UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ

Proba scrisă a examenului de licenţă, 3 iulie 2023
Informatică Română

VARIANTA 1

NOTĂ

- Toate subiectele sunt obligatorii. La toate subiectele se cer rezolvări cu soluții complete.
- Nota minimă ce asigură promovarea este 5,00.
- Timpul efectiv de lucru este de 3 ore.

SUBIECT Algoritmică şi programare

Notă

- Lipsa unui stil de programare adecvat (denumiri sugestive pentru variabile, indentarea codului, comentarii dacă sunt necesare, lizibilitatea codului) duce la pierderea a 10% din punctajul aferent subiectului.
- În Pseudocod se pot utiliza exclusiv următoarele instrucțiuni: *atribuire*, *pentru*, *dacă, câttimp*, *repetă*, *afișare* și *returnare*.
- Nu adăugați alte atribute, metode, în afara celor menționate în enunț, cu excepția constructorilor și a destructorilor, dacă e cazul. Nu modificați vizibilitatea atributelor specificate în enunț.

Problema 1. (1.5 puncte)

Fie subalgoritmii definiți în Pseudocod mai jos. Subalgoritmul **transformare** are ca date de intrare șirul x având n elemente numere întregi ($x[1]$, …, $x[n]$) și șirul y având m elemente numere întregi ($y[1]$, …, $y[m]$) și ca date de ieșire șirul z având k elemente numere întregi ($z[1]$, …, $z[k]$). Subalgoritmul **g** are ca date de intrare șirul x ($x[1]$, …, $x[n]$) având n elemente numere întregi și un număr natural i ($1 \leq i \leq n$), iar ca date de ieșire șirul y ($y[1]$,…, $y[n]$) având n elemente numere întregi. Ce va conține șirul z în urma apelului **transformare**($x, n, y, 0, z, k$)? Justificați răspunsul.

```
Subalgoritmul g(x, n, y, i)
  Dacă i ≤ n atunci
    y[i] ← x[i]
    g(x, n, y, i + 1)
  SfDacă
SfSubalgoritm

Subalgoritmul transformare(x, n, y, m, z, k)
  Dacă n = 0 atunci
      g(y, m, z, 1)
      k ← m
   altfel
      y[m +1] ← x[n]
      transformare(x, n - 1, y, m + 1, z, k)
  SfDacă
SfSubalgoritm
```

Problema 2. (3 puncte)

Se dă un șir $a[1]$, …, $a[n]$ ($4 \leq n \leq 10000$) având elemente numere întregi din intervalul $[-10^{15}, 10^{15}]$. Scrieți un program în limbajul de programare C++, având complexitatea timp în caz defavorabil $O(n^3)$, care afișează pe ecran patru numere naturale x, y, z, t ($x, y, z, t \in [1, n]$, $x < y < z < t$), astfel încât $a[x] + a[y] + a[z] + a[t] = 0$. În cazul în care nu există patru numere pe poziții distincte în șir cu suma 0, se va afișa pe ecran valoarea -1. **Notă**. *Orice soluție care nu se încadrează în clasa de complexități indicată va primi punctaj parțial. Se pot folosi algoritmi și containere din STL.*

Problema 3. (2.25 puncte)

Implementați în C++ clasele **MyObject, MyInteger, MyString, MyObjectList** și **MyListIterator** astfel încât rezultatul tipărit în urma execuției funcției de mai jos sa fie cel menționat în comentariu și memoria sa fie gestionată corect.

```
void function()
{
	MyObjectList list{};
	list.add(new MyInteger{ 2 }).add(new MyString{ "Hi" });
	MyString* s = new MyString{ "Bye" };
	list.add(s).add(new MyString{ "5" });

	MyListIterator i{ list };
	while (i.isValid()) {
		MyObject* o = i.element();
		o->print();
		i.next();
	} // prints: 2 Hi Bye 5
}
```

Problema 4. (2.25 puncte)

Implementați în limbajul C++ clasa **Persoana** care are atributele private: *nume* (șir de caractere), *prenume* (șir de caractere), *vârsta* (număr întreg), are constructor care inițializează toate atributele și metode de acces pentru toate atributele clasei. Scrieți o funcție care sortează o listă de persoane. Funcția va primi ca parametri lista de persoane și o funcție care reprezintă criteriul de sortare. Complexitatea timp a funcției de sortare, în caz defavorabil, trebuie să fie $\Theta(n \log_2 n)$, n fiind numărul de elemente din listă.

Apelați funcția pentru a sorta crescător o listă de obiecte de tip **Persoana**:

- după atributul *nume*
- după atributul *vârsta*
- după *vârsta* și *nume* (dacă au aceeași *vârsta* se compară după atributul *nume*)

Observație. Nu se vor folosi operații de sortare predefinite.

SUBIECT Baze de date

Problema 1. (4 puncte)

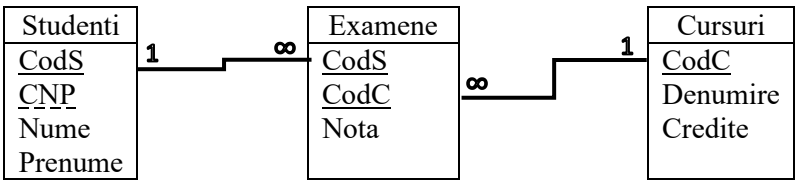
La un campionat de patinaj artistic participă concurenți care se pot înscrie la diferite probe de concurs. În cadrul probelor, concurenții sunt evaluați de arbitri. Datele se stochează într-o bază de date relațională.

- Un concurent are un cod, nume, prenume și data nașterii.
- O probă de concurs are un cod, denumire, descriere și dată calendaristică.
- Un concurent se poate înscrie la mai multe probe de concurs, iar la o astfel de probă se pot înscrie desigur mai mulți concurenți.
- Un arbitru are un cod, nume și prenume.
- O evaluare este făcută de un arbitru pentru un concurent care participă într-o anumită probă și presupune acordarea unei note întregi între 1 și 10 concurentului respectiv pentru proba respectivă. Un arbitru poate evalua mai mulți concurenți care participă în aceeași probă, respectiv mai multe probe pentru un anumit concurent. Un arbitru poate evalua cel mult o dată participarea unui anumit concurent într-o anumită probă. Un concurent poate fi evaluat într-o probă doar dacă s-a înscris la proba respectivă.

Realizați o schemă relațională BCNF pentru baza de date, evidențiind riguros cheile primare, cheile candidat și cheile externe. Realizați schema într-una din manierele indicate în exemplul de mai jos:

* exemplu pentru tabelele Studenti, Cursuri și Examene:

V1. Diagramă cu tabele, chei primare subliniate cu linie continuă, chei candidat subliniate cu linie întreruptă, legături trasate direct între cheile externe și cheile primare / candidat corespunzătoare (de exemplu, legătură trasată între coloana CodS din Examene și coloana CodS din Studenti).



V2.

Studenti[CodS, CNP, Nume, Prenume]
Cursuri[CodC, Denumire, Credite]
Examene[CodS, CodC, Nota]
Cheile primare sunt subliniate cu linie continuă, iar cheile candidat sunt subliniate cu linie întreruptă.
{CodS} este cheie externă în Examene și face referire la {CodS} din Studenti. {CodC} este cheie externă în Examene și face referire la {CodC} din Cursuri.

Problema 2. (5 puncte)

Se dau următoarele scheme relaționale:

Ziare[CodZiar, NumeZiar]
Jurnalisti[CodJurnalist, Nume, Prenume, *CodZiar*]
Categorii[CodCategorie, Denumire]
Articole[CodArticol, Titlu, DataPublicare, Text, *CodJurnalist*, *CodCategorie*]
Cheile primare sunt subliniate. Cheile externe sunt scrise cursiv și au aceeași denumire cu coloanele la care fac referire.

a. Scrieți o interogare SQL care returnează numărul total de articole din categoria cu denumirea *Economie* publicate de jurnaliști care lucrează la ziarul cu numele *Universul*.

b. Se dau instanțele următoare pentru relațiile Categorii și Articole:

Categorii

CodCategorie	Denumire
1	c1
2	c2
3	c3

Articole

CodArticol	Titlu	DataPublicare	Text	CodJurnalist	CodCategorie
1	a1	1-1-2022	t1	1	2
2	a2	1-2-2022	t2	2	3
3	a3	1-1-2022	t3	3	1
4	a4	1-5-2022	t4	1	2
5	a5	1-3-2022	t5	2	1
7	a7	1-7-2022	t7	2	2
8	a8	1-5-2022	t8	4	2

b1. Precizați rezultatul evaluării interogării de mai jos pe instanțele date. Menționați strict valorile tuplului / tuplurilor și denumirile coloanelor din rezultat fără a prezenta toți pașii evaluării interogării.

```
SELECT A.CodJurnalist, COUNT(*) Nr
FROM Articole A INNER JOIN Categorii C ON A.CodCategorie = C.CodCategorie
WHERE C.Denumire = 'c2'
GROUP BY A.CodJurnalist
HAVING 2 < (SELECT COUNT(*)
      FROM Articole A2
      WHERE A2.CodJurnalist = A.CodJurnalist)
```

b2. Explicați dacă următoarele dependențe funcționale sunt satisfăcute sau nu de datele din instanța Articole:

- $\{CodJurnalist\} \rightarrow \{Titlu\}$
- $\{CodArticol\} \rightarrow \{Text\}$

SUBIECT Sisteme de Operare

Problema 1 (5p)
Răspundeţi la următoarele întrebări despre execuţia fragmentului de cod de mai jos, presupunând că pipe-urile şi procesul se creează cu succes.

```
 1  int main(int argc, char** argv) {
 2      int a[2], b[2], c[2], i;
 3      char val=0, x=0;
 4
 5      pipe(a); pipe(b); pipe(c);
 6      if(fork() == 0) {
 7          close(b[0]); close(c[1]);
 8          for(i=0; i<3; i++) {
 9              write(a[1], &val, 1);
10              write(b[1], &x, 1);
11              read(c[0], &x, 1);
12              read(a[0], &val, 1);
13          }
14          printf("%d\n", val);
15          close(a[0]);close(a[1]);
16          close(b[1]);close(c[0]);
17          exit(0);
18      }
19      close(b[1]); close(c[0]);
20      for(i=0; i<3; i++) {
21          read(b[0], &x, 1);
22          read(a[0], &val, 1);
23          val++;
24          write(a[1], &val, 1);
25          write(c[1], &x, 1);
26      }
27      close(a[0]);close(a[1]);
28      close(b[0]);close(c[1]);
29      printf("%d\n", x);
30
31      wait(NULL);
32      return 0;
33  }
```

a) Care este rolul pipe-ului `c`?
b) Ce se va afişa în consolă?
c) Ce se va întampla dacă se interschimbă liniile 11 şi 12?
d) Cum e afectată funcţionarea proceselor dacă linia 7 se mută între liniile 14-15 şi linia 19 se mută între liniile 26-27?
e) Ce se va întampla dacă se elimină linia 17?

Problema 2 (4p)
Răspundeţi la următoarele întrebări despre execuţia scriptului Shell UNIX de mai jos.

```
1  #!/bin/bash
2
3  for F in `find /usr/sbin -type f`; do
4    if [ ! -x "$F" ] && ls -l "$F" | grep -q -E "^-(.{2}x|.{5}x|.{8}x)"
5    then
6      echo "$F:" `ps -e -f | grep -E "$F" | grep -E -v "grep" | wc -l`
7    fi
8  done
```

a) Ce valori va lua variabila `F`?
b) Explicaţi expresia regulară de pe linia 4.
c) Ce fişiere vor îndeplini prima parte a condiţiei de pe linia 4 (înainte de `&&`)?
d) Ce fişiere vor îndeplini a doua parte a condiţiei de pe linia 4 (după `&&`)?
e) Care este rolul celui de-al doilea `grep` de pe linia 6?

BAREM INFORMATICĂ
VARIANTA 1

Subiect Algoritmică şi Programare

Oficiu – 1p

Problema 1. Punctaj - 1.5p

- Răspuns corect (șirul *z* va conține elementele șirului *x* în ordine inversă): **0.75 p**
- Justificare: **0.75p**

Problema 2. Punctaj - 3p

- Soluție având complexitatea timp în caz defavorabil $O(n^3)$ - 3p
- Soluție având complexitatea timp în caz mediu $\theta(n^3)$ - 1.75p
- Soluție având complexitatea timp $O(n^3 \log_2 n)$ - 1.25p
- Soluție având complexitatea timp $O(n^4)$ - 0.5p

Problema 3. Punctaj - 2.25p

- Clasa **MyObject** (**0.2p**): abstractă, funcția abstractă **print,** virtual destructor
- Clasa **MyInteger** (**0.3p):** mostenire, camp privat de tip *int* + constructor + funcția **print**
- Clasa **MyString (0.3p)**: mostenire, camp privat de tip *string* + constructor, funcția **print**
- Clasa **MyObjectList (0.85p)**:
  - vector de elemente *Object* (pointers) (0.2), funcția **add** (0.2), funcția **length** (0.1), destructor (0.15), funcție care accesează elementul de pe o poziția dată *sau* funcție **getObjects** *sau* friend class MyListIterator (0.2).
- Clasa My**ListIterator (0.6p)**:
  - câmpuri private de tip *MyObjectList* și *int* (elementul curent) + constructor (0.25), funcția **isValid** (0.1), funcția **next** (0.1), funcția **element** (0.15).

Problema 4. Punctaj - 2.25p

- Clasa Persoana - **0.3p**
- Signatura funcției: lista/array + parametru de tip funcție (de comparare sau funcție care returnează cheia după care sortăm) - **0.3p**
- Implementare **Merge Sort** (partea recursivă/interclasare) - 0.6+0.75 = **1.35p**
- Implementare funcții de comparare/cheie (sau funcții lambda) și apel corect pentru cele 3 variante de sortare- **0.3p**

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:

CodJurnalist	Nr
2	1

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

- **b2** - {CodJurnalist}→{Titlu} nu este satisfăcută – **0.25p**; **0.25p** explicație
  - {CodArticol}→{Text} 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 – 5p

a) Previne procesul fiu de la a citi din `a` ceea ce a scris el însuşi – 1p
b) Se va afişa 0 şi 3 în orice ordine – 1p
c) Dacă procesul fiu citeşte din `a` ce a scris chiar el, se blochează, altfel functionează ca la a) – 1p
d) Funcţionarea nu este afectată – 1p
e) Procesul fiu îşi continua execuţia cu instricţiunile de după IF – 1p

Problema 2. Punctaj – 4p

a) Numele fişierelor normale din directorul /usr/bin găsite recursiv şi în subdirectoare – 0.5p
b) La început de linie, minus urmat de orice grup de caractere de lungime 2, 5 sau 8 urmat de x – 1p
c) Fişierele care nu pot fi executate de utilizatorul care rulează scriptul – 0.5p
d) Fişierele care au drept de execuţie pentru owner, group sau others – 1p
e) Elimină din lista de procese cele care conţin în linia de comandă `$F` pe cele care conţin cuvântul `grep` – 1p