Universitatea Politehnica din București
Facultatea de Automatică și Calculatoare
Admitere Master
Varianta 5

1. Limbaje de Programare

1. Care dintre următoarele fragmente reprezintă o relație de tip HAS-A între A și B?

A. class A extends B { ...}
B. class A implements B {... }
C. class A { private B myb; ... }
D. abstract class B extends A { ... }

2. Ce afișează la rulare următoarea secvență de cod?

```
class Test2 {
   static void exception(int x) {
       try {
           System.out.print("A");
           int y = 42 / x;
           return;
       } catch (Exception e) {
           System.out.print("B"); return;
       } finally { System.out.print("C"); }
  }
  public static void main(String[] args) {
       exception(0);
  }
}
```

A. nu afișează nimic deoarece apare o excepție
B. ABC
C. AB
D. A

3. Care dintre următoarele declarații este admisibilă?

```
class A { }
 class B extends A { }
 class C extends B { }
 public class Carpet<V extends B> {
 public <X extends V> Carpet<? extends V>
          method(Carpet<? super X> e) {
 // introduceti codul aici
} }
```

A. return new Carpet<V>();
B. return new Carpet<A>();
C. return new Carpet<B>();
D. return new Carpet<C>();

4. Fie următoarele definiții de clase:

```
class D {int method() {return 1;}}
class E extends D {int method() {return 2;}}
class F extends E {int fun(D d) {return d.method();}}
```

Care este rezultatul următoarei secvențe de cod?

```
F x = new F(); System.out.println(x.fun(x));
```

A. 1
B. 2
C. Eroare de compilare
D. Eroare la rulare

5. Care este rezultatul următoarei secvențe de cod?

```
1. public static void main(String[] args) {
2. try { throw new Error(); }
3. catch (Error e) {
4.     try { throw new RuntimeException(); }
5.     catch (Throwable t) { }
6. }
7. System.out.println("oioioi"); }
```

A. oioioi
B. Eroare de compilare la linia 2
C. Eroare de compilare la linia 4
D. Eroare de compilare la linia 5

6. Fie următoarele definiții de clase:

```
abstract class A {
  int met(A a) { return 0; }
  int met(B b) { return 1; }
  int met(C c) { return 2; } }
class B extends A {
  int met(A a) { return 3; }
  int met(B b) { return 4; }
  int met(C c) { return 5; } }
class C extends B {
  int fun() { return ((A)this).met((A)this); } }
```

Care este rezultatul următoarei secvențe de cod?

```
C x = new C(); System.out.println(x.fun());
```

A. 3
B. 0
C. 5
D. 2

7. Care dintre următoarele cuvinte cheie NU poate apărea în definiția unei metode?

A. implements
B. void
C. static
D. private

8. Care este rezultatul următoarei secvențe de cod?

```
class F {
 static String s = "hello";
 public static void main(String[] args) {
   new F().met();
   System.out.println(s); }
 void met() { s = "world"; }
}
```

A. hello
B. world
C. Eroare de compilare
D. Eroare la rulare

2. Algoritmi

1. Ce structură de date este indicat să se folosească pentru grafuri rare în cazul în care se dorește aplicarea algoritmului lui Dijkstra și care este cea mai bună complexitate care se poate obține?

A. matrice de adiacență - O(|V|*|E|)

B. heap binar - O(|V|^2*log|V|)

C. vectori - O(|V|*log|V|)

D. heap Fibonnaci - O(|E|+|V|*log|V|)

2. Problema ciclului hamiltonian se poate rezolva cel mai eficient cu un algoritm nedeterminist cu o complexitate temporală:

A. Exponențială

B. Polinomială

C. Logaritmică

D. Nu se poate rezolva cu un algoritm nedeterminist

3. Problema terminării programelor (Halt) este:

A. în clasa P

B. în clasa NP

C. o problemă NP-dură

D. o problemă NP-completă

4. Fie Alg un algoritm pentru o problemă de decizie arbitrară Q. Fie problemă de decizie R: "Alg(n) intoarce valoarea n ?" Atunci:

A. R este semi-decidabilă

B. R este decidabilă

C. R se rezolvă in timp polinomial

D. R se rezolvă in timp exponențial

5. Ce efect are urmatorul cod aplicat unui graf G(V, E)? (w:E --> R; d:V --> R)

```
Pentru i de la 1 la |V|-1
   Pentru fiecare (u,v) din E
      Daca d[v] > d[u] + w(u,v) atunci
        d[v] = d[u] + w(u,v)
```

A. determină arborele minim de acoperire al grafului

B. determină drumul de cost minim din graf

C. determină fluxul maxim din graf

D. realizează o sortare topologică a grafului

6. Care este complexitatea temporală pentru următoarea porțiune de algoritm (unde *i* și *n* sunt variabile de tip întreg (*n* > 1), iar *v* este un vector de numere întregi)?

```
i = 1
while (i <= n) {
  v[i] = 0
  i = i + 1
}
```

A. $\Theta(1)$

B. $\Theta(n^2)$

C. $\Theta(n)$

D. $\Theta(n * \lg n)$

7. Se știe că o anumită problema dată nu poate fi rezolvată cu o complexitate polinomială și o primă variantă propusă de rezolvare a fost cu o schemă de backtracking cronologic. Ca variantă de ameliorare a complexității ați propune o abordare bazata pe:

A. Programare dinamică

B. Algoritmi greedy

C. Propagare de restricții

D. Divide și stăpânește

8. Problema construirii arborelui binar optim la căutare în funcție de frecvența căutării cheilor este un exemplu de problemă rezolvabilă corect și eficient printr-un algoritm care folosește:

A. tehnica divide și stăpânește

B. tehnica de programare lacomă

C. orice algoritm eficient de înmulțire a două matrici poate rezolva această problemă, fără a fi necesare alte metode sau algoritmi specifici

D. tehnica programării dinamice

3. Calculatoare Numerice

1. Se consideră următoarele secvențe de cod:

```
P11: lw $1, 40($6)
P12: beq $2,$0, Label: pp $2=$0
P13: sw $6, 50($2)
Label: add $2, $3,$4
     sw $3, 50($4)

P21: lw $5, -16($5)
P22: sw $4, -16($4)
P23: lw $3, -20($4)
P24: beq $2, $0 Label: pp $2 != $0
P25: add $5, $1, $4
```

Se presupune că toate salturile sunt perfect predictibile (nu avem hazarduri structurale). Dacă avem o singură memorie (instrucțiuni și date) există un hazard structural ori de câte ori citim o instrucțiune în același ciclu de ceas în care o instrucțiune accesează datele. Acest tip de hazard va fi rezolvat întotdeauna în favoarea instrucțiunii care accesează date. Care este timpul total de execuție a instrucțiunilor prin banda de asamblare presupunând că avem o singură memorie și că banda de asamblare are 5 stagii?

A. 9 ciclii pentru primul program și 12 ciclii pentru cel de al doilea program
B. 5 ciclii pentru primul program și 10 ciclii pentru cel de al doilea program
C. 9 ciclii pentru primul program și 9 ciclii pentru cel de al doilea program
D. 10 ciclii pentru primul program și 5 ciclii pentru cel de al doilea program

2. Se consideră programul de mai jos. Să se specifice dacă există hazard precum și tipul său.

```
add $s0, $t0, $t1
sub $t2, $s0, $t3
```

A. Programul prezentat nu conține nici un tip de hazard
B. Programul prezentat conține harzard de date
C. Programul prezentat conține hazard structural
D. Programul prezentat conține hazard de control

3. La implementarea memoriei unui calculator sub forma unei ierarhii de memorie se folosește principiul:

A. Localizării temporale
B. Localizării spațiale
C. Ambele principii sunt folosite
D. Nici un principiu nu este folosit

4. În cazul unei benzi de asamblare, care din afirmațiile de mai jos sunt corecte ?

A. Între stagii trebuie să existe registre deoarece nu toate stagiile aceeași durată
B. Între stagii trebuie să existe sumatoare pentru a calcula anumite valori
C. Între stagii nu trebuie să existe nimic
D. Banda de asamblare nu este împărțită în stagii

5. Stagiile de execuție ale unei instrucțiuni pentru MIPS în bandă de asamblare sunt:

A. IF, ID, IE, MEM
B. IF, ID, IE, MEM, WB
C. IF, ID
D. MEM, WB

6. În cazul în care scrierea într-o memorie cache se face prin metoda WRITE-THROUGH mai este nevoie de dirty-bit ?

A. Nu mai este nevoie de dirty-bit
B. Este nevoie de dirty-bit
C. Este nevoie de un dirty-bit duplicat
D. Este nevoie de dirty-bit la care se adaugă valoarea constantă 1

7. Se consideră următorul program. Să se specifice dacă există hazard și de ce.

```
lw $t1, 0($t0);
lw $t2, 4($t0);
add $t3, $t1, $t2;
sw $t3, 12($t0);
lw $t4, 8($t0);
add $t5, $t1, $t4;
sw $t5, 16($t0)
```

A. Programul nu conține hazard
B. Amândouă instrucțiunile add prezintă hazard deoarece prezintă o dependință față de instrucțiunea precedentă lw
C. Amândouă instrucțiunile sw prezintă hazard deoarece prezintă o dependință față de instrucțiunea precedentă add
D. Cea de a doua instrucțiune lw introduce un hazard

8. Se consideră o memorie cache având 4 blocuri de câte 1 cuvânt. Se consideră că memoria este set asociativă cu 2 căi. Să se determine numărul de MISS-uri și cel de HIT-uri având în vedere următoarea secvență de adrese de bloc: 0, 8, 0, 6, 8

A. 4 MISS-uri și 1 HIT
B. 3 MISS-uri și 2 HIT-uri
C. 1 MISS și 4 HIT-uri
D. 2 MISS-uri și 3 HIT-uri

4. Baze de date

1. În modelul entitate asociere, asocierile pot fi:

A. Între atribute
B. Între entităţi
C. Între atribute şi/sau entităţi
D. Între alte asocieri

2. Fie R=ABCDE avand dependenţele
F = {A→C, B→A, C→B, E→D}.
Câte chei distincte are R?

A. 1
B. 2
C. 3
D. 4

3. Fie R=ABCDE având dependenţele
F = {A→C, B→A, C→B, E→D} şi mulţimea
G = {A→B, B→C, C→A, D→E}.
Care afirmaţie este adevarată:

A. F e echivalentă cu G
B. F nu e echivalentă cu G
C. F acoperă pe G
D. G acoperă pe F

4. Într-un join de tip NATURAL JOIN condiţia de join se realizează:

A. Numai după coloane de tip numeric
B. Numai după coloane cu acelaşi nume
C. Numai după coloane definite în cheile tabelelor
D. Numai după coloane care fac parte din indecşi

5. O planificare este:

A. O singură execuţie a unui program
B. Ordinea în care se execută paşii unui set de tranzacţii
C. Un proces care stabileşte când se execută paşii unor tranzacţii
D. Un pas al unei tranzacţii.

6. În modelul entitate asociere, rolurile se pot asocia:

A. Ramurilor care merg la aceeaşi entitate
B. Atributelor de descriere
C. Atributelor de identificare
D. Tabelelor rezultate din transformare în model relaţional

7. Dacă tabela EMP are 14 linii, iar tabela DEPT are 4 linii, atunci cererea
SELECT * FROM EMP, DEPT returnează un rezultat având:

A. 14 linii
B. 4 linii
C. 56 linii
D. 18 linii

8. Dacă o subcerere SQL returnează o pereche de coloane, atunci în clauza WHERE a cererii principale acest rezultat se poate compara folosind operatorul:

A. =
B. IN
C. LIKE
D. BETWEEN

5. Arhitectura sistemelor de calcul

1. Implementarea superscalara a Instruction Level Parallelism-ului inseamna:

A. In acelasi ciclu de ceas, procesorul scrie rezultatul unei instructiuni in registre, executa operatia aritmetica a instructiunii urmatoare, si citeste operanzii instructiunii de dupa instructiunea urmatoare.
B. Se lanseaza doar cate o instructiune intr-un ciclu de ceas al procesorului.
C. Lanseaza mai multe instructiuni in acelasi ciclu de ceas. Dependenta de date este verificata de hardware aditional. Daca nu pot fi lansate in paralel, instructiunile se executa secvential.
D. Instructiunile sunt reordonate in timp ce sunt executate. In modul acesta, pot sa se gaseasca usor instructiuni care nu au dependenta de date intre ele, pentru a fi executate simultan.

2. Retelel de comutatie de tip Delta sunt comutatoare ierarhice cu

A. a intrari si b iesiri ce utilizeaza CrossBar-uri (CB) axb si retele de permutare de tip intercalare inverse Shuffle pe n-1 nivele ierarhive

B. a^n intrari si b^n iesiri ce utilizeaza retele trunchi-K axb si retele de permutare de tip butterfly pe n nivele ierarhice

C. a intrari si b iesiri ce utilizeaza retele trunchi-K axb si retele de permutare de tip bit-reversal pe n-1 nivele ierarhice

D. a^n intrari si b^n iesiri ce utilizeaza CrossBar-uri (CB) axb si retele de permutare de tip intercalare perfecta (Shuffle) pe n nivele ierarhice

3. Ce calculeaza codul:

```
SUM[k] = 0 (0 ≤ k ≤ n − 1)
for i = 0 to n − 1
   LOCINDEX[k] = i (0 ≤ k ≤ n − 1)
   SUM[k] += A[LOCINDEX[k],k] (0 ≤ k ≤ n − 1)
end i loop
```

unde *LOCINDEX[k]* e un index local din Procesorul k

A. Suma elementelor unui vector
B. Suma pe linie a elementelor unei matrice
C. Suma pe coloane a elementelor unei matrice
D. Suma tuturor elementelor unei matrice

4. Pentru a avea acces simultan la datele dintr-o matrice atat pe linie cat si pe coloana simultan este necesar sa organizam datele:

A. Pe linii
B. Pe coloane
C. Deplasat circular
D. Oricum

5. In sistemele slab cuplate, comunicarea intre procesoare se face prin:

A. Transfer de mesaje sau comutare de circuite
B. Transfer DMA
C. Transfer Remote DMA
D. Transfer prin memoria globala la o rata de transfer similara cu a memoriei locale

6. Dorim sa efectuam operatia de inmultire a doua matrice C= AxB (matricele au dimensiunea n, A[n,n], B[n,n]) si avem n procesoare intr-o structura SIMD. Pentru a obtine o complexitate a algoritmului de inmultire de $O(n^2)$ este necesar sa avem:

A. Unitatea de comanda sa dispuna de instructiunea BROADCAST
B. Sistemul sa dispuna de un crossbar switch Procesor -Procesor
C. Ambele de mai sus
D. Nici una de mai sus

7. Permutarea fundamentala descrisa prin modificarea adresei resurselor sub forma urmatoare: $(a_n \dots a_1 a_0) = (a_0 a_{n-1} \dots a_1 a_n)$ Reprezinta:

A. Permutarea buterfly
B. Permutarea cu intercalare perfecta
C. Permutarea de baza
D. Permutarea cu ordine inversa

8. Daca in bucla interioara a operatiei de inmultire a doua matrice avem de executat:

```
c[i][j] += a[i][k] * b[k][j]
```

modul de acces corect si optim al elementelor din cele trei matrice a, b si c in functie de ordinea buclelor este:

A. for (i) – constant; for (j) – constant; for (k) – nesecvential
B. for (i) – secvential; for (k) – constant; for (j) – secvential
C. for (k) – constant; for (i) – constant; for (j) – secvential
D. for (k) – nesecvential; for (j) – constant; for (i) – nesecvential