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

Proba scrisă a examenului de licenţă, septembrie 2019
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 Sisteme de operare

1 Răspundeți la următoarele întrebări, considerând că toate fișierele header necesare sunt incluse în programul de mai jos și că toate instrucțiunile se execută cu succes.

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

a) Desenați ierarhia proceselor create, incluzând și procesul părinte.

b) Ce afișează execuția programului?

c) Ce afișează execuția programului, dacă linia 21 este înlocuită cu codul de mai jos?
```
w(a, 'x'); w(b, 'y'); r(a); r(b);
```

d) Ce afișează execuția programului, dacă linia 21 este înlocuită cu codul de mai jos?
```
w(a, 'x'); w(a, 'y'); r(a); r(b);
```

e) Explicați și justificați funcționarea programului, dacă linia 21 este înlocuită la fel ca la punctul d) și, în plus, se elimină linia 22.

2 Răspundeți la următoarele întrebări despre scriptul Shell UNIX de mai jos.

```
1  #!/bin/bash
2  for F in *.sh; do
3    A=`grep "^[ \t]*[^ \t#]" $F | wc -l`
4    B=`wc -l < $F`
5    if [ $A -lt `expr $B - $A` ]; then
6      echo $F
7    fi
8  done
```

a) Ce afișează execuția scriptului?

b) Ce afișează execuția scriptului, dacă e rulat într-un director conținând doar scriptul însuși?

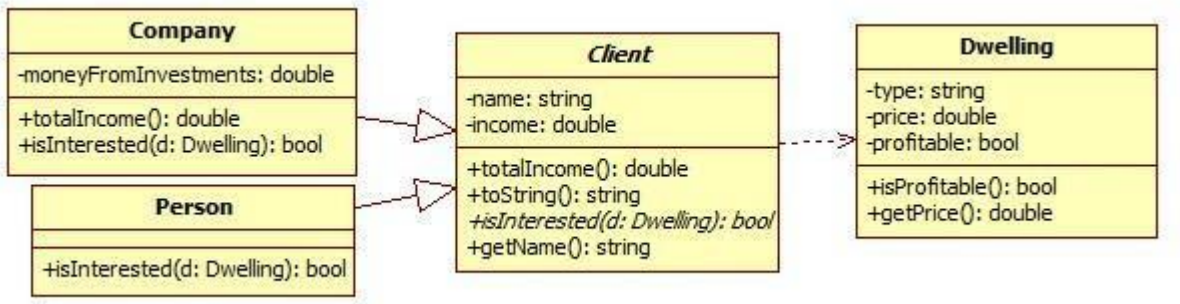
c) Explicați în detaliu expresia regulară de pe linia 3

VARIANTA 1

SUBIECT Algoritmică și programare

Scrieți un program într-unul din limbajele de programare Python, C++, Java, C#, cu următoarele cerințe:

a) Definește clasele **Client** (Client), **Person (**Persoană**)**, **Company** (Companie), **Dwelling (**Locuință**)** pe baza următoarei diagrame UML (constructorii nu sunt indicați pe diagramă). Dintre metodele indicate pe diagramă, se vor implementa doar metodele indicate la punctul b), restul doar se declară.



- Atributul *name* din clasa **Client** (numele clientului) trebuie să aibă cel puțin 3 caractere și *income* (venitul) trebuie să fie o valoare strict pozitivă. Constructorii trebuie să impună constrângerile.
- Clasa abstractă **Client** are o metodă abstractă **isInterested**.
- Venitul total al unei companii este venitul (*income*) la care se adaugă suma obținută din investiții (*moneyFromInvestments*). Venitul total al oricărui alt client este egal cu venitul său.
- O persoană este interesată de o locuință (**Dwelling**) dacă prețul locuinței împărțit la 360 este mai mic decât jumătate din venitul total al persoanei. O companie este interesată de o locuință dacă prețul locuinței împărțit la 12 este mai mic decât venitul total al companiei și dacă locuința e profitabilă.
- Metoda **toString** returnează numele (*name*) concatenat cu venitul total atât pentru persoane cât și pentru companii.

b) Implementează următoarele metode din diagrama de la punctul a): constructorii claselor **Client**, **Person**, **Company**, **Dwelling**; metodele **totalIncome** din clasele **Client** și **Company** și metodele **isInterested** din clasele **Person** și **Company**.

c) Definește o funcție care primește ca parametri o listă de clienți și o relație de ordine între clienți (furnizată sub forma unei funcții booleene care compară doi clienți) și ordonează lista de clienți în ordinea dată de relație. Sortarea se va implementa folosind un algoritm având complexitatea timp $\theta(n \cdot log_2 n)$.

d) Definește o funcție care primește ca parametri un dicționar de clienți (cheia – numele clientului, valoarea – clientul) și o locuință și returnează, folosind funcția de la punctul c), o listă cu toți clienții care sunt interesați de acea locuință, ordonați alfabetic după nume.

e) Construiește în funcția principală (main) un dicționar de clienți și adaugă următorii clienți (alegeți valori pentru proprietățile lor neprecizate): două companii, una numită "TwoStar" și una numită "Fort" și două persoane, una numită "Ana" și alta "Mihai". Construiți două locuințe: una are tipul "apartament", costă 150000 și nu e profitabilă; a doua este o "casă", costă 500000 și este profitabilă. Pentru dicționarul de clienți și fiecare locuință, apelați funcția de la punctul d). Afișați apoi clienții din listele rezultate, folosind metoda **toString**.

f) Pentru tipul de dată **Dicționar**, scrieți specificația operațiilor de **adăugare** și **căutare**.

- **Se va indica limbajul de programare folosit.**
- **Nu se vor defini alte metode decât cele specificate în diagramă (exceptând constructorii).**
- **Nu se vor folosi containere sortate și operații de sortare predefinite.**

*Pentru **tipurile de date** puteți folosi biblioteci existente (Python, C++, Java, C#).*

VARIANTA 1

SUBIECT Baze de date

Fie o bază de date care stochează datele gestionate de o aplicație de vot electronic pentru un referendum. Baza de date are următoarea structură:

- tabelul ***Persoane*** cu câmpurile **CodP**, **Nume**, **CNP**, **CodLocalitate**, **CodJudeț**, **CodRegiune, Votat**;
- tabelul ***Voturi*** cu câmpurile **CodV**, **Data, Ora**;
- tabelul ***Răspunsuri*** cu câmpurile **CodV**, **CodI**, **RăspunsDa, RăspunsNu, Anulat**;
- tabelul ***Intrebări*** cu câmpurile **CodI**, **Text, NumeCategorie**, **DescriereCategorie**.

Un răspuns pentru o întrebare este anulat dacă nu a fost selectat nici *Da* nici *Nu* sau au fost selectate ambele variante. Câmpurile **RăspunsDa**, **RăspunsNu**, **Anulat** și **Votat** pot lua doar valorile 0 sau 1.

1. Determinați cheile candidat și cheile externe pentru fiecare dintre tabelele de mai sus.
2. Determinați cel puțin 4 dependențe funcționale care se referă la câmpuri care nu reprezintă chei candidat.
3. Considerați următoarele modificări de structură și precizați care dintre ele sunt esențiale pentru ca baza de date să fie în **3NF** (a 3-a formă normală). În cazul unui răspuns afirmativ, justificați-vă opțiunea.
   a. Crearea unui tabel separat pentru stocarea categoriilor.
   b. Eliminarea câmpului **Anulat** din tabelul ***Răspunsuri***.
   c. Utilizarea în tabelul ***Voturi*** a unui singur câmp care să memoreze atât data cât și ora la care s-a realizat votarea.
   d. Adăugarea unui câmp **CodP** în tabelul ***Voturi*** care să refere înregistrări ale tabelului **Persoane.**
4. Scrieți, pe structura dată, o interogare SQL echivalentă cu următoarea interogare:

$$\Pi_{\text{Text, Ora}} (\sigma_{\text{Ora} >18:00} (\textbf{\textit{Voturi}} \otimes_{\text{Voturi.CodV=Răspunsuri.CodV}} \textbf{\textit{Răspunsuri}} \otimes_{\text{Întrebări.CodI=Răspunsuri.CodI}} \sigma_{\text{NumeCategorie='UE'}} (\textbf{\textit{Întrebări}})))$$

5. Scrieți, pe structura dată, o interogare SQL care returnează pentru fiecare întrebare numărul total de răspunsuri, numărul total de răspunsuri *Da* valide, numărul total de răspunsuri *Nu* valide și numărul total de răspunsuri anulate (***Text, NrRaspunsuri, NrDa, NrNu, NrAnulate***).

BAREM INFORMATICĂ
VARIANTA 1

Subiect (Algoritmică şi Programare)

Oficiu – 1p
Definirea clasei abstracte Client – 0.6p din care
- atribute – 0.1
- constructor si metode – 0.5

Definirea clasei Company– 1.5p din care
- relația de moștenire – 0.2
- atribut – 0.1
- constructor – 0.4
- metode – 0.8

Definirea clasei Person – 0.7p din care
- relația de moștenire – 0.2
- constructor – 0.2
- metoda – 0.3

Definirea clasei Dwelling – 0.2p din care
- atribute– 0.1
- constructor - 0.1

Funcția de la punctul c) – 3p din care
- signatura corectă - 0.4p
- sortare in $\theta(n \cdot log_2 n)$ - 2.5p
- returnare listă rezultat – 0.1

Funcția de la punctul d) – 1.5p din care
- signatura corectă - 0.1p
- construire lista clienti in ordine alfabetica dupa nume – 1.3p
- returnare listă rezultat – 0.1

Funcția principală e) – 0.5p
f) Specificaţiile operaţiilor **adăugare** și **căutare** pentru tipul de dată **Dicționar** – 1p

Subiect Baze de date

1. 0.5p (chei candidat) + 0.5p (chei externe) = 1p
2. 0.25p x 4 = 1p
3. a, b
   - 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 procesul părinte având patru procese fiu – 1p
1.b Va afișa “x y 4” sau “y x 4” – 1p
1.c Identic cu punctul 3.1.b – 1p
1.d Va afișa “x 4” sau “y 4” – 1p
1.e Va afișa “x” sau “y” și se va bloca la linia 14 pentru că pipe-ul este deschis pentru scriere – 1.5p

2.a Numele scripturilor Shell din directorul curent cu mai puține linii de cod decât restul liniilor din script – 1p
2.b Nu va afișa nimic – 1p
2.c Inceput de linie, 0 sau mai multe spații sau taburi, orice caracter care nu e spațiu, tab sau diez – 1.5p