Examen de licenţă 2024 - Informatică
– Exemple subiecte –

Proba de cunoştinţe a examenului de licenţă din sesiunile iulie/septembrie 2024 se va desfășura sub forma unei examinări orale care va consta din 3 întrebări cu răspuns multiplu pentru care e necesară o justificare orală a alegerii răspunsurilor. Tematicile din care vor fi formulate întrebările sunt: Structuri discrete şi algoritmi (proiectarea și analiza algoritmilor, structuri de date, teoria grafurilor, logică computațională, limbaje formale); Limbaje de programare şi inginerie software (limbaje de programare (Python, C, C++, Java), proiectarea aplicațiilor software, baze de date); Sisteme de calcul (arhitectura calculatoarelor, sisteme de operare, reţele de calculatoare). Pentru fiecare dintre cele trei întrebări se poate acorda maxim 3 puncte, 1 punct fiind acordat din oficiu.

Pentru neclarităţi privind enunţurile sau răspunsurile puteţi să vă adresaţi celor care au propus întrebările pentru fiecare secţiune şi la flavia.micota@e-uvt.ro.

Tematica 1: Structuri discrete și algoritmi

- **Proiectarea şi analiza algoritmilor**
  - Daniela Zaharie (daniela.zaharie@e-uvt.ro)
  - Adrian Spătaru (adrian.spataru@e-uvt.ro)
- **Structuri de date**
  - Cosmin Bonchiş (cosmin.bonchis@e-uvt.ro)
  - Adrian Spătaru (adrian.spataru@e-uvt.ro)
- **Teoria grafurilor**
  - Mircea Marin (mircea.marin@e-uvt.ro)
  - Isabela Drămnesc (isabela.dramnesc@e-uvt.ro)
- **Logică computaţională**
  - Adrian Crăciun (adrian.craciun@e-uvt.ro)
  - Alexandra Fortiş (alexandra.fortis@e-uvt.ro)
- **Limbaje formale**
  - Mircea Drăgan (mircea.dragan@e-uvt.ro)
  - Mădălina Eraşcu (madalina.erascu@e-uvt.ro)

  - Alexandra Fortiş (alexandra.fortis@e-uvt.ro)

Tematica 2: Limbaje de programare și inginerie software

- **Limbaje de programare (Python, C, C++, Java)**
  - Flavia Micota (flavia.micota@e-uvt.ro)
  - Daniel Pop (daniel.pop@e-uvt.ro)
  - Todor Ivașcu (todor.ivascu@e-uvt.ro)
- **Proiectarea aplicaţiilor software**
  - Cristina Mîndruţă (cristina.mindruta@e-uvt.ro)
  - Daniel Pop (daniel.pop@e-uvt.ro)
- **Baze de date**
  - Ioan Drăgan (ioan.dragan@e-uvt.ro)
  - Daniel Pop (daniel.pop@e-uvt.ro)

Tematica 3: Sisteme de calcul

- **Arhitectura calculatoarelor**
  - Liviu Mafteiu Scai (liviu.mafteiu@e-uvt.ro)
  - Cristian Cira (cristian.cira@e-uvt.ro)
- **Sisteme de operare**
  - Florin Fortiş (florin.fortis@e-uvt.ro)
  - Ciprian Pungilă (ciprian.pungila@e-uvt.ro)
  - Darius Galiș (darius.galis@e-uvt.ro)
- **Reţele de calculatoare**
  - Mario Reja (mario.reja@e-uvt.ro)
  - Mihail Găianu (mihail.gaianu@e-uvt.ro)

Tematică

Tematica 1: Structuri discrete și algoritmi

Tematica include teme discutate la cursurile de: algoritmi, structuri de date, teoria grafurilor şi combinatorică, logică, limbaje formale:

- Algoritmică: analiza complexității algoritmilor, algoritmi de căutare, algoritmi de sortare, algoritmi recursivi, tehnici de proiectare a algoritmilor (divide et impera, greedy, programare dinamică)
- Structuri de date: stive, cozi, liste, arbori, dicționare
- Teoria grafurilor și combinatorică:

  - Elemente de combinatorică: principii de numărare; numere Stirling; grupuri de simetrii, teoria lui Polya.
  - Elemente de Teoria Grafurilor: noţiuni şi definiţii de bază; Clase speciale de grafuri; conectivitate, distanţe drumuri minime; arbori de acoperire; reţele de transport, fluxuri; colorări.

- Logică computațională: Recunoaşterea/parcurgerea expresiilor (logica propoziţiilor/predicatelor). Semantica, calculul valorii expresiilor (logica propoziţiilor/predicatelor). Tabele de adevăr (logica propoziţiilor). Validitate/satisfiabilitate, consecinţă logică, echivalenţă logică (logica propoziţiilor/predicatelor). Raţionament, rolul raţionamentului. Teorema de deducţie. Forme normale ale formulelor propoziţionale. Rezoluţie, DP, DPLL. Raţionament în stil natural (logica propoziţiilor/logica predicatelor). Aplicaţii ale logicii: design de circuite digitale.

- Limbaje formale și teoria automatelor: limbaje, gramatici, expresii regulate și automate finite

Tematica 2: Limbaje de programare şi inginerie software

Tematica include teme discutate la disciplinele de: Programare (Python, C, C++, Java), inginerie software şi baze de date:

- Limbaje de programare: tipuri de date, clase, obiecte, relații între clase (moștenire, agregare, compoziție, dependență)

- Baze de date: modelarea unei baze de date, forme normale, interogări SQL

- Inginerie Software: activitățile procesului de dezvoltare de software, metode agile de dezvoltare de software, diagrame UML.

Tematica 3: Sisteme de calcul

Tematica include teme discutate la disciplinele: arhitectura calculatoarelor, sisteme de operare, reţele de calculatoare:

- Arhitectura calculatoarelor: structura unui sistem de calcul, unitatea centrală de prelucrare, memoria unui sistem de calcul, dispozitivele periferice ale unui sistem de calcul, magistralele unui sistem de calcul, reprezentarea numerelor în calculator

- Sisteme de operare: accesul concurent la resurse (problema secțiunii critice), probleme de comunicare, algoritmi de planificare CPU, algoritmi de paginare, algoritmi de detecţie și evitare a impasului

- Rețele de calculatoare: încapsularea protocoalelor, comunicare orientată pe datagrame, retransmisie

Bibliografie

- Bruno Buchberger, *Logic for Computer Science, Unpublished Lecture Notes*, Copyright Bruno Buchberger, 1991, http://staff.fmi.uvt.ro/ adrian.craciun/lectures/logica/pdf/buchberger-logic.pdf

- Mordechai Ben-Ari, *Mathematical Logic for Computer Science*, Ed. Springer Verlag London, 2009
- Adrian Crăciun, *Logic for Computer Science (note de curs)*, http://staff.fmi.uvt.ro/~adrian.craciun
- Mircea Marin, *Graph Theory and Combinatorics (note de curs)*, http://staff.fmi.uvt.ro/~mircea.marin
- Th. Cormen, Ch. Leiserson, R. Rivest, C. Stein, *Introduction in algorithms*, Ed. MIT Press, 2008; (trad. lb română) Ed. Teora, 1995
- Allen B. Downey, Chris Mayfield, *Think Java - How to Think Like a Computer Scientis*, Copyright Allen Downe, 2012
- H. Garcia-Molina, J. Widom, *Database Systems: The Complete Book*, ED. Pearson Education Inc, 2009
- John M. Harris, Jeffry L. Hirst, Michael J. Mossinghoff, *Combinatorics and Graph Theory*, Ed. Springer Science+Business Media, 2008
- J. E. Hopcroft, R. Motwani, J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*, Addison-Wesley Longman Publishing Co., Inc., 2006
- I. Jacobson, H. Lawson, Pan-Wei Ng, P.E. McMahon, M. Goedicke, *The Essentials of Modern Software Engineering*, ACM Books, 2019
- B. Kernighan, D. Ritchie, *The C Programming Language*, Ed. Addison-Wesley, 2001
- D. C. Kozen, *Automata and Computability*, Ed. Springer, 1997
- Dragos-Radu Popescu, *Combinatorica si Teoria Grafurilor*, Ed. Societatea de Ştiinţe Matematice din Romania, 2005
- Ian Sommerville, *Software Engineering 9-th ed.*, Ed. Pearson Education Limited, 2016
- Ian Sommerville, *Engineering software products*, Pearson Education, 2020
- Bjarne Stroustrup, *The C++ Programming Language*, ED. Pearson Education, 2005
- Andrew S. Tanenbaum, Todd Austin, *Structured Computer Organization*, Ed. Pearson, 2013
- Andrew S. Tanenbaum, *Computer Networks*, Ed. Pearson, 2011
- Andrew S. Tanenbaum, *Modern Operating Systems*, Ed. Pearson, 2009
- http://agilemodeling.com/essays/umlDiagrams.htm

Algoritmi și structuri de date

1. Care dintre următoarele afirmaţii este/sunt adevărată/adevărate pentru algoritmul corespunzător funcţiei de mai jos (se consideră că x este un tablou unidimensional cu n elemente)

```
def alg(x):
  n=len(x)
  nr=0
  i=0
  while(i<n):
     k=0
     while(((i+k)<n)and(x[i+k]>0)):
        k=k+1
     if (nr<k):
        nr=k
     i=i+k+1
  return(nr)
```

   (a) Algoritmul returnează numărul de elemente pozitive din x
   (b) Algoritmul returnează numărul de elemente din cea mai lungă subsecvenţă cu elemente pozitive din x
   (c) Algoritmul are ordinul de complexitate $\mathcal{O}(n^2)$
   (d) Algoritmul are ordinul de complexitate $\mathcal{O}(n)$
   (e) Algoritmul are ordinul de complexitate $\mathcal{O}(n \cdot k)$

2. Care dintre următoarele afirmaţii este/sunt adevărată/adevărate pentru algoritmul corespunzător funcţiei de mai jos (se consideră că a este un tablou unidimensional cu n elemente)

```
def transform(a):
  n=len(a)
  for i in range(0,n-1):
    if (a[i]<a[i+1]):
      a[i], a[i+1] = a[i+1], a[i]
   return(a)
```

   (a) Algoritmul sortează descrescător tabloul a
   (b) După aplicarea algoritmului, tabloul a satisface proprietatea $a[i] \geq a[n-1]$ pentru $0 \leq i < n$
   (c) Algoritmul sortează crescător tabloul a
   (d) După aplicarea algoritmului, tabloul a satisface proprietatea $a[i] \leq a[n-1]$ pentru $0 \leq i < n$

3. Se consideră două valori naturale `a` şi `b` şi algoritmul descris prin (`//` specifică operația de determinare a câtului împărțirii întregi):

```
def alg(a,b):
  if ((a==0)or(b==0)):
    return(0)
  elif (a%2 ==0):
    return(alg(a//2, 2*b))
  else:
    return(alg(a//2, 2*b)+b)
```

Care dintre următoarele afirmaţii este/sunt adevărată/adevărate?

(a) este posibil ca succesiunea de apeluri recursive să nu se termine

(b) algoritmul returnează întotdeauna 0

(c) algoritmul returnează produsul `a*b` dacă `a` este par şi `a*b+b` dacă `a` este impar

(d) algoritmul returnează produsul `a*b` indiferent de paritatea lui `a`

(e) dacă $a > 2b$ atunci numărul de operaţii de înmulţire efectuate este mai mic în cazul apelului `alg(b,a)` decât în cazul apelului `alg(a,b)`.

4. Se consideră un tablou unidimensional, $a[0..n-1]$, cu $n = 10^{10}$ valori numerice şi se pune problema determinării primelor 10 valori în ordine crescătoare. Care dintre următoarele abordări asigură faptul că $a[0..9]$ conține cele mai mici 10 elemente în ordine crescătoare şi este mai eficientă?

(a) se sortează $a[0..n-1]$ crescător folosind algoritmul `quicksort`

(b) se sortează parțial a aplicând algoritmul de sortare prin inserție în care ciclul exterior `for i in range(1,n)` se înlocuieşte cu `for i in range(1,11)`

(c) se sortează parțial a aplicând algoritmul de sortare prin selecție în care ciclul exterior `for i in range(0,n-1)` se înlocuieşte cu `for i in range(0,10)`

5. Se consideră algoritmul `alg` apelat pentru un număr natural `n` şi se notează cu $T(n)$ numărul de operaţii de adunare efectuate.

```
def alg(n):
  if (n<2):
   return(n)
  else:
    return(alg(n//2)+n%2)
```

Care dintre următoarele afirmaţii este(sunt) adevarată(e)?

(a) Algoritmul returnează câte cifre impare are numărul n

(b) Algoritmul returnează numărul de cifre egale cu 1 din reprezentarea binară a lui `n`

(c) $T(n) = 1$ dacă $n < 2$ şi $T(n) = T([n/2]) + n \ MOD \ 2$ dacă $n \geq 2$

(d) $T(n) = 0$ dacă $n < 2$ şi $T(n) = T([n/2]) + 1$ dacă $n \geq 2$

(e) $T(n)$ aparţine lui $\mathcal{O}(n)$

(f) $T(n)$ aparţine lui $\mathcal{O}(\log n)$

6. Algoritmul de sortare rapidă este cel mai potrivit pentru:

   (a) sortarea unei liste mari cu elemente aleatoare

   (b) sortarea unei liste aproape ordonate

   (c) sortarea unei liste scurte

   (d) nu este potrivit pentru nici un fel de sortare

7. Se consideră şirul lui Fibonacci dat de relaţia de recurenţă $F_0 = 0, F_1 = 1$ şi $F_{n+1} = F_n + F_{n-1}$. De câte ori este evaluat F_2 pentru a calcula F_5 în cazul calculului termenului F_n în mod recursiv.

   (a) de 3 ori (b) o dată (c) de 4 ori (d) de 5 ori

8. Care din următoarele afirmaţii NU sunt adevărate în cazul algoritmului de căutare binară:

   (a) Trebuie folosit un tablou sortat

   (b) Trebuie să avem acces direct la elementul din mijloc

   (c) Algoritmul nu este eficient atunci când avem mai mult de 1000 de elemente

   (d) Algoritmul realizează sortarea elementelor unui tablou

9. Se consideră următorul algoritm aplicat unui număr `n` de forma $n = c_k c_{k-1} \ldots c_1 c_0$:

```
s = 0
while n!=0:
    d = n % 10
    s = s+d
    n = n//10
```

   Dacă p este o variabilă (implicită) care contorizează numărul de execuții ale ciclului (contorul ciclului), care este proprietatea invariantă?

   (a) $s = c_0 + c_1 + \ldots + c_p$, $n = c_k c_{k-1} \ldots c_p$

   (b) $s = c_0 + c_1 + \ldots + c_{p-1}$, $n = c_k c_{k-1} \ldots c_p$

   (c) $s = c_0 + c_1 + \ldots + c_{p-1}$, $n = c_k c_{k-1} \ldots c_1 c_0$

   (d) $n = c_k c_{k-1} \ldots c_p$

10. Fie $A = (a_1, ..., a_n)$ un multiset. Pentru a găsi $S = (s_1, ..., s_k)$, un subset al lui A folosind tehnica căutarii local optimale (căutare lacomă – greedy):

(a) La fiecare pas elementul care pare a fi cel mai promițător la pasul respectiv este selectat din A și adăugat la S.

(b) La fiecare pas elementul care pare a fi cel mai promițător la pasul respectiv este selectat din A și adăugat la S. Dacă ulterior nu este bun îl putem înlocui cu o altă componentă.

(c) La fiecare pas elementul care pare a fi cel mai promițător pe ansamblul problemei este selectat din A și adăugat la S.

(d) La fiecare pas un element oarecare este selectat din A și adăugat la S.

11. O listă liniară simplu înlănţuită este:

(a) lista liniară, în care relaţia de ordonare este materializată pe suport printr-un pointer către elementul următor;

(b) o structură de date implementată prin tipul tablou;

(c) o structură de date circulară;

12. Care din următoarele afirmaţii sunt corecte?

(a) Complexitatea (cea mai defavorabilă) pentru operaţiile de inserare, ştergere sau căutare într-un arbore binar de căutare este $O(\log n)$.

(b) Complexitatea (cea mai defavorabilă) pentru operaţiile de inserare, ştergere sau căutare într-un arbore AVL este $O(\log n)$.

(c) Complexitatea (cea mai defavorabilă) pentru operaţiile de inserare, ştergere sau căutare într-un arbore splay tree este $O(\log n)$.

13. Ce structură de date poate fi folosită pentru a implementa o traversare iterativă a unui arbore binar de căutare?

(a) Queue

(b) Stack

(c) Hash Table

(d) Splay tree.

14. Care este complexitatea de inserare a unui element într-o listă înlănţuită *sortată*?

(a) $O(1)$.

(b) $\Theta(1)$.

(c) $O(n)$.

(d) $\Theta(\log n)$.

15. O stivă este:

(a) listă liniară în care inserările se fac la un capăt al listei şi suprimările la celalalt capăt al listei;

(b) o listă liniară de tipul LIFO (Last In First Out);

(c) o listă liniară cu restricţie la intrare;

(d) o listă liniară în care se manipulează mereu elementul cel mai recent introdus.

16. O structură de date de tip coadă este:

(a) structură de tip listă cu restricţie la intrare;

(b) o listă liniară de tipul FIFO (First In First Out);

(c) o lista liniară în care toate operaţiile se efectuează doar la unul din capetele listei;

(d) o listă liniară în care inserările se efectuează la un capăt al listei, iar suprimările şi ori ce alt acces se efectuează la celălalt capăt al listei.

17. În care din următoarele structuri de date, căutarea are complexitate (worst-case) $O(\log n)$?

(a) liste înlănţuite;

(b) heap;

(c) liste skip;

(d) arbori splay.

18. Care din variantele de parcurgere a unui arbore binar de căutare poate fi folosită pentru afişarea tuturor nodurilor ordonate după cheia nodului.

(a) parcurgere breadth-first;

(b) preordine;

(c) inordine;

(d) postordine.

19. Care este înălţimea minimă a unui arbore binar de căutare cu N noduri?

(a) $O(1)$;

(b) $O(N)$;

(c) $O(N \cdot log_2(N))$;

(d) $O(log_2(N))$.

20. Care din afirmaţiile următoare sunt adevărate referitoare la operaţia de căutare într-un arbore binar de căutare cu N noduri:

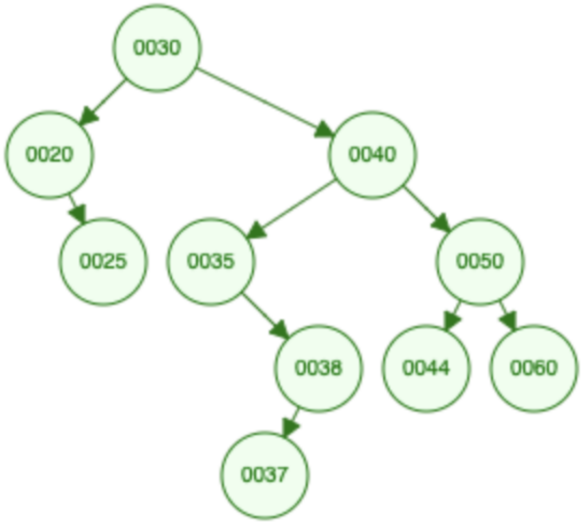
(a) căutarea unui nod se face în $O(log_2(N))$ dacă arborele este echilibrat .

(b) căutarea unui nod se face întotdeauna în $O(1)$ dacă nodul căutat nu există.

(c) căutarea unui nod se face în (worst-case) $O(N)$ dacă arborele nu este echilibrat.

(d) căutarea unui nod se face în $O(N \cdot log_2(N))$.

21. Fie următorul arbore binar de căutare, din care ștergem (prin copiere) nodul 40. Care dintre următoarele afirmații NU sunt adevărate:



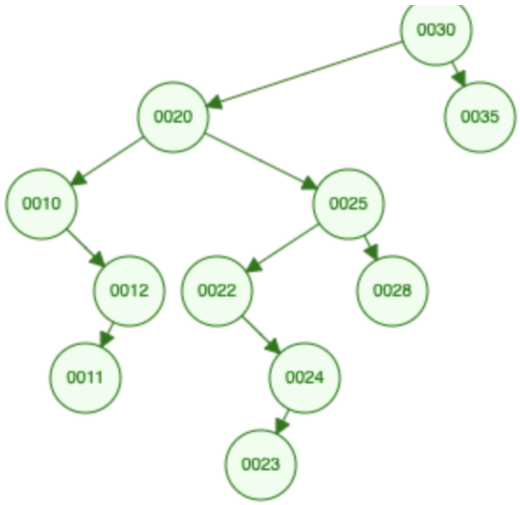
(a) La ștergerea nodului 40, acesta va putea fi înlocuit de nodul 35

(b) La ștergerea nodului 40, acesta va putea fi înlocuit de nodul 37

(c) La ștergerea nodului 40, acesta va putea fi înlocuit de nodul 38

(d) La ștergerea nodului 40, acesta va putea fi înlocuit de nodul 44

22. Fie următorul arbore binar de cautare, din care stergem (prin copiere) nodul 20. Care dintre următoarele afirmații sunt adevărate:



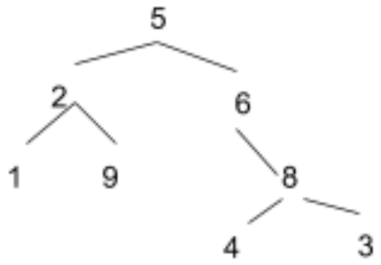
(a) La stergerea nodului 20, acesta va putea fi inlocuit cu nodul 12

(b) La stergerea nodului 20, acesta va putea fi inlocuit cu nodul 11

(c) La stergerea nodului 20, acesta va putea fi inlocuit cu nodul 23

(d) La stergerea nodului 20, acesta va putea fi inlocuit cu nodul 22

23. Fie urmatorul arbore binar. Parcurgerea in postordine corecta a acestui arbore este:



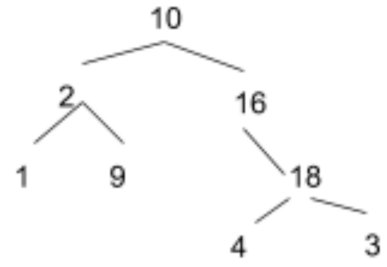
(a) 5,2,1,9,6,8,4,3

(b) 1,9,2,4,3,8,6,5

(c) 5,2,1,9,4,6,8,3

(d) 1,9,2,5,3,8,6,4

24. Fie urmatorul arbore binar. Parcurgerea in inordine a acestui arbore este:



(a) 1,9,2,4,3,18,16,10

(b) 10,2,1,9,16,18,4,3

(c) 1,2,9,10,16,4,18,3

(d) 1,2,3,4,9,10,16,18

25. Un min-heap binar este o structură de date care modelează un arbore binar aproape complet care are *proprietatea de heap*: Pentru orice nod N, dacă P este părintele lui N atunci cheia lui P este mai mică decât cheia lui N.

    Implementarea cu tablou a unui min-heap binar cu n noduri este un tablou $A[0..2^m - 1]$ cu două atribute suplimentare: capacitatea $A.length = 2^m - 1$ şi mărimea $A.size = n$, astfel încât $A.size \leq A.length$. Elementele din nodurile min-heap-ului binar sunt reţinute în primele n elemente ale lui A: rădăcina este reţinută în $A[0]$, iar dacă N este fiul stâng (resp. drept) al unui nod P reţinut în $A[i]$ atunci N este reţinut în $A[2 \cdot i + 1]$ (resp. $A[2 \cdot i + 2]$).

Pentru $0 \leq i, j < n$ definim relaţia $bunic(i, j)$ dacă $A[j]$ reţine părintele părintelui nodului din $A[i]$.

Formula ce defineşte relaţia $bunic(i, j)$ în un min-heap binar este:

(a) $(4 \cdot j + 3 \leq i) \wedge (i \leq 4 \cdot j + 6)$.

(b) $(4 \cdot i + 3 \leq j) \wedge (j \leq 4 \cdot i + 6)$.

(c) $j = \lfloor i/4 \rfloor$.

(d) $(4 \cdot j + 1 \leq i) \wedge (i \leq 4 \cdot j + 2)$.

(e) $(4 \cdot i + 1 \leq j) \wedge (j \leq 4 \cdot i + 2)$.

26. Un min-heap binar este o structură de date care modelează un arbore binar aproape complet care are *proprietatea de heap*: Pentru orice nod N, dacă P este părintele lui N atunci cheia lui P este mai mică decât cheia lui N.

    Implementarea cu tablou a unui min-heap binar cu n noduri este un tablou $A[0..2^m - 1]$ cu două atribute suplimentare: capacitatea $A.length = 2^m - 1$ şi mărimea $A.size = n$, astfel încât $A.size \leq A.length$. Elementele din nodurile min-heap-ului binar sunt reţinute în primele n elemente ale lui A: rădăcina este reţinută în $A[0]$, iar dacă N este fiul stâng (resp. drept) al unui nod P reţinut în $A[i]$ atunci N este reţinut în $A[2 \cdot i + 1]$ (resp. $A[2 \cdot i + 2]$).

    Pentru $0 \leq i, j < n$ definim relaţia $bunic(i, j)$ dacă $A[j]$ reţine părintele părintelui nodului din $A[i]$.

    Numărul maxim de noduri în un min-heap binar cu adâncimea h este:

    (a) $2^{h+1} - 1$.

    (b) 2^h.

    (c) $h^2 - 1$.

    (d) $2^{h-1} + 1$.

27. Un min-heap binar este o structură de date care modelează un arbore binar aproape complet care are *proprietatea de heap*: Pentru orice nod N, dacă P este părintele lui N atunci cheia lui P este mai mică decât cheia lui N.

    Implementarea cu tablou a unui min-heap binar cu n noduri este un tablou $A[0..2^m - 1]$ cu două atribute suplimentare: capacitatea $A.length = 2^m - 1$ şi mărimea $A.size = n$, astfel încât $A.size \leq A.length$. Elementele din nodurile min-heap-ului binar sunt reţinute în primele n elemente ale lui A: rădăcina este reţinută în $A[0]$, iar dacă N este fiul stâng (resp. drept) al unui nod P reţinut în $A[i]$ atunci N este reţinut în $A[2 \cdot i + 1]$ (resp. $A[2 \cdot i + 2]$).

    Pentru $0 \leq i, j < n$ definim relaţia $bunic(i, j)$ dacă $A[j]$ reţine părintele părintelui nodului din $A[i]$.

    (1) Dacă A este un min-heap binar atunci A este sortat în ordinea crescătoare a cheilor din noduri?

(2) Dacă A este sortat în ordinea crescătoare a cheilor din noduri atunci A este un min-heap binar?

(a) (1) fals (2) adevărat
(b) (1) fals (2) fals
(c) (1) adevărat (2) adevărat
(d) (1) adevărat (2) fals

28. Un min-heap binar este o structură de date care modelează un arbore binar aproape complet care are *proprietatea de heap*: Pentru orice nod N, dacă P este părintele lui N atunci cheia lui P este mai mică decât cheia lui N.

    Implementarea cu tablou a unui min-heap binar cu n noduri este un tablou $A[0..2^m - 1]$ cu două atribute suplimentare: capacitatea $A.length = 2^m - 1$ şi mărimea $A.size = n$, astfel încât $A.size \leq A.length$. Elementele din nodurile min-heap-ului binar sunt reţinute în primele n elemente ale lui A: rădăcina este reţinută în $A[0]$, iar dacă N este fiul stâng (resp. drept) al unui nod P reţinut în $A[i]$ atunci N este reţinut în $A[2 \cdot i + 1]$ (resp. $A[2 \cdot i + 2]$).

    Pentru $0 \leq i, j < n$ definim relaţia $bunic(i, j)$ dacă $A[j]$ reţine părintele părintelui nodului din $A[i]$.

    Timpul de execuţie a operaţiei de ştergere a nodului cu cheie minimă din un min-heap binar cu n noduri este:

    (a) $O(1)$
    (b) $O(\log n)$
    (c) $\Theta(n \log n)$
    (d) $\Theta(n)$

Teoria grafurilor și combinatorică

1. Se dau mulţimile $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, $B = \{\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}\}$ şi $C = \{\mathtt{1}, \mathtt{2}, \mathtt{3}, \mathtt{4}, \mathtt{5}\}$. Câte submulţimi ale mulţimii $A \cup B \cup C$ conţin un element din A, două din B şi cel puţin 4 elemente din C?

   (a) 15 (b) 14 (c) 100 (d) 108

2. Câte numere întregi cuprinse între 1 şi 1000 se divid cu 7, dar nu se divid cu 3?

   (a) 93 (b) 95 (c) 92 (d) 136

3. În câte feluri putem forma un buchet cu cinci trandafiri dacă putem folosi trandafiri roşii, galbeni şi albi? Ordinea punerii trandafirilor în buchet nu este relevantă.

(a) 21 (b) 120 (c) 125 (d) 243 (e) 15

4. Un mesaj transmis pe un canal de comunicare este o secvenţă de 2 tipuri de semnale: semnale de tip A care durează 1 microsecundă, şi semnale de tip B care durează 2 microsecunde. De exemplu, mesajul $ABAAB$ durează $3 \times 1 + 2 \times 2 = 7$ microsecunde.

   Fie a_n numărul de mesaje diferite care durează n microsecunde. Care este valoarea lui a_{10}?

   (a) 89 (b) 55 (c) 2917 (d) 144

5. In internet, format din reţele interconectate de calculatoare, fiecărei conexiuni de reţea dintr-un calculator i se atribuie o adresă internet. Protocolul IPv4 prevede că o adresă internet este un şir de 32 biţi, format din un număr de reţea (*netid*) urmat de un număr de gazdă (*hostid*). Sunt 3 tipuri de adrese internet:

   (a) Clasa A: acestea sunt de forma $0\underbrace{b_1b_2b_3b_4\ldots b_7}_{\text{netid}}\underbrace{b_8b_9\ldots b_{30}b_{31}}_{\text{hostid}}$

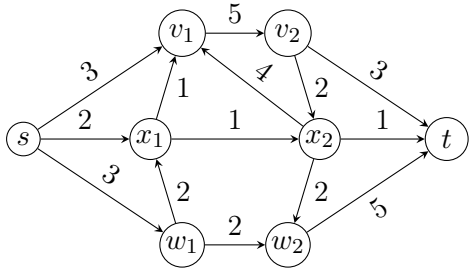
   (b) Clasa B: acestea sunt de forma $10\underbrace{b_2b_3b_4\ldots b_{15}}_{\text{netid}}\underbrace{b_{16}b_{17}\ldots b_{30}b_{31}}_{\text{hostid}}$

   (c) Clasa C: acestea sunt de forma $110\underbrace{b_3b_4b_5\ldots b_{23}}_{\text{netid}}\underbrace{b_{24}b_{25}\ldots b_{30}b_{31}}_{\text{hostid}}$

   Câte adrese IPv4 diferite sunt disponibile pentru conexiunile de reţea din internet?

   (a) $2^{29} \cdot 2^{30} \cdot 2^{31}$ (b) 2^{31} (c) $7 \cdot 2^{29}$ (d) 2^{32}

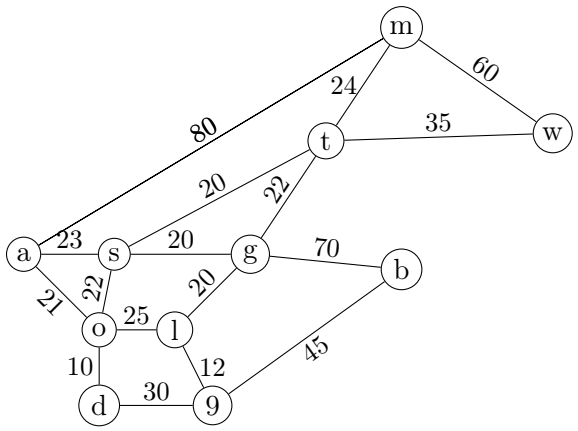
6. Fie reţeaua de transport G cu sursa s şi destinaţia t:



   Care este valoarea fluxului maxim în G?

   (a) 8 (b) 5 (c) 6 (d) 7

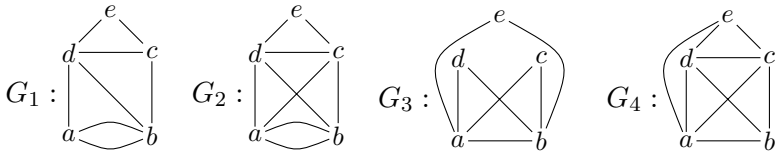
7. Fie graful ponderat



Ce greutate totală are arborele minim de acoperire al acestui graf?

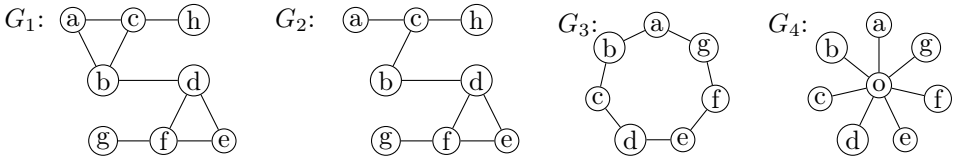
(a) 229 (b) 216 (c) 230 (d) 234

8. Care din grafurile următoare este eulerian?



(a) G_1, G_4 (b) G_2, G_3 (c) nici unul (d) G_4 (e) G_1, G_3

9. Care din grafurile următoare are un cuplaj perfect?



(a) G_1 (b) G_2 (c) G_3 (d) G_4 (e) 216

10. Câţi arbori diferiţi cu 5 noduri, numerotate de la 1 la 5, există?

(a) 10 (b) 273 (c) 32 (d) 120 (e) 125

11. Câte cuplaje maxime are graful bipartit complet $K_{m,n}$ dacă $1 \leq m \leq n$? Observaţi că un astfel de cuplaj trebuie să aibe m muchii.

(a) m (b) $m+n$ (c) $P(n,m)$ (d) $C(n,m)$ (e) m^n (f) n^m

12. Câte muchii are graful complet K_n?

(a) $C(n,2)$ (b) n^2 (c) 2^n (d) $n \cdot (n-1)$ (e) n

13. Se presupune că $n \geq 3$. Câte regiuni are o reprezentare planară a grafului bipartit complet $K_{n,2}$?

(a) $n+1$ (b) n (c) $2 \cdot n$ (d) $C(n,2)$ (e) $P(n,2)$

14. Un şir ternar este un şir care conţine doar cifrele 0,1 şi 2. Fie a_n numărul şirurilor ternare de lungime n care nu conţin apariţii consecutive ale cifrei 0. De exemplu, $a_1 = 3$ şi $a_2 = 8$. Care din formulele următoare are loc pentru orice $n \geq 3$:

(a) $a_n = 2\,a_{n-1} + 2\,a_{n-2}$
(b) $a_n = 2\,a_{n-1} + 2\,a_{n-2} + 3^{n-1}$
(c) $a_n = 2\,a_{n-1} + 3\,a_{n-2}$
(d) $a_n = 2\,a_{n-1} + 3^{n-1}$

15. Care dintre grafurile următoare au numărul cromatic 2:

(a) Orice arbore cu $n \geq 2$ noduri.
(b) Orice graf complet K_n cu număr par de noduri.
(c) Orice graf complet K_n cu număr impar de noduri mai mare sau egal ca 3.
(d) Orice graf ciclic C_n cu $n \geq 2$ număr par.
(e) Orice graf ciclic C_n cu $n \geq 2$ număr impar.
(f) Orice graf bipartit $K_{m,n}$.

16. $A \oplus B = C \in M_n$ dacă $C[i][j] = \max(A[i][j], B[i][j])$ şi $A \odot_k B = D \in M_n$ dacă $D[i][j] = A[i][k] \cdot B[k][j]$

Fie G un graf simplu neorientat cu n noduri numerotate de la 1 la n, şi $A_G \in M_n$ matricea lui de adiacenţă. Numărul de grafuri neorientate simple cu n noduri numerotate de la 1 la n este:

(a) $2^{n(n-1)/2}$
(b) 2^n
(c) $n(n-1)/2$
(d) $n-1$
(e) $n \cdot (n-1)$

17. Fie M_n mulţimea matricilor de dimensiune $n \times n$ cu elemente 0 sau 1. Pentru orice două matrici $A, B \in M_n$ şi $1 \leq k \leq n$ definim operaţiile $A \oplus B$ şi $A \odot_k B$ în felul următor: $A \oplus B = C \in M_n$ dacă $C[i][j] = \max(A[i][j], B[i][j])$ şi $A \odot_k B = D \in M_n$ dacă $D[i][j] = A[i][k] \cdot B[k][j]$ pentru toţi $1 \leq i, j \leq n$.

    Fie G un graf simplu neorientat cu n noduri numerotate de la 1 la n, şi $A_G \in M_n$ matricea lui de adiacenţă.

    Fie I_n matricea identitate de dimensiune $n \times n$, şi $1 \leq p \leq n$. Se consideră algoritmul următor de calcul al matricii $B \in M_n$

    $B = A_G \oplus I_n$;
    for $k := 1$ to p **do**
        $C = B \oplus (B \odot_k B)$;
        $B = C$;

    Complexitatea temporală a calculului matricii B este:

    (a) $\Theta(p \cdot n^2)$
    (b) $\Theta(p \cdot n)$
    (c) $\Theta(p \cdot n^3)$
    (d) $\Theta(n), \Theta(p \cdot 2^n)$

18. Fie M_n mulţimea matricilor de dimensiune $n \times n$ cu elemente 0 sau 1. Pentru orice două matrici $A, B \in M_n$ şi $1 \leq k \leq n$ definim operaţiile $A \oplus B$ şi $A \odot_k B$ în felul următor: $A \oplus B = C \in M_n$ dacă $C[i][j] = \max(A[i][j], B[i][j])$ şi $A \odot_k B = D \in M_n$ dacă $D[i][j] = A[i][k] \cdot B[k][j]$ pentru toţi $1 \leq i, j \leq n$.

    Fie G un graf simplu neorientat cu n noduri numerotate de la 1 la n, şi $A_G \in M_n$ matricea lui de adiacenţă.

    Fie I_n matricea identitate de dimensiune $n \times n$, şi $1 \leq p \leq n$. Se consideră algoritmul următor de calcul al matricii $B \in M_n$

    $B = A_G \oplus I_n$;
    for $k := 1$ to p **do**
        $C = B \oplus (B \odot_k B)$;
        $B = C$;

    Care din afirmaţiile următoare este adevărată când $B[i][j] = 1$?

    (a) Există o cale de lungime cel mult $p + 1$ de la nodul i la nodul j.
    (b) Există o cale de lungime p de la nodul i la nodul j
    (c) Există o cale de la nodul i la nodul j care trece prin nodul p.
    (d) Există o cale de la nodul i la nodul j care trece prin toate nodurile din mulţimea $\{1, 2, \ldots, p\}$.

Logică computațională

1. Consideraţi, în logica predicatelor, limbajul care conţine următoarele simboluri:

   - variabile, pentru care se folosesc litere mici,
   - simboluri funcţionale $\mathcal{F}$: $+$ binar, infix, $-$ unar, prefix, $*$ binar, infix.
   - simboluri predicative $\mathcal{P}$: $=, <, \leq$ toate binare, infix.
   - simboluri pentru constante $\mathcal{C}$: $0, 1$.

   Care din următoarele sunt termeni peste acest limbaj?

   (a) $(0 * x) - 1$,
   (b) $1 + (z * x) < 0$,
   (c) $x + ((-1) * 0)$,
   (d) $0 * (y + 1)$.

2. Pentru următoarele formule propoziţionale, şi pentru interpretarea $\{P, \neg Q\}$:

   (a) $((P \Rightarrow Q) \wedge ((\neg Q) \wedge P))$ are valoarea sub interpretare $\mathbb{A}$,
   (b) $((P \Rightarrow Q) \Rightarrow (Q \Rightarrow P)$ are valoarea sub interpretare $\mathbb{A}$,
   (c) $((\neg(P \vee Q)) \wedge (\neg Q))$ are valoarea sub interpretare $\mathbb{F}$.

3. Care din următoarele afirmaţii este adevărată:

   (a) dacă o formulă propoziţională este validă, atunci este satisfiabilă,
   (b) dacă o formulă propoziţională nu este validă, atunci este nesatisfiabilă,
   (c) dacă o formulă propoziţională nu este validă, atunci negaţia sa este satisfiabilă,
   (d) dacă o formulă propoziţională nu este validă, atunci negaţia sa este validă.

4. Care este relaţia dintre propoziţiile

   $$(F \wedge G) \Rightarrow H$$

   şi

   $$F \Rightarrow (G \Rightarrow H)?$$

   (a) sunt logic echivalente
   (b) prima este o consecinţă logică a celei de-a doua,
   (c) a doua este o consecinţă logică a primeia,
   (d) nu se relaţionează în niciun fel descris mai sus.

5. Formula

   $$P \Leftrightarrow Q$$

   este

(a) logic echivalentă cu conjuncţia formulelor de mai jos,

(b) o consecinţă logică a formulelor de mai jos,

(c) logic echivalentă cu disjuncţia formulelor de mai jos,

aceste formule fiind:

$$Q \Rightarrow R,$$
$$R \Rightarrow (P \wedge Q),$$
$$P \Rightarrow (Q \vee R).$$

6. Care din formulele de mai jos sunt în formă normală disjunctivă?

(a) P,

(b) $\neg P \vee Q$,

(c) $P \wedge \neg Q \wedge S$,

(d) $(P \wedge \neg Q \wedge S) \vee \neg S$.

7. Care din următoarele este un rezolvent al clauzelor $\{P, \neg Q, R\}$ şi $\{\neg P, Q, S\}$?

(a) $\emptyset$,

(b) $\{P, \neg P, R, S\}$,

(c) $\{R, S\}$.

8. Pentru a verifica faptul că o formulă G este o consecinţă logică a formulelor $F_1, \ldots, F_n$, se poate:

(a) verifica dacă $(F_1 \wedge \ldots \wedge F_n) \Rightarrow G$ este nesatisfiabilă,

(b) verifica dacă $\neg F_1 \vee \ldots \vee \neg F_n \vee G$ este nesatisfiabilă,

(c) verifica dacă $\neg F_1 \vee \ldots \vee \neg F_n \vee G$ este validă,

(d) verifica dacă $F_1 \wedge \ldots \wedge F_n \wedge \neg G$ este nesatisfiabilă.

9. Există o formulă logic echivalenta cu

$$\left(\begin{array}{c} ((P_1 \Rightarrow (P_2 \vee P_3) \wedge (\neg P_1 \Rightarrow (P_3 \vee P_4)))) \\ \wedge \\ ((P_3 \Rightarrow (\neg P_6)) \wedge (\neg P_3 \Rightarrow (P_4 \Rightarrow P_1))) \\ \wedge \\ (\neg(P_2 \wedge P_5)) \wedge (P_2 \Rightarrow P_5) \end{array} \right) \Rightarrow \neg(P_3 \Rightarrow P_6),$$

care conţine doar conectori propoziţionali din mulţimea:

(a) $\{\neg, \vee\}$,

(b) $\{\vee, \wedge\}$,

(c) $\{|\}$,

(d) $\{\bot, \rightarrow\}$.

unde | este conectorul NŞI (i.e. $P|Q = \neg(P \wedge Q)$).

10. Mulţimea de clauze ce corespunde formulei

$$(\neg P \Rightarrow (Q \wedge R)) \Rightarrow (P \Rightarrow \neg Q)$$

este:

(a) $\{\{\neg P, \neg Q\}\}$,
(b) $\{\{P, \neg Q\}, \{P, R\}, \{\neg Q, R\}\}$,
(c) $\{\{P, \neg Q, \neg R\}, \{P, Q, R\}, \{\neg P, \neg Q, R\}\}$.

11. Consideraţi mulţimea de clauze:

(1) $\{P, Q, \neg R\}$,
(2) $\{\neg P, R\}$,
(3) $\{P, \neg Q, S\}$,
(4) $\{\neg P, \neg Q, \neg R\}$,
(5) $\{P, \neg S\}$.

Formula corespunzătoare acestei mulţimi este:

(a) validă,
(b) satisfiabilă,
(c) nesatisfiabilă.

12. Metoda Davis-Putnam returnează răspunsul satisfiabil:

(a) când se generează clauza vidă,
(b) când se generează mulţimea vidă de clauze,
(c) când nu se pot genera clauze noi, şi clauza vidă nu este în mulţimea de clauze.

13. Pentru a demonstra o formulă G când se cunoaşte o disjuncţie $A \vee B$:

(a) se presupune A şi se demonstrează G, apoi se presupune B şi se demonstrază G,
(b) se presupune A şi se demonstrează G,
(c) se presupune $\neg A$ şi se demonstrează B şi G.

14. Fie P, Q, R variabile propoziţionale. Care din formulele propoziţionale de mai jos corespund funcţiei booleene cu trei argumente care returnează $\mathbb{A}$ dacă argumentele sale reprezintă codificarea binară a unui număr prim este:

(a) $P \wedge Q \wedge \neg R$;
(b) $(\neg P \wedge Q) \vee (P \wedge R)$;
(c) $(\neg P \wedge Q \wedge \neg R) \vee (((\neg P \wedge Q) \vee P) \wedge R)$;

(d) $((\neg P \wedge Q) \vee (P \wedge \neg Q) \vee (P \wedge Q))$.

15. Fie mulţimea de clauze:

$$\{\{\neg P, \neg Q, R\}, \{P, \neg Q, \neg R\}, \{\neg P, R\}, \{P, \neg Q, R\}, \{\neg Q, \neg R\}\}.$$

Primul pas în aplicarea metodei Davis Putnam asupra mulţimii de clauze consistă în:

(a) aplicarea regulii de împărţire (folosind literalul $\neg P$);
(b) aplicarea regulii literalului pur (unde literalul pur este $\neg Q$);
(c) aplicarea unui pas de rezoluţie;
(d) aplicarea regulii clauzei cu un singur literal (folosind ultima clauză).

Limbaje formale şi teoria automatelor

1. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$; $V_T = \{a, b\}$; $P = \{S \to aSb|aAb, A \to aA|\lambda\}$, este:

(a) $L = \{a^m b^n | m \geq n \geq 1, \}$;
(b) $L = \{a^n a^m b^n | n \geq 0, m \geq 0\}$;
(c) $L = \{a^{i+1} b^i | i \geq 1\}$;
(d) $L = \{a^{n+i} b^n | n \geq 1, i \geq 0\}$;

2. Regulile gramaticii $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$, $V_T = \{PCR, PDAR, UDMR\}$, $P = \{S \to PCR|PDAR|UDMR\}$, respectă restricţiile impuse gramaticilor:

(a) regulate (tip 3);
(b) independente de context (tip 2);
(c) dependente de context (tip 1);
(d) de tipul 0, 1, 2, 3;

3. Regulile gramaticii $G = (V_N, V_T, S, P)$, unde $P = \{S \to abc|aAbc,\ Ab \to bA,\ Ac \to Bbcc,\ bB \to Bb,\ aB \to aaA|aa\}$, respectă restricţiile impuse gramaticilor:

(a) regulate (tip 3);
(b) independente de context (tip 2);
(c) dependente de context (tip 1);
(d) de tipul 0;

4. Expresia regulată ce notează limbajul $L = \{w|$ şiruri de 0 şi 1 ce conţin cel putin un simbol 1 $\}$, este:

(a) $(0|1)^*1$;

(b) $1|(0|1)^*1(1|0)^*$;

(c) 0^*11^*;

(d) $(0|1)^*1(0|1)^*$;

5. Limbajul L notat de expresia regulata $01^*|1$; este:

   (a) $L = \{0, 1, 00, 01, 10, 11, 000, \ldots\}$;

   (b) $L = \{w \in \{0,1\}^* | w$ începe cu 1 $\}$;

   (c) $L = \{w \in \{0,1\}^* | w$ începe cu 0 $\}$;

   (d) $L = \{01^n | n \geq 0\} \bigcup \{1\}$;

6. Limbajul L notat de expresia regulata $(1|0)^*0(0|1)$ este:

   (a) $L = \{0, 1, 00, 01, 10, 11, \ldots\}$;

   (b) $L = \{w \in \{0,1\}^* | w$ se termină cu 00 sau 01$\}$;

   (c) $L = \{w \in \{0,1\}^* | w$ are cel putin un simbol 0 $\}$;

   (d) $L = \{w \in \{0,1\}^* | w$ are 0 în penultima poziţie $\}$;

7. Dacă L_1 este un limbaj regulat iar despre L_2 nu se cunoaşte tipul dar$L_1 \setminus L_2$ este regulat, atunci L_2 poate să fie

   (a) mulţimea vidă

   (b) independent de context

   (c) decidabil

   (d) regulat

8. Aplicând Lema de pompare pentru un limbaj regulat, considerăm un cuvânt w suficient de lung care aparţine limbajului L şi îl descompunem în ..... părţi.

   (a) 2

   (b) 5

   (c) 3

   (d) 6

9. O producţie de forma $A \to B$, unde A şi B nu sunt terminale, se numeşte

   (a) Regulă de ştergere

   (b) Redenumire

   (c) Formă normală Greibach

   (d) Formă normală Chomsky

10. Orice gramatică în formă normală Chomsky este

(a) regulată

(b) dependentă de context

(c) independentă de context

(d) toate cele menţionate anterior

11. Care dintre producţiile de mai jos poate fi acceptată de gramatica în formă normală Chomsky, unde A, B, C şi S sunt neterminale iar a este terminal?

(a) $A \to BC$

(b) $A \to a$

(c) $S \to \lambda$

(d) toate cele menţionate anterior

12. Care dintre următoarele seturi de reguli corespund unei gramatici în formă normală Chomsky?

(a) $A \to AB|BC|CD$, $A \to 0$, $B \to 1$, $C \to 2$, $D \to 3$

(b) $A \to AB$, $S \to BCA|0|1|2|3$

(c) $S \to ABa$, $A \to aab$, $B \to Ac$

(d) toate cele menţionate anterior

13. Fie L limbajul generat de gramatica $G = (V_N, V_T, S, P)$ unde $V_N = \{S, A, B, C, X, Y, Z\}$, $V_T = \{a, b, c\}$, şi

$$\begin{aligned} P = \{\ & S \to \lambda \mid AX \mid BY \mid CZ, \\ & X \to \lambda \mid BY \mid CZ, \\ & Y \to \lambda \mid AX \mid CZ, \\ & Z \to \lambda \mid AX \mid BY, \\ & A \to a, B \to b, C \to c\}. \end{aligned}$$

Alegeţi răspunsurile corecte.

(a) L este limbaj regulat

(b) L nu este limbaj regulat

(c) L este limbaj de programare

14. Fie L limbajul generat de gramatica $G = (V_N, V_T, S, P)$ unde $V_N = \{S, A, B, C, X, Y, Z\}$, $V_T = \{a, b, c\}$, şi

$$\begin{aligned} P = \{\ & S \to \lambda \mid AX \mid BY \mid CZ, \\ & X \to \lambda \mid BY \mid CZ, \\ & Y \to \lambda \mid AX \mid CZ, \\ & Z \to \lambda \mid AX \mid BY, \\ & A \to a, B \to b, C \to c\}. \end{aligned}$$

Care din afirmaţiile următoare este adevărată?

(a) $L = \{w \in V_T^* \mid w$ nu conţine litere consecutive identice$\}$

(b) $L = \{w \in V_T^* \mid w$ conţine litere consecutive identice$\}$

(c) $L = \{w \in V_T^* \mid w$ conţine subşirul $abc\}$

(d) $L = \{\lambda\}$.

15. Fie L limbajul generat de gramatica $G = (V_N, V_T, S, P)$ unde $V_N = \{S, A, B, C, X, Y, Z\}$, $V_T = \{a, b, c\}$, şi

$$\begin{aligned} P = \{\ & S \to \lambda \mid AX \mid BY \mid CZ, \\ & X \to \lambda \mid BY \mid CZ, \\ & Y \to \lambda \mid AX \mid CZ, \\ & Z \to \lambda \mid AX \mid BY, \\ & A \to a, B \to b, C \to c\}. \end{aligned}$$

Fie s_n numărul şirurilor din L cu lungimea n. Pentru orice $n > 1$ are loc relaţia de recurenţă:

(a) $s_n = 2 \cdot s_{n-1}$

(b) $s_n = 3 \cdot s_{n-1}$

(c) $s_n = s_{n-1} + 2 \cdot s_{n-2}$

(d) $s_n = 3\, s_{n-1} + s_{n-2}$

16. Fie L limbajul generat de gramatica $G = (V_N, V_T, S, P)$ unde $V_N = \{S, A, B, C, X, Y, Z\}$, $V_T = \{a, b, c\}$, şi

$$\begin{aligned} P = \{\ & S \to \lambda \mid AX \mid BY \mid CZ, \\ & X \to \lambda \mid BY \mid CZ, \\ & Y \to \lambda \mid AX \mid CZ, \\ & Z \to \lambda \mid AX \mid BY, \\ & A \to a, B \to b, C \to c\}. \end{aligned}$$

Câte şiruri cu lungimea 4 conţine L?

(a) 16

(b) 24

(c) 32

(d) 81

(e) 243

Răspunsuri

Algoritmi şi structuri de date

1. (b), (d)
2. (b)
3. (d), (e)
4. (c)
5. (b), (d), (f)
6. (a)
7. (a)
8. (c),(d)
9. (b)
10. (a)
11. (a)
12. (b)
13. (b)
14. (c)
15. (b),(d)
16. (b),(d)
17. (c)
18. (c)
19. (d)
20. (a),(c)
21. (a), (b)
22. (a), (d)
23. (b)
24. (c)

Algoritmi şi structuri de date

25. (a)
26. (a)
27. (a)
28. (b)

Teoria grafurilor și combinatorică

1. (d)
2. (b)
3. (a)
4. (a)
5. (c)
6. (d)
7. (a)
8. (b)
9. (a)
10. (e)
11. (c)
12. (a)
13. (b)
14. (a)
15. (a), (d), (f)
16. (a)
17. (a)
18. (a)

Logică Computațională

1. (c),(d)
2. (b),(c)
3. (a),(c)
4. (a),(b),(c)
5. (b)
6. (a),(b),(c),(d)
7. (b)
8. (c),(d)
9. (a),(c),(d)
10. (a)
11. (b)
12. (b),(c)
13. (a)
14. (b), (c)
15. (b)

Limbaje formale şi teoria automatelor

1. (a),(d)
2. (a),(b),(c),(d)
3. (d)
4. (b),(d)
5. (d)
6. (b),(d)
7. (a), (b), (d)
8. (c)
9. (b)
10. (b), (c)
11. (a), (b), (c), (d)
12. (a)
13. (a)
14. (a)
15. (a)
16. (b)

Limbajul Python

1. Ce afișează următoarea secvență de cod:

```
y = 9
def test(x = 42, y = 3):
  x = x + y
  y += 1
  print(x, y)
test()
```

   (a) 45 4
   (b) 51 10
   (c) Eroare la execuție deoarece variabila `y` nu este definită
   (d) Eroare la compilare deoarece funcția este apelată fară argumente
   (e) `None`

2. Considerând secvența de cod:

```
s = "Timisoara"
```

   Care dintre secvențele de cod de mai jos returnează șirul de caractere `'aom'` :

   (a) `s[::-3]`
   (b) `s[::2]`
   (c) `s[-1:3:1]`
   (d) `[i for i in s if i in 'aom']`
   (e) `''.join([i for i in s if i in 'aom'])`

3. Care este valoarea expresiei:

```
1 + 2 ** 3 * 4
```

   (a) 33
   (b) 4097
   (c) 36
   (d) 108
   (e) 42

4. Este sigur să folosim operatorul `==` pentru a verifica dacă două variabile de tip `float` sunt egale ?

(a) Nu deoarece reprezentarea internă a valorilor de tip `float` nu este precisă

(b) Da, bineînțeles

(c) Nu, operatorul pentru comparație este `!=`

(d) Da, deoarece reprezentarea în virgulă mobilă asigură precizia necesară

(e) Da, Python asigură aproximările necesare comparării valorilor de tip `float`

5. Care dintre opțiunile de mai jos reprezintă o inițializare corectă a unui dicționar:

(a) d = ('nume': 'valoare')

(b) d = 'nume': 'valoare'

(c) d = {'nume':'valoare'}

(d) d = ['nume'='valoare']

6. Care este rezultatul execuției următoarei secvențe de cod?

```
nume1 = ['Maria', 'Ioana', 'Cristian', 'Mirel']
nume2 = nume1
nume3 = nume1[:]
nume2[0] = 'Vasile'
nume3[1] = 'Andrada'
sum = 0
for ls in (nume1, nume2, nume3):
  if ls[0] == 'Vasile':
    sum += 1
  if ls[1] == 'Andrada':
    sum += 10

print(sum)
```

(a) 11

(b) 12

(c) 21

(d) 22

(e) 33

(f) 0

7. Fie următoarea secvență de cod:

```
class Persoana(object):
  def __init__(self, nume):
    self.nume = nume
```

```
  def salut(self):
    print ("Salut")
class Profesor(Persoana):
  def __init__(self, nume, titlu):
     Persoana.__init__(self, nume)
     self.titlu = titlu
  def salut(self):
    print ("Buna ziua")
prof1 = Profesor("Decebal Popescu", "Asistent")
```

Cum putem apela metoda `salut()` din clasa `Persoana`?

(a) `Persoana.salut(prof1)`

(b) `prof1.salut()`

(c) `super(prof1).salut()`

(d) `Persoana(prof1).salut()`

(e) Nu putem apela metoda `salut()` din clasa `Persoana` deoarece este supraîncărcată în clasa `Profesor`

8. Care este rezultatul execuției următoarei secvențe de cod?

```
class A:
  def __init__(self):
     self.calcI(30)
  def calcI(self, i):
     self.i = 2 * i;

class B(A):
  def __init__(self):
     super().__init__()
     print("i from B is", self.i)
  def calcI(self, i):
     self.i = 3 * i;

b = B()
```

(a) Doar metoda `__init__` din clasa `B` este invocată

(b) Metoda `__init__` din clasa `A` este invocată și se afișează ‘‘`i from B is 0`’’.

(c) Metoda `__init__` din clasa `A` este invocată și se afișează ‘‘`i from B is 60`’’.

(d) Metoda `__init__` din clasa `A` este invocată și se afișează ‘‘`i from B is 90`’’.

9. Când este execută clauza `finally` dintr-un bloc `try: except`:

(a) când nu are loc nici o excepție
(b) când are loc o excepție
(c) tot timpul
(d) doar dacă o anumită condiție specificată este îndeplinită
(e) niciodată

10. Care dintre variantele de mai jos deschide un fișier pentru scriere?

(a) output_file = open("hello.txt", "r")
(b) output_file = open("hello.txt", "w")
(c) output_file = openFile("hello.txt", "w")
(d) output_file = File("hello.txt", "w")
(e) output_file = open("hello.txt", encoding="utf8")

Limbajul C

1. Fie următoarea secvență de cod:

```
#define swap(a,b) {int aux; aux=a; a=b; b=aux;}
float x=10.5, y=3.75;
```

în urma apelului $swap(x, y)$; valorile variabilelor x, respectiv y vor fi:

(a) x=3.75, y=10.5; (b) x=3.0, y=10.5; (c) x=3.75, y=10.0; (d) x=3.0, y=10.0;

2. Ce reprezintă domeniul de vizibilitate a unei variabile?

(a) plaja de valori pe care le poate lua;
(b) locul unde se creează;
(c) locul din textul sursă unde poate fi folosită;
(d) dacă are semn sau nu;

3. La execuţia programului următor se tastează 20. Ce se va afişa după execuţie?

```
#include <stdio.h>
void main() {
    char a;
    scanf("%c",&a);
    printf("%c",a); }
```

(a) 20; (b) 2; (c) 0;

4. Care este rezultatul numeric al evaluării expresiei:

   a < b < c

   dacă a=-2, b=-1 si c=0?

   (a) 1; (b) TRUE; (c) 0; (d) FALSE;

5. Ce se va afişa în urma execuţiei secvenţei următoare:

```
#include <stdio.h>
void main() {
    unsigned char x=25;
    x=x<<2;
    printf("%d",(int)x); }
```

   (a) 27; (b) 100; (c) 23;

6. Ce afişează programul?

```
#include <stdio.h>
void main() {
    int x,y,z;
    x=y=z=4;
    printf("%d",(x<<z)-(x|y)+(z&y));}
```

   (a) 64 (b) 4 (c) 0 (d) 5 (e) 33 (f) 32

7. Care din următoarele variante reprezintă modalităţi de comunicare între funcţii?

   (a) apel;
   (b) prin variabile locale;
   (c) prin argumentele actuale;
   (d) prin valoarea returnată;
   (e) prin variabile globale
   (f) prin includere

8. Ce este contextul de apel al unei funcţii?

   (a) lista argumentelor formale

(b) o zona de memorie (de pe stivă)

(c) locul din textul sursă în care se apelează

(d) instrucţiunile (definiţia funcţiei)

9. Care este valoarea variabilei n după execuţia secvenţei:

```
char t[ ]="timisoara", *p,*q,n;
p=q=t;
while(*(q++));
n=q-p;
```

(a) n=0

(b) n=9

(c) n=10

(d) n='\0'-'t'

10. Ce reprezintă declaraţia: int *(*f)(int *) ?

(i) funcţie ce primeşte argument pointer la întreg şi întoarce pointer la întreg

(ii) o declaraţie greşită

(iii) pointer către o funcţie care aşteaptă ca argument un pointer la int şi întoarce un pointer la int

(iv) pointer către o funcţie care întoarce un int

(a) i (c) iii (e) nici una (g) i şi iv

(b) ii (d) iv (f) toate

11. Ce se afişează în urma execuției următoarei secvențe de cod?

```
#include <stdio.h>
void main() {
    char *u[2]={"abc","def"}, **v;
    v=&u[0];
    printf("%c",(*v)[1]);
}
```

(a) a (c) c (e) e (g) abc

(b) b (d) d (f) f

Limbajul C++

1. Un constructor se caracterizează prin următoarele proprietăţi în limbajele C++/Java:
   (a) Este o funcţie membră care are acelaşi nume ca şi clasa în care este declarată
   (b) Este o funcţie membră care întoarce o valoare
   (c) Este o funcţie membră care nu are valoare de return
   (d) Este o funcţie membră utilizată pentru a iniţializa un obiect
   (e) Este o funcţie membră utilizată pentru a dealoca spaţiu de memorie
   (f) O clasă nu poate avea mai mult de un constructor
2. Care dintre următoarele afirmaţii sunt adevărate despre template-uri(C++)/generice(Java)?
   (a) Template-urile / genericele sunt o facilitate a limbajului care permit folosirea aceluiaşi cod pentru diferite tipuri de date
   (b) Template-urile / genericele sunt exemple de polimorfism.
   (c) Template-urile / genericele nu pot fi folosite cu tipuri de date abstracte definite de utilizator.
   (d) În afară de template-urile / genericele implicite nu pot fi definite altele noi.
3. Care dintre următoarele afirmaţii sunt adevărate în limbajele C++/Java?
   (a) O funcţie statică nu poate arunca o excepţie.
   (b) O funcţie statică nu poate accesa o variabilă membru non-statică a clasei.
   (c) O funcţie statică nu poate accesa o variabilă membru statică a clasei.
   (d) O variabilă membru statică a clasei nu poate fi modificată într-o funcţie membră non-statică.
4. Excepţiile sunt:
   (a) Erori care apar la compilarea programului
   (b) Situaţii speciale tratate în program prin teste de tipul if(variabila == NULL)
   (c) Erori care apar la rularea programului
   (d) 'Aruncate' folosind instrucţiunea try şi 'tratate' folosind instrucţiunea catch
   (e) 'Aruncate' folosind instrucţiunea throw(s) şi 'tratate' în blocuri try - catch (finally)
5. Selectați afirmația corectă pentru instrucțiunea cout de mai jos

```
void f(list<string>& lst) {
    list<string>::const_iterator x = find(lst.begin(), lst.end(), "Timisoara");

    if (x==lst.end()) std::cout << "_____________________";
}
```

   (a) A gasit Timisoara

(b) Nu a gasit Timisoara
(c) Timisoara este ultimul element al listei
(d) Timisoara este primul element al listei

6. Principiul OCP (Open Close Principle) din cadrul principiilor SOLID se referă la:
   (a) Responsabilităţile pe care trebuie să le implementeze o clasă
   (b) Realizarea de ierarhii de clase consistente
   (c) Problemele care apar din cauza codului duplicat
   (d) Posibilitatea extinderii claselor şi evitarea modificărilor claselor şi codului existent
7. Care din următoarele afirmaţii sunt false în contextul supraîncărcării operatorilor în limbajul C++:
   (a) Se pot adăuga noi operatori limbajului
   (b) Se poate schimba n-aritatea (numărul de operanzi ai operatorului)
   (c) Nu toţi operatorii pot fi supraîncărcaţi
   (d) Operatorii se pot supraîncărca prin intermediul funcţiilor membre clasei şi funcţii prietene clasei.
8. În limbajul C++, dacă o clasă X are ca membri variabile pointer, este recomandat să conţină:
   (a) Un destructor care nu face nimic $X(){}$
   (b) O funcţie friend care să permită copierea obiectului
   (c) Supraîncărcarea operatorului =
   (d) Constructorul de copiere
   (e) Supraîncărcarea operatorului ==
   (f) Un destructor în care se dealocă memoria adresată de membrii de tip pointer ai clasei
9. Se dă următoarea secvență de cod:

```
class Date {
public:
Date(int day=0, int month=0, int year=0);
Date(const Date& ref);
};

void g(Date d) { }
void f() {
  Date d{7 , 3, 2007};
  Date d1;
  Date d2 = d;
  d2 = d;
  g(d);
}
```

De câte ori sunt apelați constructorii clasei *Date*?

(a) 2
(b) 3
(c) 4
(d) 5

10. Care din următoarele afirmații sunt adevărate?

```
class Shape { virtual void draw() = 0; };

struct Rectangle : Shape {
void addControlPoint(int x, int y);
};

int main() { Rectangle c; return 0; }
```

(a) clasa *Rectangle* nu poate fi instanțiată deoarece metoda *addControlPoint()* nu este implementată

(b) clasa *Rectangle* nu poate fi instanțiată deoarece este clasă abstractă

(c) codul se compilează cu succes

Limbajul Java

1. Dacă o metodă are ca parametru o referinţă la un obiect inmutabil, de exemplu clasa String, poate metoda să modifice starea obiectului?

   (a) Da, dacă avem o referinţă la un obiect imutabil îi putem modifica starea
   (b) Da, dar trebuie să folosim un operator special "+" ca să realizăm acest lucru
   (c) Nu, starea obiectelor imutabile nu poate fi modificată de nimeni după ce obiectul a fost creat
   (d) Nu, doar proprietarul obiectului imutabil poate să îi schimbe starea.

2. Care dintre afirmaţiile de mai jos este adevărată în relaţie cu limbajul Java?

   (a) Specificatorul `final` se aplică la clase, atribute si metode;
   (b) O metodă declarată `final` poate apela doar metode declarate final în aceeaşi clasă;
   (c) O metodă declarată `final` nu poate fi suprascrisă la subclasare;
   (d) Un atribut declarat `final` poate fi setat doar o singură dată;

3. Care dintre afirmaţiile de mai jos este/sunt adevărată/e despre conceptul de moştenire între clase?

   (a) Mecanismul prin care o clasă preia structura (datele membru) şi comportamentul (metodele) unei alte clase la care adaugă elemente specifice;
   (b) Prin mecanismul de moştenire avem acces la datele private ale clasei(claselor) de bază;
   (c) Nu toate limbajele de programare implementează conceptul de moştenire multiplă;
   (d) O metodă definită în clasa de bază nu poate fi suprascrisă în clasa derivată;
   (e) Moştenirea poate fi întotdeauna înlocuită cu compunerea obiectelor.

4. Care dintre afirmaţiile de mai jos este/sunt adevărată/e referitor la conceptul de clasă abstractă în limbajul Java:

   (a) O clasă care nu poate fi instanţiată;

(b) Nu există nicio diferenţă între clasele abstracte şi interfeţe;

(c) O clasă abstractă poate conţine variabile membre care nu sunt constante.

5. Care dintre afirmaţiile de mai jos sunt adevărate despre clase şi obiecte:

   (a) O clasă modelează caracteristici comune mai multor obiecte;

   (b) Un obiect este o realizare/instanță a unei clase;

   (c) O clasă este o realizare/instanță a unui obiect;

   (d) Nu există nicio legătură între clase şi obiecte.

6. Care din următoarele afirmaţii sunt false relativ la limbajul Java:

   (a) Orice clasă este derivată din clasa Object

   (b) O clasă poate extinde una sau mai multe clase de bază

   (c) O clasă poate implementa una sau mai multe interfeţe

   (d) Subclasele moştenesc atributele, metodele şi constructorii clasei de bază

7. Care din următoarele afirmații este/sunt adevărare despre clasa PreparedStatement?

   (a) interogarea este compilată de fiecare dată când este executată

   (b) permite rularea aceleiași interogări, prin compilarea ei o singură dată și parametrizarea cu valori diferite

   (c) folosirea clasei PraparedStatement evită întotdeauna SQL Injection

   (d) SQL Injection poate fi evitat prin folosirea placeholderului "?"

8. Două funcţii care au acelaşi nume într-o clasă Java, pot fi diferenţiate datorită:

   (a) Numărului de variabile din lista de parametri

   (b) Nu pot fi diferenţiate

   (c) Tipului variabilelor din lista de variabile

   (d) Tipului de return al funcției

9. Se consideră următoarea secvență de cod:

```
class A implements Runnable {
  int counter = 0;
  public void run() {
    while (true) counter ++;
  }
}
public class Test{
  public static void main(String [] arg) {
```

```
    A a = new A();
  }
}
```

Care din variantele de mai jos instantiază şi lansează în execuție un fir de execuţie?

(a) (new Thread()).start()

(b) a.start( );

(c) a.run( );

(d) new Thread.run( );

10. Considerand următoarea secvență de cod, stabiliţi care este ordinea execuției (in ipoteza că o clasă are un constructor, o metodă, un bloc static și un bloc de instanță):

```
public class A {
  public A() {
    System.out.print("constructor; ");
  }
  public void method() {
    System.out.print("method; ");
  }
  {
    System.out.print("instance block; ");
  }
  static {
    System.out.print("static block; ");
  }
  public static void main(String[] args) {
    A a = new A();
    a.method();
  }
}
```

(a) instance block; method; static block; constructor;

(b) static block; instance block; constructor; method;

(c) method; constructor; instance block; static block;

(d) static block; method; instance block; constructor;

(e) constructor; static block; instance block; method;

11. Se consideră următoarea secvență de cod:

```
public class Main {
  public static void main(String[] args) {
    Set<String> fructe = new TreeSet<>(Arrays.asList("mere", "pere",
                         "banane", "mere", "kiwi", "ananas", "portocale"));
    System.out.println(fructe);
  }
}
```

Care din variantele de mai jos se va afișa:

(a) $[mere, pere, banane, kiwi, ananas, portocale]$

(b) $[ananas, banane, kiwi, mere, pere, portocale]$

(c) $[Ljava.lang.String; @15db9742]$

(d) $[portocale, ananas, kiwi, mere, banane, pere]$

Proiectarea aplicaţiilor software

1. Verificarea software-lui poate implica
   (a) analiza statică automată
   (b) evaluarea utilității și utilizabilității software-lui în situații operaționale.
   (c) depanarea erorilor
   (d) inspectări ale software-lui
   (e) testarea în vederea descoperirii existenței erorilor
   (f) testarea faptului că software-ul îndeplinește cerințele utilizator
2. Diagrama de stări și tranziții reprezintă
   (a) funcțiile sistemului
   (b) răspunsul sistemului la evenimente interne
   (c) răspunsul sistemului la evenimente externe
   (d) interacțiuni între obiecte din sistem
   (e) structura datelor
   (f) interacțiunile actorilor cu sistemul
   (g) fluxul de prelucrare a datelor în sistem
3. Metodele agile de dezvoltare de software implică
   (a) Furnizare incrementală
   (b) Implicarea clientului pe parcursul procesului de dezvoltare

(c) Instituirea de procese normative pentru lucrul în echipă

(d) Acțiuni periodice de eliminare a complexității din sistem

(e) Modelarea completă a software-lui înainte de scrierea codului

4. Bifați situațiile in care se reutilizează doar concepte:

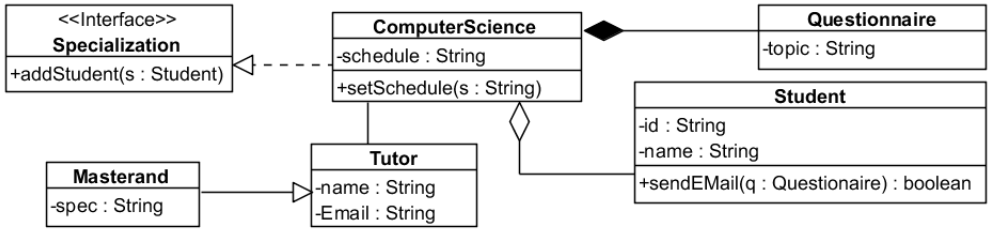
   (a) Servicii software

   (b) Șabloane de proiectare (design patterns)

   (c) Biblioteci de programe

   (d) Șabloane arhitecturale

5. Fie următoarea diagramă de clase.



   Bifați afirmațiile adevărate.

   (a) Un obiect de tip `ComputerScience` conține o colecție de obiecte de tip `Student`

   (b) Clasa `Questionnaire` are un atribut public de tip `String`

   (c) Clasa `ComputerScience` are operația publică `addStudent(s:String)`

   (d) Clasa `ComputerScience` are operația privată `setSchedule(s:String)`

   (e) Clasa `Tutor` este superclasă pentru clasa `Masterand`

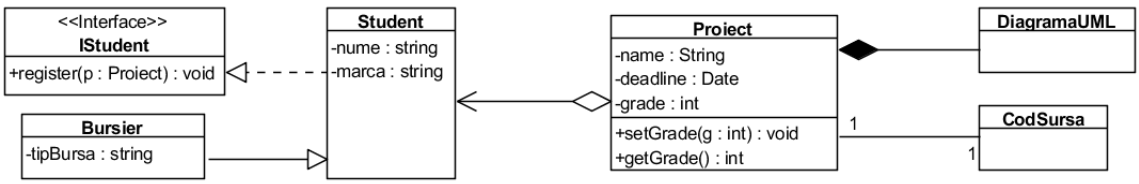
   (f) Clasa `ComputerScience` definește o compoziție de obiecte de tip `Questionnaire`.

   (g) Între clasa `ComputerScience` și clasa `Tutor` există o asociere unidirecțională.

   (h) Clasa `ComputerScience` moștenește interfața `Specialization`.

   (i) Clasa `Tutor` definește un agregat de obiecte de tip `Masterand`.

6. Fie următoarea diagramă de clase.



   Care secvență de cod Java descrie corect și complet relațiile clasei `Proiect` ?

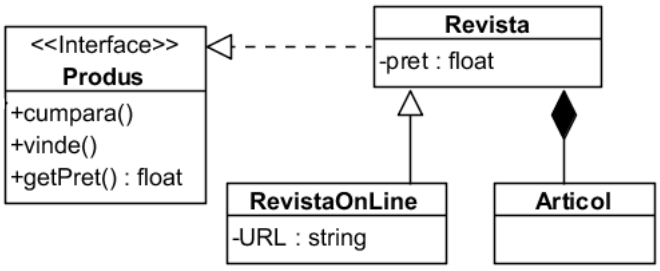
(a)
```
class Proiect extends Student {
        private Collection <DiagramaUML> diagramele = new ArrayList<>();
        private CodSursa codul;
        ...}
```

(b)
```
class Proiect {
        private Collection <Student> studenti;
        private Collection <DiagramaUML> diagramele = new ArrayList<>();
        private CodSursa codul;
        ...}
```

(c)
```
class Proiect {
        private Student student;
        private DiagramaUML diagrama;
        private CodSursa codul;
        ...}
```

(d)
```
class Proiect {
        private Collection <Student> studenti = new ArrayList<>();
        private Collecction <DiagramaUML> diagramele;
        private CodSursa codul;
        ...}
```

7. Fie următoarea diagramă de clase.



Care secvență de cod Java descrie corect și complet relațiile clasei `Revista` ?

(a)
```
class Revista extends RevistaOnLine implements Produs {
        private Collection <Articol> articole;
        ...}
```

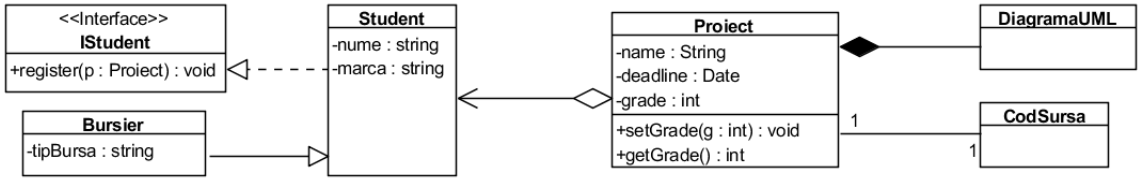
(b)
```
class Revista implements Produs {
        private Collection <Articol> articole = new LinkedList<>();
        private RevistaOnLine revista;
```

```
        ...}
```

(c)
```
class Revista implements Produs {
        private Collection <Articol> articole = new LinkedList<>();
        ...}
```

(d)
```
class Revista extends RevistaOnLine {
        private Collection <Articol> articole = new LinkedList<>();
        private Produs produs;
        ...}
```

8. Fie următoarea diagramă de clase.



Selectați descrierea corectă și completă a relațiilor reprezentate în diagramă:

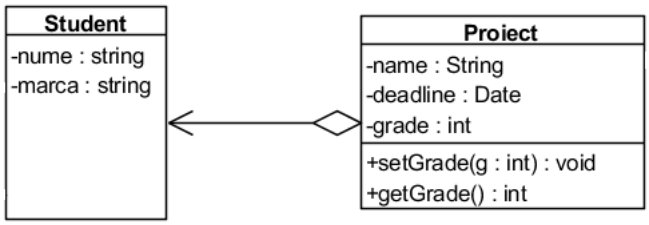
(a) Asociere bidirecțională, între clasele `Proiect` și `CodSursa`; compoziție între clasele `Proiect` (compozit) și `DiagramaUML` (componenta); agregare între clasele `Proiect` (agregat) și `Student` (componenta); generalizare între interfața `IStudent`(implementată) și clasa `Student` (implementează); realizare între clasa `Student`(superclasa) și clasa `Bursier` (subclasa)

(b) Asociere bidirecțională, între clasele `Proiect` și `CodSursa`; compoziție între clasele `Proiect` (compozit) și `DiagramaUML` (componenta); agregare între clasele `Proiect` (agregat) și `Student` (componenta); realizare între interfața `IStudent` (implementată) și clasa `Student` (implementează); generalizare între clasa `Student`(superclasa) și clasa `Bursier`(subclasa)

(c) Asociere bidirecțională, între clasele `Proiect` și `CodSursa`; agregare între clasele `Proiect` (agregat) și `DiagramaUML`(componenta); compoziție între clasele `Proiect` (agregat) și `Student` (componenta); realizare între interfața `IStudent` (implementată) și clasa `Student` (implementează); generalizare între clasa `Student` (superclasa) și clasa `Bursier` (subclasa)

(d) Asociere bidirecțională, între clasele `Proiect` și `CodSursa`; compoziție între clasele `DiagramaUML` (compozit) și `Proiect`(componenta); agregare între clasele `Student` (agregat) și `Proiect` (componenta); realizare între interfața `IStudent`(implementată) și clasa `Student` (implementează); generalizare între clasa `Student` (superclasa) și clasa `Bursier` (subclasa)

9. Fie următoarea diagramă de clase.



Care secvență de cod Java descrie corect și complet relația clasei `Proiect` cu clasa `Student`?

(a)
```
class Student extends Proiect{...}

class Proiect{...}
```

(b)
```
class Proiect extends Student{...}

class Student{...}
```

(c)
```
class Proiect {
    private Collection <Student> studenti ...}

class Student{...}
```

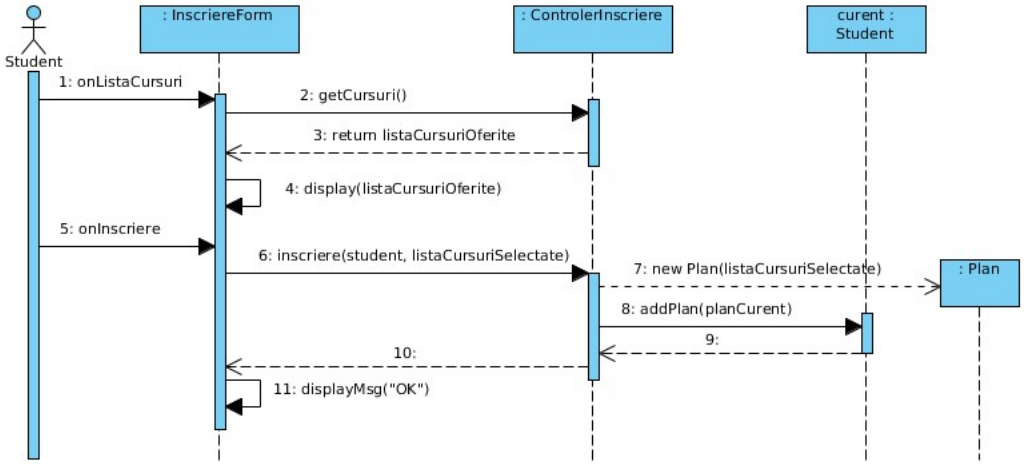
(d)
```
class Proiect {
    private Collection <Student> studenti ...}

class Student {
    private Proiect proiect;
     ...}
```

(e)
```
class Proiect {
    private Collection <Student> studenti ...}

class Student {
    private Collection<Proiect> proiecte;
     ...}
```

10. Fie următoarea diagramă de secvențe.



Ce operații ale clasei `ControlerInscriere` rezultă din aceasta?

(a) `getCursuri()`

(b) `display(listaCursuriOferite)`

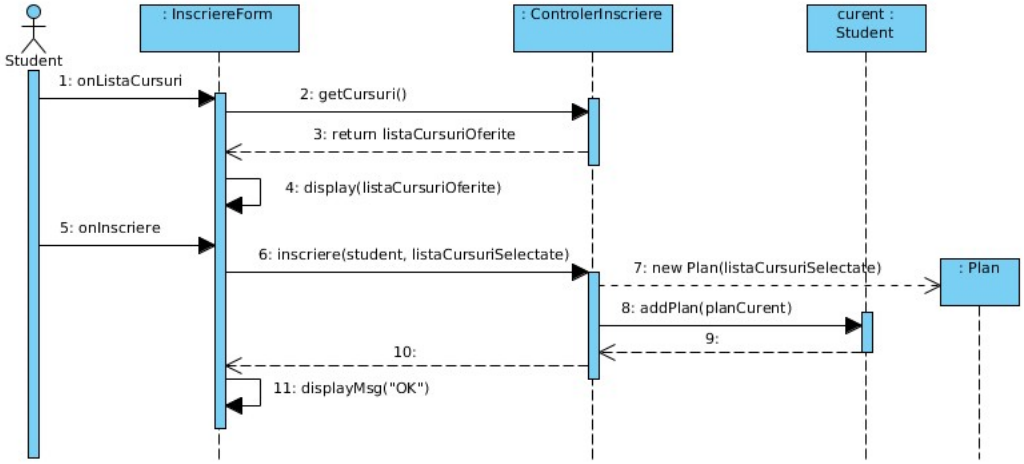
(c) `inscriere(student, listaCursuriSelectate)`

(d) `Plan(listaCursuriSelectate)`

(e) `addPlan(planCurent)`

(f) `displayMsg(''OK'')`

11. Fie următoarea diagramă de secvențe.



Selectați clasele din care sunt instanțiate obiectele implicate în interacțiune:

(a) InscriereForm

(b) ControlerInscriere

(c) listaCursuriSelectate

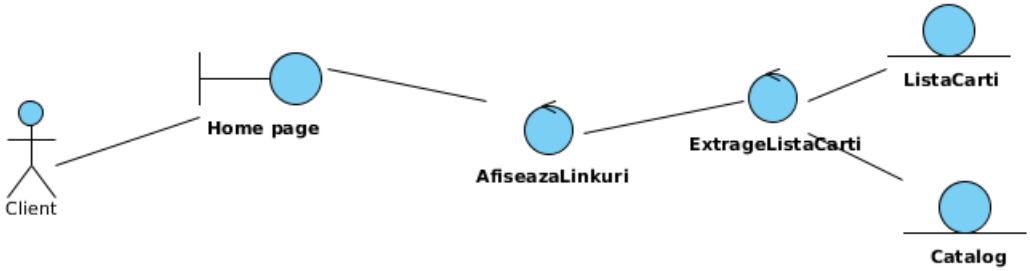
(d) curent

(e) Student

(f) Plan

(g) planCurent

12. Fie următoarea diagramă de robustețe.



Care afirmații sunt adevărate?

(a) `Home page` este obiect *boundary*.

(b) `ExtrageListaCarti` poate fi obiect persistent.

(c) `AfiseazaLinkuri` este obiect de interacțiune cu actor.

(d) `ExtrageListaCarti` ar putea fi implementat ca metodă a unei clase *entity*.

(e) `ListaCarti` este obiect de interacțiune cu actor.

(f) `Catalog` este obiect *entity*.

13. Selectați perechea de termeni cu care se înlocuiesc spațiile libere din următoarea frază:

"În jocul Essence 'Checkpoint construction', un checkpoint este un set de ..... ce trebuie îndeplinite la un anume moment în timp, în cadrul unui efort de dezvoltare de software, ȘI este definit în termeni de ....."

(a) criterii / activități

(b) componente / roluri

(c) alphas / activity spaces

(d) criterii / alpha states

(e) alpha states / criterii

14. Realizaţi corespondenţa corectă între concept şi definiţia sa:

(a) Caz de testare	[1] re-aplicarea unui set de teste existent.
(b) Testarea regresiilor	[2] procesul de testare a componentelor individuale în regim de izolare.
(c) Testare la suprasolicitare	[3] serie de teste în care încărcarea este crescută treptat.
(d) Testarea performanţei	[4] utilizarea sistemului dincolo de încărcarea maximă pentru care a fost proiectat.
(e) Testare unitate	[5] specificaţiile intrărilor testului şi a rezultatelor aşteptate de la sistem, plus precizarea entităţii testate.

(a) a – 5, b – 1, c – 4, d – 3, e – 2

(b) a – 5, b – 3, c – 4, d – 1, e – 2

(c) a – 2, b – 1, c – 3, d – 4, e – 5

(d) a – 1, b – 2, c – 3, d – 4, e – 5

(e) a – 3, b – 2, c – 4, d – 5, e – 1

15. Selectați metodele de evitare a introducerii de erori în dezvoltarea de programe:

    (a) Fiecare clasă trebuie să aibă o singură reponsabilitate

    (b) Folosire expresii regulate pentru validarea datelor de intrare

    (c) Evitarea încuibării multiple de instrucțiuni condiționale

    (d) Jurnalizarea interacțiunii utilizatorilor cu programul

    (e) Înlocuirea corpului unei metode cu un nou algoritm mai clar care returnează acelaşi rezultat

    (f) Minimizarea adâncimii ierahiilor de clase

    (g) Salvare automată a datelor utilizatorului la intervale presetate

    (h) Identificarea aspectelor care variază ale unei aplicații și separarea lor de cele care nu variază

    (i) Verificare valori date de intrare față de domeniul definit cu reguli de intrare

    (j) Folosire aserțiuni pentru verificare rezultat de la un serviciu extern

Baze de date

1. Se consideră tabela R(A) conţinând înregistrările {(1), (2)} şi tranzacţiile

```
(T1) UPDATE R SET A = A*2
(T2) SELECT AVG(A) FROM R; SELECT MAX(A) FROM R
```

   Dacă tranzacţia T2 se execută folosind nivelul de izolare *repeatable read*, care dintre următoarele afirmaţii este adevărată

   (a) Dacă AVG(A) = 1.5 atunci MAX(A) poate să ia valorile 2 sau 4

   (b) AVG(A) va fi întodeauna 1.5 iar MAX(A) = 2

(c) AVG(A) poate să ia valorile 1.5 sau 3, iar MAX(A) valorile 2 sau 4

(d) Dacă MAX(A) = 4 atunci AVG(A) este 1.5

2. Trei dintre expresiile relaţionale de mai jos returnează numele studenţilor care nu au aplicat la specializările CS sau PH. Care returnează altceva? (Observaţie: numele studenţilor pot fi duplicate.)

(a) $\pi_{sName}(Student \bowtie (\pi_{sID}(Student) - (\pi_{sID}(\sigma_{major='CS'}Apply) \cup \pi_{sID}(\sigma_{major='PH'}Apply))))$

(b) $\pi_{sName}(Student \bowtie (\pi_{sID}Student - \pi_{sID}(\sigma_{major='CS' \vee major='PH'}Apply)))$

(c) $\pi_{sName}(\pi_{sID,sName}Student - \pi_{sID,sName}(Student \bowtie \pi_{sID}(\sigma_{major='CS' \vee major='PH'}Apply)))$

(d) $\pi_{sName}Student - \pi_{sName}(Student \bowtie \pi_{sID}(\sigma_{major='CS' \vee major='PH'}Apply))$

3. Tabelul de mai jos conţine un eşantion de date reprezentativ pentru toate dependenţele funcţionale dintr-un set de date real. Care sunt dependenţe funcţionale ce pot fi extrase din aceste date?

A	B	C	D	E	F
100	Timişoara, str. Revolutiei	TM	1234	PA	Descriere A
100	Timişoara, str. Revolutiei	TM	123	PB	Descriere B
100	Timişoara, str. Revolutiei	TM	10	A	Descriere A
101	Timişoara, str. Revolutiei	TM	10	A	Descriere A
151	Deta, str. Principală	TM	1234	PA	Descriere A
200	Arad, str. Republicii	AR	1234	PB	Descriere B

(a) A $\rightarrow$ {B, C}, E $\rightarrow$ {F}

(b) D $\rightarrow$ {A, B, C, E, F}, F $\rightarrow$ {E}

(c) {A, E} $\rightarrow$ {D}

(d) {D, E} $\rightarrow$ {A, B}

4. Considerând relaţia R(A, B, C, D, E, F) unde A - F sunt coloanele din tabelul de mai sus şi dependenţele funcţionale tot cele de mai sus, care dintre următoarele seturi de atribute este cheie candidată?

(a) {B, F}

(b) {A, E}

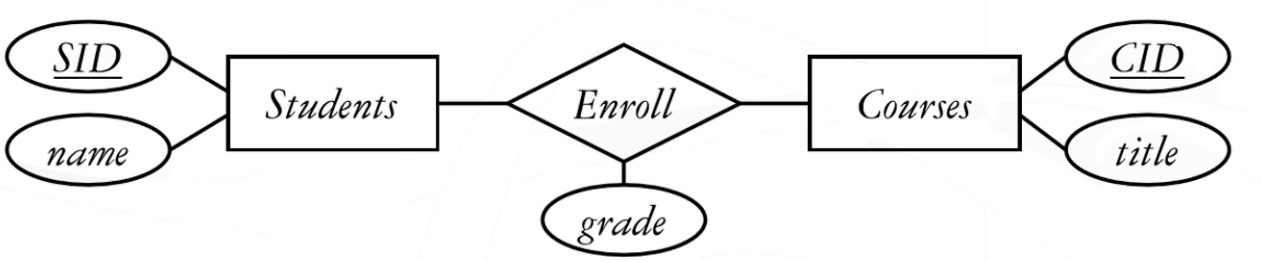
(c) {A, E, D}

(d) {A}

5. Relativ la relaţia R de mai sus, care dintre următoarele afirmaţii sunt false?

(a) Relaţia se află în prima formă normală
(b) Relaţia se află în a doua formă normală
(c) La o adresă (coloana B) nu pot corespunde valori diferite în coloana A
(d) Nu pot exista două înregistrări în tabela R cu valori diferite în coloana D dacă valorile din coloana A sunt identice

6. Fie următoarea diagramă E/R.



Care dintre următoarele relații fac parte din reprezentarea acestei diagrame în modelul relaţional?

(a) Courses(CID, title)
(b) Students(SID, name)
(c) Enroll(SID,CID)
(d) Enroll(SID, name, CID, title, grade)
(e) Students(SID, name, CID)
(f) Enroll(SID, CID, grade)

7. Presupunem că în tabela EMPLOYEES sunt 10 angajați care au salarii de 1000 euro fiecare, cu excepția unuia care are valoarea NULL și că primii 5 angajați lucrează în departamentul 1 și ceilalți 5 angajați în departamentul 2. Câte linii va afișa interogarea următoare:

```
SELECT department_id, SUM(salary)
FROM employees
GROUP BY department_id
HAVING SUM(NVL(salary,1000)) > 4000;
```

(a) Nicio linie
(b) O linie
(c) Două linii
(d) Un alt răspuns

8. Care dintre interogările de mai jos selectează identificatorul (ID) și numele departamentelor care nu au angajați? Coloana DEPARTMENT_ID este comună tabelelor EMPLOYEES și DEPARTMENTS.

(a)
```
SELECT d.department_id, d.department_name
FROM departments d
WHERE NOT EXISTS (SELECT 1 FROM employees e WHERE e.department_id=d.department_id);
```

(b)
```
SELECT d.department_id, d.department_name
FROM employees e LEFT OUTER JOIN departments d
ON (d.department_id=e.department_id) WHERE e.department_id IS NULL;
```

(c)
```
SELECT d.department_id, d.department_name
FROM employees e RIGHT OUTER JOIN departments d
ON (d.department_id=e.department_id) WHERE e.department_id IS NULL;
```

(d)
```
SELECT department_id, department_name
FROM departments
MINUS
SELECT d.department_id, d.department_name
FROM employees e JOIN departments d ON e.department_id=d.department_id;
```

9. Dacă se dorește afișarea tuturor locațiilor care nu au niciun departament, precum și a departamentelor care nu au fost asignate niciunei locații, ce tip de JOIN între tabelele DEPARTMENTS și LOCATIONS va afișa aceste informații ca parte din liniile returnate?

   (a) NATURAL JOIN
   (b) FULL OUTER JOIN
   (c) LEFT OUTER JOIN
   (d) RIGHT OUTER JOIN

10. Care afirmații referitoare la indecși sunt corecte?

    (a) Când un tabel este șters, indecșii corespunzători acestuia sunt șterși automat.
    (b) Pentru fiecare operație DML realizată, indecșii corespunzători sunt actualizați automat.
    (c) Indecșii se recomandă a fi creați pentru coloane care sunt în mod frecvent referite ca parte a unei expresii.
    (d) Pentru fiecare constrângere de cheie primară sau cheie unică (PRIMARY KEY sau UNIQUE) dintr-un tabel se creează automat un index unic.

11. Prin următoarea instrucțiune SQL se dorește a regăsi liniile din tabelul ORDER_ITEMS aferente produselor identificate prin PRODUCT_ID care au un UNIT_PRICE mai mare decât 1000 și care au fost comandate de mai mult de 5 ori.

```
SELECT product_id, COUNT(order_id) total, unit_price
FROM order_items
WHERE unit_price>1000 AND COUNT(order_id)>5
GROUP BY product_id, unit_price;
```

Care afirmații sunt adevărate cu privire la instrucțiunea SQL de mai sus:

(a) Instrucțiunea se execută și furnizează rezultatul așteptat.

(b) Instrucțiunea nu se execută întrucât funcția de agregare este utilizată în clauza WHERE.

(c) Instrucțiunea ar trebui să utilizeze operatorul logic OR în loc de AND.

(d) Instrucțiunea nu se execută deoarece în clauza SELECT coloana UNIT_PRICE este adăugată după coloana pentru care este utilizată funcția de agregare.

12. Se dorește afișarea coloanelor FIRST_NAME și SALARY aferente angajaților din tabelul EMPLOYEES care au același șef (MANAGER_ID) cu al angajatului identificat prin EMPLOYEE_ID = 101 și care au un salariu mai mare sau egal cu al aceluiași angajat identificat prin EMPLOYEE_ID = 101. Care dintre interogările următoare vor afișa rezultatul dorit?

(a)
```
SELECT first_name, salary
FROM employees
WHERE (manager_id, salary) >= ALL (SELECT manager_id, salary
                                   FROM employees WHERE employee_id = 101)
AND employee_id <> 101;
```

(b)
```
SELECT first_name, salary
FROM employees
WHERE (manager_id, salary) >= ( SELECT manager_id, salary
                                FROM employees
                                WHERE employee_id = 101)
AND employee_id <> 101;
```

(c)
```
SELECT first_name, salary
FROM employees
WHERE (manager_id, salary) >= ANY (SELECT manager_id, salary
                                   FROM employees
                                   WHERE employee_id = 101 AND employee_id <> 101);
```

(d)
```
SELECT first_name, salary, manager_id
FROM employees
WHERE manager_id = (SELECT manager_id
                    FROM employees
                    WHERE employee_id = 101)
AND salary >= (SELECT salary
```

```
                FROM employees
                WHERE employee_id = 101)
    AND employee_id <> 101;
```

(e)
```
SELECT e.first_name, e.salary, e.manager_id
FROM employees e, employees m
WHERE m.employee_id = 101 AND e.manager_id = m.manager_id
AND e.salary >= m.salary AND e.employee_id <> m.employee_id;
```

13. Evaluați următoarea instrucțiune CREATE TABLE:

```
CREATE TABLE products
(product_id NUMBER(6) CONSTRAINT prod_id_pk PRIMARY KEY,
 product_name VARCHAR2(15));
```

Care afirmații sunt adevărate cu privire la constrângerea PROD_ID_PK?

(a) Ar fi creată numai dacă în prealabil este creat manual un index unic.

(b) Ar fi creată și ar folosi un index unic creat automat.

(c) Ar fi creată și ar folosi un index non-unic creat automat.

(d) Ar fi creată și ar rămâne în stare dezactivată deoarece nu e specificat explicit niciun index în comandă.

14. Fiind dată vederea V de mai jos, în ce condiții comanda SQL de mai jos va genera o excepție cauzată de constrângerea WITH CHECK OPTION?

```
UPDATE V SET Curs = 'Baze de date II' WHERE ID=101;

CREATE VIEW V(ID, Curs, EDate) AS
SELECT StudID, CourseTitle, EnrollmentDate
FROM Enrollments
WHERE CourseTitle='Algebra' AND Accepted=1
WITH CHECK OPTION;
```

(a) Întotdeauna datorită WITH CHECK OPTION

(b) Dacă studentul cu ID-ul 101 a fost acceptat la cursul de Algebra

(c) Dacă studentul cu ID-ul 101 a fost acceptat la cursul de Baze de date II

(d) Dacă studentul cu ID-ul 101 a aplicat la cursul de Algebră (indiferent de starea câmpului Accepted)

Răspunsuri

Limbajul Python

1. (a)
2. (a)
3. (a)
4. (a)
5. (c)
6. (b)
7. (a), (d)
8. (d)
9. (c)
10. (b)

Limbajul C

1. (c)
2. (c)
3. (b)
4. (c)
5. (b)
6. (a)
7. (c), (d), (e)
8. (b)
9. (c)
10. (c)
11. (b)

Limbajul C++

1. (a), (c), (d)
2. (a), (b)
3. (b)
4. (c), (e)
5. (b)
6. (d)
7. (a), (b)
8. (c), (d), (f)
9. (c)
10. (b)

Limbajul Java

1. (c)
2. (a), (c), (d)
3. (a), (c)
4. (a), (c)
5. (a), (b)
6. (b), (d)
7. (b), (d)
8. (a), (c)
9. (b)
10. (b)
11. (b)

Proiectarea aplicaţiilor software

1. (a), (d), (e)
2. (b), (c)
3. (a), (b), (d)
4. (b), (d)
5. (a), (c), (e), (f)
6. (b)
7. (c)
8. (b)
9. (c)
10. (a),(c)
11. (a),(b),(e),(f)
12. (a),(d),(f)
13. (d)
14. (a)
15. (a), (c), (e), (f), (h)

Baze de date

1. (c)
2. (d)
3. (a), (c)
4. (b)
5. (b), (c), (d)
6. (a), (b), (f)
7. (c)
8. (a), (c), (d)
9. (b)
10. (a), (b), (d)
11. (b)
12. (d), (e)
13. (b)
14. (b)

Arhitectura calculatoarelor

1. Registrul Program Contor conține:
   (a) adresa următoarei instrucțiuni din memorie
   (b) adresa datelor extrase din memorie
   (c) rezultatul operațiilor aritmetice efectuate
   (d) adresa de unde va fi extrasă instrucțiunea ce urmează a fi executată
2. Timpul mediu necesar pentru a accesa o locație de memorie și a prelua conținutul acesteia poartă numele de:
   (a) timp de latență
   (b) timp de căutare
   (c) timp de acces
   (d) timp de răspuns
3. Tipul de memorie folosit pentru a crește viteza de procesare a unui calculator este:
   (a) RAM
   (b) Cache
   (c) BIOS
   (d) ROM
4. Suma dintre -6 şi – 13, folosind reprezentarea în complement faţă de 2, este:
   (a) 11101101
   (b) 11100001
   (c) 01010101
   (d) 10101011
5. Care este valoarea stocată la adresa c după executarea următoarei secvențe de cod?

```
x      DATA   4B
y      DATA   F
c      DATA   0
start  LDA    x
       CMP    y
       JGE    adr1
       ADD    y
       STA    c
       JMP    adr2
adr1   SUB    y
       STA    c
adr2   STOP
       END    start
```

(a) $3C_{16}$
(b) 20_8
(c) 59_{10}
(d) 111100

6. Ce tehnică ajută procesorul să ruleze un program concomitent cu operațiile de intrare/ieșire?

(a) DMA
(b) transfer prin program
(c) niciunul dintre răspunsuri
(d) transfer prin întreruperi

7. Care este cea mai mare unitate de transfer a datelor din memoria unui sistem de calcul?

(a) cuvânt
(b) bloc
(c) bit
(d) niciuna dintre acestea

8. Pentru un număr dat/fix de biți, numărul de valori distincte care pot fi reprezentate este:

(a) mai mare pentru întregi fără semn
(b) mai mare pentru întregi cu semn
(c) același pentru întregi cu semn și întregi fără semn

9. Diferența între procesoarele CISC și RISC constă în:

(a) procesoarele CISC au un număr mai mare de instrucțiuni
(b) procesoarele RISC prezintă riscuri mai mari în utilizare
(c) procesoarele RISC execută o instrucțiune într-un singur ciclu

10. O întrerupere hardware reprezintă:

(a) un semnal sincron/asincron la un periferic care semnalizează apariția unui eveniment care trebuie tratat de către procesor
(b) întreruperea funcționării procesorului în urma unui bug software
(c) întreruperea unui circuit de conectare între două sau mai multe componente hardware

11. Arhitectura Harvard are:

(a) două magistrale, una pentru date și una pentru instrucțiuni
(b) două memorii, una pentru date și una pentru instrucțiuni/program

(c) drept dezavantaj major faptul că magistrala de date este mai ocupată decât magistrala de program

12. La magistralele sincrone:

(a) ciclurile de transfer sunt direct corelate cu semnalul de tact (CPU clock)

(b) nu există o legatură directă între evoluția în timp a unui ciclu de transfer și semnalul de tact al ceasului procesorului

(c) se face permanent o sincronizare a acestora cu magistralele asincrone

13. O poartă logică este:

(a) un program software capabil sa efectueze o anumită operație (logică) asupra unor semnale electrice

(b) un conector care permite preluarea valorilor logice de la utilizator

(c) un dispozitiv hardware capabil să efectueze o anumită operație (logică) asupra unor semnale electrice

14. Circuitele de memorare SR-latch (S-Set, R-Reset):

(a) Stochează o singură valoare binară

(b) Ieșirile celor două porți logice folosite pentru realizare sunt tot timpul complementare

(c) Setează si resetează în mod continuu valorile numerice stocate

15. Circuit combinațional:

(a) Este un circuit format din porți logice, care nu are memorie și al cărui output depinde doar de inputul prezent/curent, NU de inputul precedent sau de starea curentă în care se afla circuitul

(b) Este un circuit electric in care informațiile sunt combinate în mod aleator pentru a obține valori aleatoare

(c) Este un circuit care transpus într-un graf nu conține cicluri

16. Procesoarele RISC se deosebesc de procesoarele CISC prin:

(a) un număr mai mare de regiștrii

(b) mai multe moduri de adresare a memoriei

(c) mai puține moduri de adresare a memoriei

(d) execuția mai rapidă a operațiilor aritmetice și logice

17. În cazul unui circuit full-adder:

(a) niciodată sum și carry nu pot avea simultan valorile 1

(b) se ia în considerare transportul precedent (carry-in) ca și intrare

(c) se folosesc doua circuite half-adder

(d) sum și carry pot avea simultan valorile 1

18. Un circuit de memorare de tip SR-latch:

(a) poate fi realizat numai cu porți NAND

(b) poate stoca o valoare de tip real

(c) poate stoca doar o valoare de tip Boolean

19. Un microprocesor îndeplinește următoarele funcții:

(a) manipulare informaţii (instrucţiuni, date transmise, date primite)

(b) execuţie operaţii de calcul

(c) control şi supervizare

(d) verifica semantica unui program software

20. Semnalele unui microprocesor sunt

(a) semnale de întrerupere

(b) semnale pentru arbitrarea magistralei

(c) semnale de verificare conexiune Internet

21. În cazul unei magistrale sincrone:

(a) producerea unui eveniment depinde de producerea evenimentului anterior;

(b) toate dispozitivele conectate la magistrală pot citi linia de sincronizare

(c) doar dispozitivele de intrare conectate la magistrală pot citi linia de sincronizare

22. În cazul familiei de procesoare x86, registrul pointer de instrucțiuni (IP – Instruction Pointer)

(a) conține adresa primei instrucțiuni de executat dintr-un program

(b) nu poate fi modificat sau citit în mod direct

(c) indică adresa de deplasament plecând de la adresa conținută în registrul CS (adresa de baza)

23. Care este rolul unității de comandă și control în cadrul UCP?

(a) Stochează instrucțiunile programului

(b) Decodifică instrucțiunile programului

(c) Efectuează operații logice

(d) Toate variantele enumerate

24. Care dintre următoarele unități de memorie comunică direct cu UCP?

    (a) Memoria auxiliară
    (b) Memoria principală
    (c) Memoria secundară
    (d) Niciuna dintre variantele enumerate

25. O magistrală a unui sistem de calcul constă din:

    (a) Un set de linii paralele
    (b) Acumulatori
    (c) Registre
    (d) Niciuna dintre variantele enumerate

26. Reprezentarea în virgulă mobilă, simplă precizie, a numărului zecimal $-12,25$ este:

    (a) 1 10000010 10001000000000000000000
    (b) 1 01111100 10001000000000000000000
    (c) 0 10000010 10001000000000000000000

Sisteme de operare

1. Care dintre următoarele mecanisme pot fi folosite pentru o implementare corectă a problemei secţiunii critice fără utilizarea asteptării active?

    (a) semafoare
    (b) algoritmul lui Dekker
    (c) algoritmul lui Peterson
    (d) mecanismul TSL
    (e) monitoare

2. Se consideră 5 segmente de memorie (A, B, C, D, E) cu mărimile $2k$, $6k$, $4k$, $3k$, $4k$ şi o memorie totală de $16k$. În memorie se încarcă în ordine segmentele A, B, C, D. Care dintre aceste segmente urmează să fie evacuat pentru a putea permite încărcarea ulterioară a segmentului E?

    (a) A
    (b) B
    (c) C
    (d) D
    (e) niciunul

3. Care dintre problemele clasice de comunicare între procese pune în evidenţă o situaţie de impas?

(a) Problema filosofilor la masă

(b) Problema cititorilor şi scriitorilor

(c) Problema producător-consumator

(d) Problema frizerului

(e) niciuna dintre aceste probleme

4. Se consideră un set de 10 procese planificate conform algoritmului Round Robin. Se cunosc următoarele informaţii: cuanta utilizată este 8, numărul final de comutări de context este 25, valoarea raportului $T_1/T_2 = 2.25$, unde T_1 şi T_2 sunt timpii estimaţi de execuţie pentru cel mai lung si cel mai scurt proces, pentru care timpul estimat de execuţie depăşeşte valoarea cuantei (cu cel putin 25%). Determinaţi cea mai mică valoare posibilă pentru perechea (T_1, T_2).

(a) 27, 12

(b) 31, 14

(c) 34, 15

(d) 36, 16

5. Se consideră un set de 10 procese planificate conform algoritmuluii SJF, în care procesele sosesc în sistem la momentele $T_1 = 0, T_2 = 13, T_3 = 26$. La fiecare moment sosesc cel puţin 3 procese, cu timpi estimaţi de execuţie diferiţi. Dacă pentru procesele P_1, P_5 şi P_7 avem timpii de aşteptare $0, 0, 0$ şi timpii de răspuns $14, 3, 6$, identificaţi la ce moment a sosit procesul P_1, apoi răspundeţi la următoarea întrebare: determinaţ timpii estimaţi de execuţie pentru cel puţin 6 procese, ştiind că nu există perioade de pauză ale procesorului şi că pot exista procese cu acelaşi timp estimat de execuţie, sosite la momente diferite?

(a) 3,4,5,6,7,8

(b) 3,4,5,6,6,8

(c) 3,4,5,6,7,14

(d) 3,4,6,6,7,14

6. Într-un sistem pe 16 biţi cu 8 pagini virtuale şi 4 pagini cadru este folosit pentru paginare algoritmul FIFO. La un moment dat, în memorie sunt mapate paginile $(2, 5, 7, 1)$. După acest moment vor fi accesate următoarele adrese: $12411, 16729, 2560, 9215, 20952$. Care va fi maparea paginilor după ultima accesare, ştiind că următoarea pagină evacuată este prima pagină cadru?

(a) 5 2 0 4

(b) 3 4 2 5

(c) 2 5 0 3

(d) 5 4 0 2

7. Într-un sistem pe 16 biţi cu 8 pagini virtuale şi 4 pagini cadru este folosit pentru paginare algoritmul Ceasului, fără îmbătrânire. La un moment dat, în memorie sunt mapate paginile $(2, 5, 7, 1)$ iar următoarea pagină testată va fi prima pagină cadru. Dupa încercările de acces către paginile, nu neaparat în această ordine, $1, 2, 3, 4, 5, 7$ maparea devine $(1, 5, 7, 2)$. Care dintre cele 4 pagini iniţiale au avut bitul R setat?

   (a) toate
   (b) niciuna
   (c) paginile 5 si 7
   (d) paginile 2, 5 si 7

8. Se consideră situaţia a 3 fire de control a execuţiei care partajează variabilele A şi B. Accesul către cele două variabile este nerestricţionat, iar operaţiile realizate sunt, în fiecare fir de control a execuţiei, următoarele $A = A+B, B = A-B, A = A-B$. Precizaţi care dintre următoarele valori sunt posibile la sfârşitul execuţiei celor trei fire de control a execuţiei, atunci când $A = 5, B = 7$

   (a) $A = 5, B = 7$
   (b) $A = 7, B = 5$
   (c) $A = -29, B = 17$
   (d) $A = -31, B = 19$

9. Se consideră o situaţie simplă cu 6 procese şi un singur tip de resurse. Starea sistemului este descrisă prin $Are = (4, 2, 0, 5, 1, 4), Max = (8, 10, 10, 25, 25, 30), Disponibil = D$. Determinaţi valoarea minimă D pentru care starea este sigură şi permite apoi alocarea a 3 resurse suplimentare către procesul al doilea.

   (a) $D = 12$
   (b) $D = 14$
   (c) $D = 13$
   (d) $D = 11$

10. Se consideră o situaţie simplă cu 6 procese şi un singur tip de resurse. Starea sistemului este descrisă prin $Are = (4, 2, 0, 5, 1, 4), Max = (14, 6, 8, 29, 27, 24), Disponibil = D$. Determinaţi valoarea minimă D pentru care starea este sigură şi permite apoi alocarea a 3 resurse suplimentare către procesul al treilea.

    (a) $D = 11$
    (b) $D = 13$
    (c) $D = 14$
    (d) $D = 12$

11. Care dintre următoarele reprezintă operații operații atomice permise asupra semafoarelor?

(a) wait
(b) up
(c) sleep
(d) down
(e) niciuna dintre acestea

12. O instrucțiune TSL ar trebui să fie executată?

(a) după fiecare proces în parte
(b) periodic
(c) într-o manieră atomică
(d) în niciuna dintre situațiile precizate

13. O unitate de execuție care nu poate fi intreruptă este

(a) simplă
(b) statică
(c) atomică
(d) dinamică
(e) niciuna dintre acestea

14. Precizați principalele dezavantaje ale spinlock-urilor

(a) nu sunt eficiente pentru multe multe procese
(b) implică tehnici de așteptare activă
(c) pot fi nesigure ocazional
(d) sunt prea complexe pentru programatori
(e) niciuna dintre acestea

15. Fiecare dintre procesele P_i, cu $i = 0, 1, \ldots, 9$ este caracterizat prin următoarea secvență de cod

```
repeat
  down(mutex)
  {Critical Section}
  up(mutex)
forever
```

Într-un al 11-lea proces, P_{10}, liniile $up(mutex)$ și $down(mutex)$ au fost inversate. Care este numărul maxim de procese care s-ar putea găsi simultan în regiunea critică în acest caz, știind că inițial este permisă trecerea prin mutex?

(a) 1

(b) 2

(c) 3

(d) Oricare dintre variantele a), b), c)

(e) Niciuna dintre variantele a), b), c)

16. Într-un sistem se găsesc trei procese concurente, în care se găsesc următoarele bucăți de cod, sincronizate prin trei semafoare binare. Semafoarele sunt inițializate cu valorile $S_0 = 1$, $S_1 = 0$ și $S_2 = 0$.

```
Process P0
while(true)
{
down(S0);
print '0';
up(S1);
up(S2);
}

Process P1
down(S1);
up(S0);

Process P2
down(S2);
up(S0);
```

De câte ori va executa primul proces linia *print '0'*?

(a) Cel puțin de două ori

(b) Exact de două ori

(c) Exact de trei ori

(d) Cel mult de trei ori

(e) Exact o dată

17. Modificarea unui semafor este posibilă în

(a) secțiunea critică

(b) zona aflată înaintea unei secțiuni critice

(c) zona situată după secțiunea critică

(d) oricare dintre aceste zone

(e) niciuna dintre aceste zone

18. Se consideră un set de patru procese, cu timpi estimați de execuție $2, 3, 4$ și 5. Fiecare dintre procese are două execuții succesive separate printr-un ciclu de operații de intrare/ieșire. Planificarea proceselor este FIFO. Știind că timpii de răspuns, după cea de-a doua execuție, raportat la momentul 0, sunt $30, 28, 20, 25$, precizați care sunt duratele operațiilor de intrare/ieșire pentru două dintre aceste procese.

(a) 7 și 13

(b) 17 și 14

(c) 5 și 17

(d) 9 și 15

19. Se consideră un set de patru procese, cu timpi estimați de execuție $2, 3, 4$ și 5. Fiecare dintre procese are două execuții succesive separate printr-un ciclu de operații de intrare/ieșire. Planificarea inițială a proceselor este FIFO, după momentul 15, algoritmul de planificare este modificat la SJF. Știind că timpii de răspuns, după cea de-a doua execuție, raportat la momentul 0, sunt $22, 25, 20, 30$, și că timpii asociați operațiilor de intrare/ieșire sunt $3, 7, 13, 17$, precizați care dintre procese nu pot avea operații de intrare/ieșire de durată 13.

(a) Procesele 1 și 3

(b) Procesele 2 și 3

(c) Procesele 1 și 2

(d) Procesele 3 și 4

20. Se consideră un set de patru procese, cu timpi estimați de execuție $2, 3, 4$ și 5. Fiecare dintre procese are două execuții succesive separate printr-un ciclu de operații de intrare/ieșire. Planificarea proceselor este SJF. Știind că timpii de răspuns, după cea de-a doua execuție, raportat la momentul 0, sunt $22, 25, 20, 30$, precizați, în ordine crescătoare, care pot fi duratele operațiilor de intrare/ieșire pentru procesele 3 și 4, știind că procesul 4 a revenit în sistem înaintea procesului 2.

(a) Procesele 3 și 13

(b) Procesele 7 și 11

(c) Procesele 3 și 7

(d) Procesele 3 și 11

21. Într-un sistem sunt definite patru pagini cadru și opt pagini virtuale. La un moment dat, în timpul utilizării unui algoritm de paginare, după înlocuirea unei pagini din memorie cu pagina 3, maparea în memorie este $3, 6, 4, 2$. Știind că bitul R este setat și pentru a treia pagină cadru, estimați maparea paginilor în memorie după încercări de acces către paginile $3, 7, 2, 1, 3, 5, 0$ (cu mențiunea că primul moment este cel din enunț), folosind algoritmul FIFO

(a) 0, 7, 5, 1
(b) 0, 7, 1, 5
(c) 5, 1, 7, 0
(d) 3, 6, 4, 2
(e) 2, 4, 6, 3

22. Într-un sistem sunt definite patru pagini cadru și opt pagini virtuale. La un moment dat, în timpul utilizării unui algoritm de paginare, după înlocuirea unei pagini din memorie cu pagina 3, maparea în memorie este $3, 6, 4, 2$. Știind că bitul R este setat și pentru a treia pagină cadru, estimați maparea paginilor în memorie după încercări de acces către paginile $3, 7, 2, 1, 3, 5, 0$ (cu mențiunea că primul moment este cel din enunț), folosind algoritmul 'a doua șansă' (SC)

(a) 3, 0, 1, 2
(b) 2, 1, 0, 3
(c) 3, 6, 4, 2
(d) 3, 1, 0, 2
(e) 7, 3, 5

23. Într-un sistem sunt definite patru pagini cadru și opt pagini virtuale. La un moment dat, în timpul utilizării unui algoritm de paginare, după înlocuirea unei pagini din memorie cu pagina 3, maparea în memorie este $3, 6, 4, 2$. Știind că bitul R este setat și pentru a treia pagină cadru, estimați maparea paginilor în memorie după încercări de acces către paginile $3, 7, 2, 1, 3, 5, 0$ (cu mențiunea că primul moment este cel din enunț), folosind algoritmul OPTIM (după ultimul moment se consideră că paginile vor fi accesate în ordine strict crescătoare, de la 0 la 7).

(a) 3, 1, 0, 2
(b) 3, 0, 1, 2
(c) 2, 1, 0, 3
(d) 3, 6, 4, 2
(e) 0, 1, 2, 3

24. Un calculator are 6 GB de RAM, din care sistemul de operare ocupă 1 GB. Procesele au toate 512 MB și au aceleași caracteristici. Dacă obiectivul este o utilizare a CPU de 98%, care este timpul maxim de așteptare I/O tolerat?

(a) 2%
(b) 85%
(c) 50%
(d) 67%

25. Se consideră un sistem cu 10 procese concurente. Fiecare proces petrece 60% din timpul său așteptând operații I/O. Care va fi procentul de utilizare al procesorului?

    (a) $\sim 100\%$
    (b) $\sim 60\%$
    (c) $\sim 40\%$
    (d) $\sim 10\%$

26. Într-un sistem cu fire de execuție, există câte o stivă pe fir sau o stivă pe proces atunci când sunt utilizate fire de execuție la nivel de utilizator și fire de execuție la nivel de kernel?

    (a) o stivă pe fir pentru ambele cazuri
    (b) o stivă pe fir pentru firele de execuție la nivel de utilizator, în timp ce firele de execuție la nivel de kernel folosesc o stivă pe proces
    (c) o stivă pe fir pentru firele de execuție la nivel de kernel, în timp ce firele de execuție la nivel de utilizator folosesc o stivă pe proces
    (d) o stivă pe proces pentru ambele cazuri

27. Să considerăm o linie de asamblare de fabricație în care diferiți muncitori efectuează diferite sarcini pentru a asambla un produs:
    (1) Furnizorii de componente care furnizează componentele necesare pentru procesul de asamblare,
    (2) Asamblorii care primesc componentele și le asamblează în produsul final,
    (3) Inspectorii, care verifică calitatea produselor finite pentru a se asigura că îndeplinesc standardele cerute,
    (4) Ambalatorii, care primesc produsele inspectate și le ambalează pentru expediere sau distribuție,
    (5) Transportatorii care primesc produsele ambalate și le pregătesc pentru transport către clienți sau comercianți. Prin relaționarea acestui model cu procesele din UNIX, ce formă de comunicare între procese descrie cel mai bine această interacțiune?

    (a) funcțiile fork pentru a crea procese
    (b) semnale
    (c) pipe-uri
    (d) fire de execuție

28. Se consideră următorul cod C; câte procese copil sunt create la executarea acestui program?

```
void main() {
fork();
fork();
exit();
fork();
}
```

(a) 2

(b) 3

(c) 4

(d) 8

29. Pentru adresa virtuală zecimală 20202, calculați numărul paginii virtuale și offset-ul, știind că dimensiunea unei pagini este 4 KB:

(a) 16, 3818

(b) 4, 3818

(c) 5, 202

(d) 20, 202

30. Aplicația de mai jos citește dintr-un fișier ce conține doar șirul de caractere "licență". Care va fi mesajul afișat pe ecran dacă citirea este reușită:

```
#define SIZE 4
 while((n = read(fd, buffer, SIZE)) > 0)
   {
       printf("%s\n",buffer);
   }
```

(a) licență

(b) lice

(c) lice
nță

(d) lice
nțăe

31. Următorul fragment de cod ar trebui să folosească pipe-uri pentru a comunica între procesul copil și procesul părinte. Este ceva în neregulă cu mecanismul de pipe-uri din fragmentul de cod de mai jos?

```
int pipe_fd[2];
char buffer[512];
pid_t pid=fork();
if(pid<0){
  perror("error");
  exit(1);
}
if(pid==0){
```

```
  write(pipe_fd[1],buffer,sizeof(buffer));
}
else{
  close(pipe_fd[0]);
  read(pipe_fd[0],buffer,sizeof(buffer));
  close(pipe_fd[1]);
  printf("%s\n",buffer);
  wait(NULL);
}
```

(a) pipe-ul nu este deschis

(b) capetele pipe-ului nu sunt închise în procesul copil

(c) capetele pipe-ului sunt corect închise în procesul părinte

(d) capetele pipe-ului sunt închise incorect în procesul părinte

32. Fragmentul de cod următor ar trebui să folosească pipe-uri pentru a comunica între procesul copil și procesul părinte. Este ceva în neregulă cu mecanismul de pipe-uri din fragmentul de cod de mai jos?

```
int pipe_fd[2];
char buffer[512];
pid_t pid=fork();
if(pid<0){
  perror("error");
  exit(1);
}
if(pid==0){
  write(pipe_fd[1],buffer,sizeof(buffer));
}
else{
  close(pipe_fd[0]);
  read(pipe_fd[0],buffer,sizeof(buffer));
  close(pipe_fd[1]);
  printf("%s\n",buffer);
  wait(NULL);
}
```

(a) pipe-ul nu este deschis

(b) capetele pipe-ului nu sunt închise în procesul copil

(c) capetele pipe-ului sunt corect închise în procesul părinte

(d) capetele pipe-ului sunt închise incorect în procesul părinte

33. Care dintre următoarele este o funcție a unui sistem de operare?

    (a) Managementul memoriei

    (b) Protecția antivirus

    (c) Managementul fișierelor

    (d) Managementul bazelor de date

34. Un grup de proiectanți de sisteme de operare ia în considerare modalități de reducere a dimensiunii memoriei necesare în noul lor sistem de operare. Liderul echipei a sugerat să nu se deranjeze să salveze secțiunea de text a programului în zona de swap, ci pur și simplu să-l pagineze direct din fișierul binar ori de câte ori este necesar. În ce condiții, funcționează această idee pentru textul programului?

    (a) funcționează pentru program dacă programul nu poate fi modificat

    (b) funcționează pentru date, dacă datele nu pot fi modificate

    (c) funcționează dacă datele sunt modificate și programul nu

    (d) funcționează dacă programul este modificat și datele nu sunt modificate

Rețele de calculatoare

1. Intr-un segment TCP ce câmp permite nodului receptor să determine dacă un segment TCP a fost deteriorat în timpul transmisiei?

    (a) suma de control;

    (b) flags;

    (c) hash;

    (d) padding;

2. Care dintre următoarele dispozitive se află la nivelul 2 OSI?

    (a) bridge(punte);

    (b) repeater(repetor);

    (c) router;

    (d) switch;

    (e) hub;

3. Pentru a asigura integritatea datelor, protocoalele orientate pe conexiune (ca TCP) folosesc:

    (a) semnătura digitala;

    (b) certificate digitale;

    (c) algoritmi de criptare simetrici;

(d) suma de control (checksum);

4. Ordinea corectă a încapsulării mesajelor este:

   (a) date, cadre, pachete, segmente, biţi;
   (b) segmente, date, pachete, cadre, biţi;
   (c) date, segmente, pachete, cadre, biţi;
   (d) date, segmente, cadre, pachete, biţi;

5. Ce afirmaţii sunt adevărate despre protocolul TCP?

   (a) este un protocol orientat de datagrame;
   (b) este un protocol orientat pe conexiune;
   (c) nu foloseşte sume de control;
   (d) asigura segmentare şi reasamblare;

6. Care din următoarele adrese IP se încadrează în blocul CIDR din 115.64.4.0/22? (Alegeti două.)

   (a) 115.64.8.32;
   (b) 115.64.6.255;
   (c) 115.64.8.31;
   (d) 115.64.5.128;

7. Ce informații sunt conținute de headerul unui frame Etherne?

   (a) sursa și destinația adresei de hardware;
   (b) sursa și destinația adresei de rețea;
   (c) codul de corectare a erorilor;
   (d) codul de autentificare;

8. Care din următoarele afirmații sunt adevărate referitor la switch?

   (a) crează un singur domeniu de coliziune și un singur domeniu de broadcast ;
   (b) crează diferite domenii de coleziune dar un singur domeniu de ibroadcast;
   (c) crează diferite domenii de coleziune și diferite domenii de broadcast;

9. Care din următoarele adrese IP este un exemplu valid de adresă IPv4 ?

   (a) 144.92.254.253;
   (b) 144-92-43-178;
   (c) 144.92.256.176;
   (d) 144,92,43,178;

10. Care este semnificția aconimului CSMA/CD?

    (a) Carrier Service Multiple Access with Collision Detection;
    (b) Carrier Sense Multiple Access with Collision Avoidance;
    (c) Carrier Sense Multiple Access with Collision Detection;
    (d) Control Sense Multiple Access with Collision Direction;

11. Cum se comportă un switch când primește un cadru pe o anumită interfață iar adresa destinație nu este cunoscută?

    (a) interoghează serverul DNS;
    (b) trimite pachetul în întreaga rețea (broadcast);
    (c) ignoră pachetul;
    (d) interoghează nodul sursă;
    (e) trimite pachetul pe prima interfață de rețea disponibilă;

12. Întârzierea de propagare se referă la:

    (a) Timpul necesar pentru a "pune" pe sârmă un mesaj de o anumită dimensiune;
    (b) Numărul de biți aflați în propagare cu o anumită întârziere;
    (c) Numărul de octeți transmiși de o aplicație;
    (d) Timpul necesar biților pentru a parcurge un canal de transmitere de o anumită lungime;

13. Întârzierea de transmitere se refera la:

    (a) timpul necesar pentru a pune pe canalul de transmisie un mesaj de dimensiune M;
    (b) viteza necesara biților pentru a se propaga peste canal de o anumita lungime;
    (c) timpul necesar biților de a parcurge dus-întors canalul de comunicare;
    (d) timpul suficient pentru a transmite un 8 de biți pe fir;;
    (e) timpul necesar a transmite un grup de biți peste un fir de o anumita lungime;

14. Presupunând ca distanța dintre Timișoara și Arad este de 47km de kilometrii și avem o rată de transfer de 500Mbps, latența transmiterii unui mesaj de 1MB este de este

    (a) $16.0128 milisecunde$
    (b) $0.546 Mb/s$
    (c) $16.235 milisecunde$
    (d) $273.008 secunde$
    (e) $8 * 10^3 milisecunde$

15. Doriți să folosiți full-duplex Ethernet în locul half-duplex. Care sunt avantajele pentru rețea?

    (a) mai multe domenii de coliziune;
    (b) ar trebui să fie mai rapid;
    (c) elimină nevoia de switch-uri;
    (d) mai puține domenii de broadcast;

16. Calculați BDP pentru o conexiune cu următoarele repere: R=1Gbps, Distanta=100km

    (a) $10 * 2^2 Mb$
    (b) 50MB
    (c) 0.5Mb
    (d) 125b
    (e) 10MB

17. Un ISP deține o rețea de 100Mbps și dorește să aloce fiecărui client 50Mbps din lățimea de bandă disponibilă. Presupunem că fiecare client folosește 75% din lățimea de bandă, în medie de utilizare independentă. Care este probabilitatea statistică ca toată lățimea de bandă să fie utilizată?

    (a) 81/256
    (b) 16/128
    (c) 27/16
    (d) 1
    (e) 9/16

18. Produsul Lățime de Bandă – Întârziere (BDP) măsoară:

    (a) Cantitatea de date aflată în tranzit la un moment dat;
    (b) Dimensiunea maximă a unui mesaj transmis;
    (c) Cantitatea de date ce poate fi transmisă la un moment dat de un emițător;
    (d) Numărul de octeți recepționați după o anumită întârziere;
    (e) Viteza cu care sunt transmise datele la o anumită întârziere;

19. Calculați BDP pentru o conexiune cu următoarele repere: R=500Mbps, Distanța=100km

    (a) 250Kb
    (b) $10 * 2^2 Mb$
    (c) 250KB
    (d) 125b

(e) 0.5Mb

20. În modelul OSI nivelul rețea se ocupă cu:

    (a) Trimiterea de mesaje între două dispozitive
    (b) Trimiterea de mesaje peste mai multe legături
    (c) Codificarea semnalului folosind analiză Fourier
    (d) Crearea unei conexiuni între două aplicații
    (e) Încapsularea mesajelor primite de la nivelul legături de date

21. Care este topologia rețelei unde fiecare nod este legat de cele mai apropiate două noduri în așa fel ca toată rețeaua să formeze un cerc?

    (a) magistrala;
    (b) inel;
    (c) magistrala-stea;
    (d) stea;
    (e) LAN;

22. Care dintre afirmațiile de mai jos sunt adevărate ?

    (a) Nivelul Fizic se preocupă cu transmiterea unui flux de biți peste un mediu de transmisie;
    (b) Nivelul Fizic se preocupă cu transmiterea fișierelor peste un mediu de transmisie;
    (c) Nivelul Fizic se preocupă cu segmentarea și compresia de octeți peste un mediu de transmisie;
    (d) Nivelul Fizic se preocupă cu codificarea semnalelor sub formă de octeți;
    (e) Nivelul Fizic se preocupă cu codificarea semnalului optic sub formă de impulsuri electrice binare;

Răspunsuri

Arhitectura calculatoarelor

1. (d)
2. (c)
3. (b)
4. (a)
5. (a), (d)
6. (d)
7. (b)
8. (c)
9. (a), (c)
10. (a)
11. (a), (b), (c)
12. (a)
13. (c)
14. (a), (b)
15. (a), (c)
16. (c)
17. (a), (b), (c)
18. (c)
19. (a), (b), (c)
20. (a), (b)
21. (b)
22. (b), (c)
23. (b)
24. (b)
25. (a)
26. (a)

Sisteme de operare

1. (a),(e)
2. (b),(c),(d)
3. (a)
4. (a)
5. (d)
6. (d)
7. (c)
8. (b),(d)
9. (b)
10. (c)
11. (b), (d)
12. (c)
13. (c)
14. (b)
15. (d)
16. (a)
17. (a)
18. (a)
19. (d)
20. (c)
21. (b)
22. (d)
23. (a)

Sisteme de operare

24. (d)
25. (a)
26. (a)
27. (c)
28. (b)
29. (b)
30. (d)
31. (a), (b), (d)
32. (a), (b), (c)
33. (a), (c)
34. (a), (b)

Rețele de calculatoare

1. (a)
2. (a),(d)
3. (d)
4. (c)
5. (b),(d)
6. (b),(d)
7. (a)
8. (b)
9. (a)
10. (c)
11. (b)
12. (d)
13. (a)
14. (c)
15. (a), (b)
16. (c)
17. (e)
18. (a)
19. (a)
20. (b)
21. (b)
22. (a)