat de `def` pe altă linie – 1p
d) Fişierul `$F` conţine cel puţin 2 linii formate exclusiv din una sau mai multe valori `$A` separate prin cel puţin câte un _ - 1p
e) Întreaga linie e o secvenţă care începe cu valoarea `$A` urmată de 0 sau mai multe secvenţe formate din unul sau mai multe _ urmate de valoarea `$A` – 1p