

Examen de licență 2017 - Informatică

Exemple de întrebări - Structuri discrete și algoritmi

In atenția studenților:

Proba scrisă a examenului de licență din sesiunile iulie-septembrie 2017 va consta din 60 de întrebări similare, ca structură și nivel de dificultate, celor din această culegere. Pentru fiecare dintre cele trei categorii (Structuri discrete și algoritmi, Limbaje de programare și inginerie software, Sisteme de calcul și baze de date) vor fi câte 20 de întrebări.

Pentru neclarități privind enunțurile sau răspunsurile puteți să vă adresați celor care au propus întrebările pentru fiecare secțiune.

Algoritmi:

- Daniela Zaharie (daniela.zaharie@e-uvv.ro)

Bazele informaticii:

- Monica Tirea (monica.tirea@e-uvv.ro)

Structuri de date:

- Cosmin Bonchiș (cosmin.bonchis@e-uvv.ro)

Teoria grafurilor și combinatorică:

- Mircea Marin (mircea.marin@e-uvv.ro)

Limbaje formale și teoria automatelor:

- Mircea Drăgan (mircea.dragan@e-uvv.ro)

1 Algoritmi

1. Se consideră secvența:

```
a:=1
WHILE a<>n DO
  a=2*a
ENDWHILE
```

Pentru ce valori ale lui n ciclul de mai sus NU este infinit?

- (a) pentru orice valoare naturală nenulă
- (b) pentru orice valoare naturală nenulă pară
- (c) pentru orice putere a lui 2
- (d) pentru orice putere pară a lui 2
- (e) pentru orice putere impară a lui 2

2. Se consideră secvența:

```
a:=0
FOR i:=1,n DO
  i:=i-h
  a:=a+1
ENDFOR
```

Pentru ce valori ale variabilei întregi h ciclul de mai sus este infinit?

- (a) pentru $h=0$
- (b) pentru orice h negativ
- (c) pentru nici o valoare a lui h
- (d) pentru orice h mai mare sau egal cu 1

3. Se consideră secvența:

```
c:=a MOD q
a:=a DIV q
WHILE a>0 DO
  IF c<a MOD q THEN c:=a MOD q ENDIF
  a:=a DIV q
ENDWHILE
```

Știind că variabila a conține o valoare naturală, iar q este o valoare naturală cuprinsă între 2 și 10, după execuția secvenței, variabila c va conține:

- (a) cifra de pe poziția cea mai puțin semnificativă din reprezentarea în baza q a numărului a
 - (b) cifra de pe poziția cea mai semnificativă din reprezentarea în baza q a numărului a
 - (c) cifra minimă întâlnită în reprezentarea în baza q a numărului a
 - (d) cifra maximă întâlnită în reprezentarea în baza q a numărului a
 - (e) cel mai mare multiplu al lui q care îl divide pe a
4. Se consideră un tablou $b[0..n]$ care conține cifrele reprezentării binare a unui număr natural ($b[n]$ corespunde celei mai semnificative cifre, iar $b[0]$ celei mai puțin semnificative cifre) și care are proprietatea că există cel puțin un indice i pentru care $b[i]=0$. Stabiliți care dintre afirmațiile de mai jos sunt adevărate pentru algoritmul următor:

```
alg(b[0..n])
  i:=0
  WHILE b[i]=1 DO
    b[i]:=0
    i:=i+1
  ENDWHILE
  b[i]:=1
  RETURN b[0..n]
```

- (a) Transformă toate elementele din $b[0..n]$ care sunt egale cu 1 în 0
 - (b) Aparține clasei de complexitate $\Theta(n)$
 - (c) Aparține clasei de complexitate $\mathcal{O}(n)$
 - (d) Realizează incrementarea cu 1 a valorii naturale reprezentate în baza doi prin tabloul $b[0..n]$
5. Se consideră un tablou $b[0..n]$ care conține cifrele reprezentării binare a unui număr natural ($b[n]$ corespunde celei mai semnificative cifre, iar $b[0]$ celei mai puțin semnificative cifre) și care are proprietatea că există cel puțin un indice i pentru care $b[i]=1$. Stabiliți care dintre afirmațiile de mai jos sunt adevărate pentru algoritmul următor:

```
alg(b[0..n])
  i:=0
  WHILE b[i]=0 DO
    b[i]:=1
    i:=i+1
  ENDWHILE
  b[i]:=0
  RETURN b[0..n]
```

- (a) Aparține clasei de complexitate $\mathcal{O}(n)$
- (b) Aparține clasei de complexitate $\Theta(n)$
- (c) Realizează decrementarea cu 1 a valorii naturale reprezentate în baza doi prin tabloul $b[0..n]$
- (d) Transformă toate elementele din $b[0..n]$ care sunt egale cu 0 în 1

6. Se consideră secvența:

```
d:=m
i:=n
r:=d MOD i
WHILE r<>0 DO
  d:=i
  i:=r
  r:=d MOD i
ENDWHILE
```

Pentru ce valori ale lui m și n , variabila i va conține după execuția prelucrărilor cel mai mare divizor comun al lui m și n ?

- (a) pentru m și n naturale nenule ce satisfac $m < n$
- (b) pentru m și n naturale nenule ce satisfac $m > n$
- (c) pentru orice numere naturale nenule m și n
- (d) pentru nici o valoare

7. Se consideră secvența:

```
FOR d:=2,n DO
  IF n MOD d=0 THEN
    WRITE d
    WHILE (n MOD d=0) DO
      n:=n DIV d
    ENDWHILE
  ENDIF
ENDFOR
```

Pentru un număr natural n , prin execuția secvenței se vor afișa:

- (a) toți divizorii pozitivi ai lui n
- (b) toți divizorii lui n
- (c) toți divizorii naturali ai lui n care sunt numere prime

- (d) toți divizorii naturali ai lui n care sunt numere impare
- (e) nici o valoare

8. Fie $x[1..n]$ un tablou cu elemente întregi și secvența:

```
FOR i:=1,m DO
  a:=x[i]
  x[i]:=x[n-i+1]
  x[n-i+1]:=a
ENDFOR
```

Ce valoare ar trebui să aibă m astfel încât efectul secvenței de mai sus să fie inversarea ordinii elementelor tabloului și numărul de interschimbări să fie cât mai mic?

- (a) n
- (b) $n \text{ DIV } 2$
- (c) $n \text{ DIV } 2 + 1$
- (d) $3n \text{ DIV } 2$
- (e) nici o valoare a lui m nu asigură inversarea ordinii elementelor tabloului

9. Se consideră un tablou $x[1..n]$ și secvența de prelucrări:

```
k1:=1
k2:=1
FOR i:=2,n DO
  IF x[k1]>x[i] THEN k1:=i
  ELSE IF x[k2]<=x[i] THEN k2:=i ENDIF
ENDIF
ENDFOR
```

Care dintre următoarele afirmații sunt adevărate după execuția prelucrărilor?

- (a) $x[k1] \leq x[k2]$
- (b) $x[k1] \geq x[k2]$
- (c) $k1$ este ultima poziție pe care se află valoarea minimă din tablou, iar $k2$ este prima poziție pe care se află valoarea maximă din tablou
- (d) $k1$ este prima poziție pe care se află valoarea minimă din tablou, iar $k2$ este ultima poziție pe care se află valoarea maximă din tablou
- (e) $k1$ este prima poziție pe care se află valoarea maximă din tablou, iar $k2$ este ultima poziție pe care se află valoarea minimă din tablou
- (f) $k1$ este ultima poziție pe care se află valoarea maximă din tablou, iar $k2$ este prima poziție pe care se află valoarea minimă din tablou

10. Fie $x[1..n]$ un tablou cu elemente reale. Se consideră secvența de prelucrări:

```

i:=1
j:=n
WHILE i<j DO
  WHILE (x[i]<0) AND (i<n) DO i:=i+1 ENDWHILE
  WHILE (x[j]>0) AND (j>1) DO j:=j-1 ENDWHILE
  IF i<j THEN
    aux:=x[i]
    x[i]:=x[j]
    x[j]:=aux
  ENDIF
ENDWHILE

```

Ce efect are secvența de prelucrări asupra tabloului x ?

- (a) ordonează crescător elementele lui x
 - (b) elimină toate valorile pozitive din x
 - (c) elimină toate valorile negative din x
 - (d) plasează toate elementele negative înaintea elementelor pozitive;
 - (e) plasează toate elementele pozitive înaintea celor negative
11. Se consideră o matrice pătratică, $a[1..n,1..n]$, cu elemente din $\{0,1\}$ și algoritmul:

```

alg (a[1..n,1..n])
FOR r:=1,n-1 DO
  s1:=0; s2:=0;
  FOR i:= r+1,n DO
    s1:=s1+a[i,i-r]
    s2:=s2+a[i-r,i]
  ENDFOR
  IF NOT(s1=1) OR NOT(s2=1) THEN RETURN False ENDIF
ENDFOR
RETURN True

```

Care dintre următoarele afirmații este(sunt) adevărată(e) pentru algoritmul de mai sus:

- (a) Algoritmul returnează True în toate cazurile în care fiecare linie și fiecare coloană a matricii conține o singură valoare egală cu 1
- (b) Algoritmul returnează True doar dacă fiecare dintre diagonalele paralele cu diagonala principală conține exact un element egal cu 1
- (c) Algoritmul execută de $n(n+1)/2$ ori corpul ciclului FOR după i

- (d) Algoritmul returnează False doar dacă toate diagonalele paralele cu diagonala principală a matricii conțin doar elemente egale cu 0
- (e) Algoritmul execută de $n(n-1)/2$ ori corpul ciclului FOR după i

12. Se consideră algoritmul:

```
FOR k:=0,n-1 DO
  FOR j:=1,n-k DO
    WRITE a[j+k,j]
  ENDFOR
ENDFOR
```

Numărul de afișări efectuate este:

- (a) $n^2/2$
- (b) $n + n^2/2$
- (c) $n(n - k)$
- (d) $n(n + 1)/2$

13. Se consideră algoritmul:

```
FOR i:=1,n-1 DO
  IF x[i]>x[i+1] THEN
    aux:=x[i]
    x[i]:=x[i+1]
    x[i+1]:=aux
  ENDIF
ENDFOR
```

Ordinul de complexitate al algoritmului în raport cu numărul de interschimbări efectuate este:

- (a) $\mathcal{O}(\lg n)$
- (b) $\mathcal{O}(n)$
- (c) $\mathcal{O}(n \lg n)$
- (d) $\mathcal{O}(n^2)$
- (e) $\Theta(n)$

14. Fie $x[1..n]$ un tablou cu elemente reale ordonate crescător ($n > 1$), iar v o valoare reală oarecare. Se consideră algoritmul:

```

s:=1
d:=n
r:=0
WHILE (s<=d) AND (r=0) DO
  m:=(s+d) DIV 2
  IF v=x[m] THEN r:=1
    ELSE IF v<x[m] THEN d:=m-1
      ELSE s:=m+1
    ENDIF
  ENDIF
ENDWHILE

```

Care dintre următoarele afirmații referitoare la algoritmul de mai sus sunt adevărate?

- (a) numărul minim de comparații cu valoarea v este 2
 - (b) ordinul de complexitate în raport cu numărul de comparații este $\mathcal{O}(n)$
 - (c) ordinul de complexitate în raport cu numărul de comparații este $\mathcal{O}(\lg n)$
 - (d) numărul minim de comparații cu valoarea v este 1
 - (e) variabila r conține valoarea 1, dacă v se află în tablou și 0 în caz contrar
 - (f) variabila r conține valoarea 0, dacă v se află în tablou și 1 în caz contrar
15. Fie $x[1..n]$ un tablou. Se consideră următorul algoritm:

```

nr:=0
i:=1
WHILE i<=n DO
  k:=0
  WHILE (x[i+k]>0) AND ((i+k)<n) DO k:=k+1 ENDWHILE
  IF nr<k THEN nr:=k ENDIF
  i:=i+k+1
ENDWHILE

```

Care este ordinul de complexitate al algoritmului?

- (a) $\mathcal{O}(n^2)$
 - (b) $\mathcal{O}(n)$
 - (c) $\mathcal{O}(\lg n)$
 - (d) $\mathcal{O}(n \cdot k)$
 - (e) $\mathcal{O}(n \cdot \lg k)$
16. Fie $x[1..n]$ un tablou neordonat. Se pune problema determinării unei perechi, $(imax, jmax)$, de indici din $\{1, 2, \dots, n\}$ care are proprietatea că $|x[imax] - x[jmax]|$ este maximă. Care dintre următorii algoritmi realizează acest lucru în $\mathcal{O}(n)$?

- (a) `imax:=1; jmax:=2`
`FOR i:=1,n-1 DO`
`FOR j:=i+1,n DO`
`IF |x[i]-x[j]|>|x[imax]-x[jmax]| THEN imax:=i; jmax:=j ENDIF`
`ENDFOR`
`ENDFOR`
- (b) `imax:=1; jmax:=1`
`FOR i:=2,n DO`
`IF x[imax]>x[i] THEN imax:=i ENDIF`
`IF x[jmax]<x[i] THEN jmax:=i ENDIF`
`ENDFOR`
- (c) `imax:=1; jmax:=1`
`FOR i:=2,n DO`
`IF x[imax]<x[i] THEN imax:=i ENDIF`
`IF x[jmax]>x[i] THEN jmax:=i ENDIF`
`ENDFOR`

17. Se consideră un tablou, $a[1..n]$, ordonat crescător și o valoare x . Se pune problema verificării existenței cel a puțin unei perechi de elemente $a[i]$ și $a[j]$ ($i \neq j$) care satisfac proprietatea $a[i]+a[j]=x$ și se consideră următorii algoritmi:

```
alg1 (a[1..n],x)
  i:=1; j:=n
  WHILE i<j DO
    IF a[i]=x-a[j] THEN RETURN True
    ELSE IF a[i]<x-a[j] THEN i:=i+1 ELSE j:=j-1 ENDIF
  ENDIF
ENDWHILE
RETURN False
```

```
alg2(a[1..n],x)
  FOR i:=1,n DO
    FOR j:=1,n DO
      IF a[i]+a[j]=x THEN RETURN True
    ENDFOR
  ENDFOR
RETURN False
```

```
alg3(a[1..n],x)
  FOR i:=1,n-1 DO
    FOR j:=i+1,n DO
      IF a[i]+a[j]=x THEN RETURN True
```

```
ELSE RETURN FALSE
ENDFOR
ENDFOR
```

Care dintre următoarele afirmații este(sunt) adevărată(e)?

- (a) Algoritmul alg1 rezolvă corect problema și are ordinul de complexitate $\mathcal{O}(n)$
- (b) Algoritmul alg3 are un ordin de complexitate mai mic decât algoritmul alg2
- (c) Algoritmul alg2 rezolvă corect problema și are ordinul de complexitate $\mathcal{O}(n^2)$
- (d) Algoritmul alg3 rezolvă corect problema și are ordinul de complexitate $\mathcal{O}(n^2)$

18. Se consideră secvența de prelucrări:

```
k:=0
i:=1
WHILE i<=n DO
    k:=k+1
    i:=2*i
ENDWHILE
```

Numărul de înmulțiri efectuate în secvența de mai sus este de ordinul:

- (a) $\mathcal{O}(n)$
- (b) $\mathcal{O}(n^2)$
- (c) $\mathcal{O}(\lg n)$
- (d) $\mathcal{O}(2n)$
- (e) $\mathcal{O}(n/2)$

19. Se consideră secvența de prelucrări:

```
s:=0
m:=1
FOR i:=1,n DO
    m:=2*m
    FOR j:=1,m DO s:=s+1 ENDFOR
ENDFOR
```

Numărul de incrementări ale variabilei s este de ordinul:

- (a) $\mathcal{O}(n)$
- (b) $\mathcal{O}(n^2)$
- (c) $\mathcal{O}(n \lg n)$

(d) $\mathcal{O}(2^n)$

(e) $\mathcal{O}(n^4)$

20. Se consideră următorii algoritmi pentru calculul factorialului unui număr natural:

```
fact1(n)
  f:=1
  FOR i:=2,n DO
    f:=f*i
  ENDFOR
  RETURN f
```

```
fact2(n)
IF n<=1 THEN RETURN 1
  ELSE RETURN n*fact2(n-1)
ENDIF
```

Care dintre următoarele afirmații este(sunt) adevărată(e) în condițiile în care în analiza complexității se ia în considerare doar operația de înmulțire?

(a) ordinul de complexitate al algoritmului `fact1` este mai mare decât cel al algoritmului `fact2`

(b) ordinul de complexitate al algoritmului `fact2` este mai mare decât cel al algoritmului `fact1`

(c) algoritmi `fact1` și `fact2` au același ordin de complexitate

21. Se consideră algoritmul:

```
alg(n)
  IF n>0 THEN
    alg(n DIV 2)
    WRITE n MOD 2
  ENDIF
```

Dacă este apelat pentru un număr natural nenul, n , algoritmul va afișa:

(a) cifrele corespunzătoare reprezentării în baza 2 a numărului n (începând de la cifra cea mai puțin semnificativă)

(b) cifrele corespunzătoare reprezentării în baza 2 a numărului n (începând de la cifra cea mai semnificativă)

(c) restul împărțirii lui n la 2

(d) câtul împărțirii lui n la 2

22. Se consideră algoritmul:

```
alg(n)
  IF n=0 THEN RETURN 0
    ELSE RETURN n MOD 2+alg(n DIV 2)
  ENDIF
```

Presupunând că algoritmul este apelat pentru un număr natural, n , care dintre afirmațiile de mai jos sunt adevărate?

- (a) algoritmul returnează numărul de cifre din reprezentarea binară a lui n
- (b) algoritmul returnează numărul de cifre egale cu 1 din reprezentarea binară a lui n
- (c) algoritmul returnează suma tuturor cifrelor din reprezentarea binară a lui n
- (d) algoritmul returnează numărul de cifre egale cu 0 din reprezentarea binară a lui n

23. Se consideră următorul algoritm recursiv care are acces la conținutul unui tablou $x[1..n]$:

```
alg(i, j)
  IF i<j THEN
    aux:=x[i]
    x[i]:=x[j]
    x[j]:=aux
    alg(i+1,j-1)
  ENDIF
```

Efectul apelului $\text{alg}(1,n)$ este:

- (a) interschimbă primul element cu ultimul element al tabloului $x[1..n]$
- (b) inversează ordinea tuturor elementelor tabloului $x[1..n]$
- (c) lasă tabloul x nemodificat
- (d) interschimbă elementele vecine din tabloul inițial

24. Se consideră un tablou $x[1..n]$ și algoritmul:

```
sort(x[1..n])
FOR i:=2,n DO
  aux:=x[i]
  j:=i-1
  WHILE (j>=1) AND <conditie> DO
    x[j+1]:=x[j]
    j:=j-1
  ENDWHILE
  x[j+1]:=aux
ENDFOR
```

Cu care dintre expresiile de mai jos trebuie completată condiția de la WHILE astfel încât algoritmul să realizeze ordonarea crescătoare a tabloului x ?

- (a) $x[j] \leq aux$
- (b) $x[j] \geq aux$
- (c) $x[j] < aux$
- (d) $x[j] > aux$

25. Se consideră un tablou ordonat crescător, $x[1..n]$, o valoare v și algoritmul:

```
alg(x[1..n], v)
  g:=0
  i:=1
  WHILE (g=0) AND (i<=n) DO
    IF v<=x[i] THEN g:=1 ELSE i:=i+1 ENDIF
  ENDWHILE
  RETURN i
```

Care dintre următoarele afirmații sunt adevărate?

- (a) algoritmul are complexitate liniară
- (b) la ieșirea din ciclu, g are valoarea 0 dacă și numai dacă $v > x[n]$
- (c) algoritmul returnează numărul de elemente din x care sunt strict mai mici decât v
- (d) algoritmul returnează poziția pe care poate fi inserat v astfel încât tabloul să rămână ordonat crescător
- (e) algoritmul returnează întotdeauna $(n+1)$

26. Se consideră un tablou $x[1..n]$ și algoritmul:

```
Alg(x[1..n])
  FOR i:=n,2,-1 DO
    FOR j:=1,i-1 DO
      IF x[j]<x[j+1] THEN
        aux:=x[j]
        x[j]:=x[j+1]
        x[j+1]:=aux
      ENDIF
    ENDFOR
  ENDFOR
  RETURN x[1..n]
```

Care dintre următoarele afirmații sunt adevărate?

- (a) algoritmul ordonează crescător tabloul $x[1..n]$
- (b) algoritmul ordonează descrescător tabloul $x[1..n]$
- (c) algoritmul nu realizează nici ordonarea crescătoare nici cea descrescătoare a tabloului;
- (d) indiferent de configurația inițială a tabloului algoritmul are complexitate liniară (aparține lui $\Theta(n)$);
- (e) indiferent de configurația inițială a tabloului algoritmul are complexitate pătratică (aparține lui $\Theta(n^2)$)

27. Se consideră o variabilă reală x , o variabilă naturală nenulă n și algoritmul:

```

alg(x,n)
  IF n=1 THEN RETURN x
  ELSE
    p:=alg(x,n DIV 2)
    IF n MOD 2=0 THEN RETURN p*p
    ELSE RETURN p*p*x
  ENDIF
ENDIF

```

Care dintre următoarele afirmații sunt adevărate?

- (a) algoritmul returnează x^2 dacă n este par și x^3 dacă n este impar
- (b) este posibil ca succesiunea de apeluri recursive să nu se termine
- (c) algoritmul returnează x^n dacă n este par x^{n+1} și dacă n este impar
- (d) algoritmul returnează x^n indiferent de paritatea lui n
- (e) algoritmul are complexitate liniară ($\mathcal{O}(n)$)
- (f) algoritmul are complexitate logaritmică ($\mathcal{O}(\lg n)$)

28. Se consideră două valori naturale a și b și algoritmul:

```

alg(a,b)
  IF a=0 OR b=0 THEN RETURN 0
  ELSE
    IF a MOD 2=0 THEN RETURN (alg(a DIV 2,2*b))
    ELSE RETURN (alg(a DIV 2,2*b)+b)
  ENDIF
ENDIF

```

Care dintre următoarele afirmații sunt 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 și $a*b+b$ dacă a este impar
 (d) algoritmul returnează produsul $a*b$ indiferent de paritatea lui a
29. Se consideră algoritmul `alg` apelat pentru un număr natural n și se notează cu $T(n)$ numărul de operații de adunare efectuate:

```
alg(n)
  IF n<=9 THEN RETURN n
  ELSE RETURN alg(n DIV 10)+n MOD 10
ENDIF
```

Care dintre următoarele afirmații este(sunt) adevărată(e):

- (a) Algoritmul returnează câte cifre nenule are numărul n
 (b) Algoritmul returnează suma cifrelor numărului n
 (c) $T(n) = 1$ dacă $n \leq 9$ și $T(n) = T(n \text{ DIV } 10) + n \text{ MOD } 10$ dacă $n > 9$
 (d) $T(n) = 0$ dacă $n \leq 9$ și $T(n) = T(\lfloor n/10 \rfloor) + 1$ dacă $n > 9$
 (e) $T(n)$ aparține lui $\mathcal{O}(n)$
 (f) $T(n)$ aparține lui $\mathcal{O}(\lg n)$
30. Se consideră că timpul de execuție al unui algoritm recursiv folosit pentru rezolvarea unei probleme de dimensiune n satisface următoarea relație de recurență:

$T(n) = 1$ dacă $n = 1$, respectiv $T(n) = nT(n - 1) + 2$ dacă $n > 1$

Stabiliți cărei clase de complexitate aparține $T(n)$:

- (a) $\mathcal{O}(2^n)$
 (b) $\mathcal{O}(n!)$
 (c) $\mathcal{O}(n^2)$
 (d) $\mathcal{O}(2n)$

2 Bazele informaticii

1. Informația
 - (a) este o formulă scrisă, susceptibilă de a aduce o cunoștință.
 - (b) există atunci când se precizează starea actuală a unui fenomen care se poate situa într-un număr finit de stări.
 - (c) este un mesaj despre anumite evenimente care au avut sau vor avea loc.
2. Informatica este
 - (a) știința prelucrării raționale a informației considerată ca suport al cunoștințelor umane și al comunicațiilor în domeniile tehnice, economice și sociale.
 - (b) știința procesării sistematice a informației, în special a procesării cu ajutorul calculatoarelor.
 - (c) știința care studiază relațiile cantitative, modelele de structură, de schimbare și de spațiu.
3. Entropia informațională ($E(X)$) a unui sistem X complet de evenimente este
 - (a) cantitatea medie de informație obținută prin precizarea unei stări, din n stări posibile ale unui sistem complet de evenimente.
 - (b) este exprimată de obicei în biți.
 - (c) mijloc de măsură a cantității de informație.
 - (d) cantitatea minimă de informație conținută într-un mesaj.
4. Care din afirmațiile următoare sunt adevărate?
 - (a) entropia informațională este o entitate nenegativă.
 - (b) entropia unui sistem este maximă dacă stările sistemului sunt echiprobabile.
 - (c) entropia produsului mai multor surse independente de informație este egală cu suma entropiilor fiecărei surse luate separat.
 - (d) stările imposibile ale unui sistem complet de evenimente nu influențează entropia sistemului.
5. Care din afirmațiile următoare sunt adevărate?
 - (a) Codul este o aplicație surjectivă definită pe mulțimea simbolurilor primare cu valori în mulțimea secvențelor de cod.
 - (b) Codul este o aplicație bijectivă definită pe mulțimea simbolurilor primare cu valori în mulțimea secvențelor de cod.
 - (c) Codul este o aplicație injectivă definită pe mulțimea simbolurilor primare cu valori în mulțimea secvențelor de cod.

6. Lungimea medie a cuvintelor de cod s_i , asociate simbolurilor primare x_i , cu probabilitatea de realizare $p(x_i)$ este dată de expresia
- $L(X) = \sum_{i=1}^n p(x_i) l(s_i)$
 - $L(X) = \sum_{i=1}^n p(x_i) \log_2 p(x_i)$
 - $L(X) = \sum_{i=1}^n m^{-l_i}$
7. Pentru a efectua o codificare neuniformă univocă, pentru o sursă de informație X , ale cărei stări le identifică prin n simboluri primare, întrebuintând m simboluri elementare, cu ajutorul cărora să transmitem o cantitate de informație $E(X)$ este necesar ca
- lungimea medie a cuvintelor de cod să fie $L(X) \geq \frac{E(X)}{\log_2 m}$
 - lungimea minima a secvențelor de cod să fie $L(X) > \frac{E(X)}{\log_2 2}$
 - lungimea minima a secvențelor de cod să fie $L(X) < \frac{E(X)}{\log_2 m}$
8. Sistemul de numerație este definit ca
- totalitatea regulilor de reprezentare a numerelor folosind un anumit set de simboluri distincte, numite alfabet; simbolurile sunt numite cifre.
 - un sistem lingvistic și un mod de notație matematică pentru reprezentarea numerelor folosind în mod coerent un set de simboluri.
 - totalitatea regulilor folosite pentru scrierea codurilor cu ajutorul unor simboluri.
9. Legile logice sau tautologiile sunt definite ca
- enunțurile compuse care au valoarea de adevăr adevărat, indiferent de valorile de adevăr ale componentelor sale.
 - enunțurile compuse care pot fi adevărate sau false în funcție de valorile de adevăr ale componentelor sale.
 - enunțurile compuse care au valoarea de adevăr fals, indiferent de valorile de adevăr ale componentelor sale.
10. Fie $X = 28$ și $Y = -15$. Să se calculeze, în complement față de 2, suma $X + Y$.
- 00001101
 - 10101011
 - 11110001
 - 00101011
11. Fie $X = -35$ și $Y = -17$. Să se calculeze, în complement față de 2, diferența dintre X și Y :
- 11101101

- (b) 10010010
(c) 11101110
(d) 10110100
12. Fie $X = BD56$ și $Y = 2D6$, două numere în baza 16. Produsul celor două numere este:
- (a) BAD5804
(b) 218F1E4
(c) 21C4O2C
(d) 2E12AE4
13. Care este reprezentarea în virgulă flotantă simplă precizie a numărului $N = 122,25$ în hexazecimal?
- (a) 43E25000
(b) 42F48000
(c) 21C4O2C1
(d) 2E12AE43
14. O imagine alb-negru se compune din 1024×256 pixeli. Fiecare pixel poate avea un nivel de gri dintre 64 posibile. Care este informația furnizată de un pixel:
- (a) 12
(b) 6
(c) 64
(d) 32
15. Fie dată mulțimea $X = \{x_1, x_2, \dots, x_p\}$ și mulțimea de simboluri $A = \{a_1, a_2, \dots, a_m\}$. Care este lungimea minimă n a secvențelor de cod care se pot realiza cu simboluri din A , pentru o codificare uniformă a elementelor lui X ?
- (a) n este cel mai mic întreg $\geq m$
(b) n este cel mai mare întreg $\geq \lceil \log_p m \rceil$
(c) n este cel mai mic întreg $\geq \lceil \log_m p \rceil$
16. Stabiliți lungimea minimă a secvențelor de cod alfanumeric uniform binar, care conțin literele mari ale alfabetului latin, cifrele zecimale și alte 20 de caractere speciale.
- (a) $n = 6$
(b) $n = 5$
(c) $n = 7$

17. Sursa de informație X poate emite trei simboluri x_1, x_2, x_3 cu probabilitățile $p(x_1) = 0.7, p(x_2) = 0.2, p(x_3) = 0.1$. Folosind o codificare binară neuniformă univocă, asociind secvențe binare la fiecare simbol primar, să se determine care este lungimea medie a secvențelor de cod.
- (a) $L(X) = 1.3$
 - (b) $L(X) = 2.33$
 - (c) $L(X) = 1.16$
18. O secvență de cod este reprezentată prin 32479210000. Care este cifra de control c corespunzătoare, știind că ponderile sunt (de la stânga la dreapta): 8, 7, 5, 4, 2, 1 și apoi se repetă? Formula de calcul a cifrei de control este $c = q - (\sum_{i=1}^p x_i a_i) \bmod q$ iar $q = 9$.
- (a) $c = 3$
 - (b) $c = 5$
 - (c) $c = 9$
19. Care din variantele următoare reprezintă codificarea mesajului "Azi 20" folosind codul EBCDIC
- (a) 11000001 10101001 10001001 01000000 11110010 11110000
 - (b) 111111 110001 110000 011001 111001 000010 000000
 - (c) 1000001 1111010 1101001 0100000 0110010 0110000
20. Care va fi reprezentarea în baza 10 a numărului $(63.2)_8$?
- (a) $(51.25)_{10}$
 - (b) $(30, 25)_{10}$
 - (c) $(34, 25)_{10}$
21. Se aruncă 2 zaruri. Care este cantitatea medie de informație obținută în urma aruncării cu zarurile, tinând cont că toate fețele au probabilitate egală de apariție.
- (a) $E(X) = 2 \log_2 6$
 - (b) $E(X) = \log_2 6$
 - (c) $E(X) = \log_2 12$
22. O cultură de seră este dirijată prin 4 factori de mediu, variabili după cum urmează: temperatura (12 trepte), iluminatul (8 trepte), îngrășământ (6 trepte), umiditate (7 trepte). Care este cantitatea de informație conținută într-o rețetă de mediu, în ipoteza că toate rețetele sunt egal posibile?
- (a) $E(X) = 12$ biți
 - (b) $E(X) = 6$ biți
 - (c) $E(X) = 7$ biți

23. De la un proces tehnologic se culeg valorile debitului, presiunii și temperaturii, transmițându-se spre procesare la un calculator numeric. Eșantioanele de date sunt culese la anumite intervale de timp și se codifică binar. În secvența de cod există câte o zonă de 2 biți pentru definirea fiecărui parametru, urmată de zone în care se trec valorile acestuia. Știind că temperatura variază în 30 de trepte, presiunea în 25 și debitul în 14, lungimea unei secvențe de cod este
- (a) 14 biți
 - (b) 20 biți
 - (c) 6 biți
24. Care este reprezentarea pe 8 biți în complement față de 2 a lui -1?
- (a) $N = 00000000$
 - (b) $N = 11111111$
 - (c) $N = 11111001$
25. Care este reprezentarea pe 8 biți în complement față de 2 a numărului -6 ?
- (a) 11111010
 - (b) 11111001
 - (c) 00000110
26. Care este reprezentarea în virgulă flotantă simplă precizie a numărului $N = 178,125$?
- (a) 0 10000110 011001000100000
 - (b) 0 10000001 010000000000000
 - (c) 0 10000111 110000010000000
27. Care este numărul reprezentat în virgulă flotantă simplă precizie: 1 10000001 0100000000000?
- (a) $N = -5$
 - (b) $N = -12$
 - (c) $N = +6$
28. Care din următoarele formule logice sunt tautologii?
- (a) $((P \vee Q) \wedge \neg Q) \Rightarrow P$
 - (b) $(P \Rightarrow Q) \Leftrightarrow (\neg Q \Rightarrow \neg P)$
 - (c) $(P \wedge (P \Rightarrow Q)) \Rightarrow Q$
 - (d) $P \vee \neg P$
29. Formula logică următoare $((P \Rightarrow Q) \wedge R) \Rightarrow ((\neg R \vee P) \Rightarrow Q)$ este
- (a) tautologie

- (b) antilogie
 - (c) realizabilă
30. Care dintre scopurile de mai jos nu se realizează prin codare:
- (a) adaptarea naturii diferite a sursei la natura canalului;
 - (b) multiplexarea mai multor semnale purtătoare de informație în vederea transmiterii pe același canal;
 - (c) compresia sursei;
 - (d) protejarea informației împotriva perturbațiilor;
 - (e) protejarea informației împotriva receptorilor neautorizați (secretizarea).

3 Structuri de date

/ * Se presupune, acolo unde este cazul, că fișierele care conțin prototipul funcțiilor apelate din bibliotecă C, vor fi corect incluse în programele care apelează funcțiile din testele enunțate */

1. Componentele unei structuri de date:
 - (a) trebuie să fie toate de același tip;
 - (b) pot fi la rândul lor structuri de date;
 - (c) pot fi toate de același tip;
 - (d) nu pot fi tipuri definite de către utilizator;
 - (e) pot avea unul din tipurile standard ale limbajului de programare utilizat.
2. Se numește listă liniară:
 - (a) structura de date în care inserările, ștergerile și orice alt acces se efectuează la unul din capetele listei;
 - (b) o colecție de $n \geq 0$ elemente de același tip, ale căror proprietăți structurale se reduc la pozițiile relative liniare (unidimensionale) ale elementelor în cadrul structurii;
 - (c) o structură de date asimilată cu un tablou în care elementele sunt memorate într-o zonă continuă de memorie, în locații succesive;
 - (d) o structură avansată de date ce se caracterizează printr-un grad de abstractizare ridicat.
3. Selectați care din următoarele operații sunt considerate operații de bază asupra listelor în general:
 - (a) Sortarea elementelor listei în ordinea crescătoare/ descrescătoare a valorilor anumitor câmpuri;
 - (b) Insumarea elementelor listei ;
 - (c) Accesul la elementul de pe poziția i din listă;
 - (d) Ștergerea unui nod din listă ;
 - (e) Copierea unei liste pe un alt suport;
4. Care din declarațiile de mai jos definesc corect o structură de tip listă liniară l , implementată prin tipul tablou?
 - (a)

```
#define lungmax 5
typedef int nod;
typedef struct{
nod elemente[lungmax];
int ultim;
}lista;
lista l;
```

- (b) `#define lung_max 5`
`struct {`
`nod elemente [lung_max];`
`int ultim;`
`}l;`
- (c) `#define lung_max 5`
`typedef int nod;`
`typedef struct {`
`nod elemente [lung_max];`
`}l;`
- (d) `#define lung_max 5`
`typedef int nod;`
`struct {`
`nod elemente [lung_max];`
`int ultim;`
`} ;`
`lista l;`

5. Se consideră descrierile în limbajul C:

```
#include <stdio.h>
#include <conio.h>
#define lung_max 5
typedef int nod;

typedef struct {
nod elemente[lung_max];
int ultim; /* indexul ultimului element al listei */

}lista;
lista l;
int k; /* k indica o pozitie din interiorul listei*/
void initializare(void) {
...
l.ultim=-1;
} /* initializare */

void ins(void) {
int i;
if(l.ultim>=lungime_max-1)
printf( "Depasire\n ");
```

```

    else {
        for(i=l.ultim;i>=k;i--)
            l.elemente[i+1]=l.elemente[i];
        printf( "Introduceti val. noului el.(nr.intreg):  ");
        scanf( "%d ",&l.elemente[k]);
        printf( "\n  ");

        l.ultim++;
    }
} /* insp */

```

și o listă l astfel implementată, cu $l.elemente=(3, 2, 5, 7,0)$, $l.ultim=3$. Care va fi imaginea listei l după apelul funcției ins ,

pentru $k=2$ și citirea de la tastatură a numărului 9 ?

- (a) $l.element = \{3, 9, 2, 5, 7\}$ $l.ultim = 4$;
- (b) $l.element = \{3, 2, 9, 5, 7\}$ $l.ultim = 4$;
- (c) $l.element = \{2, 5, 5, 7, 3\}$ $l.ultim = 4$;
- (d) $l.element = \{3, 2, 5, 7, 9\}$ $l.ultim = 4$.

6. Se consideră descrierile în limbajul C:

```

#include <stdio.h>
#include <conio.h>
#define lung_max 20
typedef int nod;

typedef struct {
    nod elemente[lung_max];
    int ultim;
}lista;
lista l;
int i;
/* i indica o pozitie din interiorul listei*/
void initializare(void) {
    l.ultim=-1;
} /* initializare */

```

Ce înțelegeți prin suprimarea nodului de pe poziția i , diferit de ultimul nod, dintr-o listă astfel implementată?

- (a) Se atribuie valoarea 0 elementului de pe poziția i ;

- (b) Se decrementează câmpul ultim, apoi se deplasează toate elementele de după poziția i spre sfârșitul tabloului;
 - (c) Se deplasează toate elementele de după poziția i cu o poziție spre începutul tabloului, apoi se decrementează valoarea câmpului ultim;
 - (d) Se deplasează toate elementele de dinaintea poziției i spre începutul tabloului, apoi se decrementează valoarea câmpului ultim.
7. O listă liniară simplu înlănțuită este:
- (a) lista liniară, în care relația de ordonare este materializată pe suport printr-un pointer către elementul următor;
 - (b) o structură de date implementată prin tipul tablou;
 - (c) o structură explicită și recursivă;
8. Se consideră descrierile în limbajul C:

```
struct nod {
    int cheie;
    char info[10];
    struct nod *urm;
};
typedef struct nod Tnod;

typedef Tnod *ref;
ref p=NULL,q;
/* p - pointerul spre primul element al listei */
```

Se dă funcția:

```
void func(void) {
    q=(ref)malloc(sizeof(Tnod));
    printf( "Introdu cheia= ");
    scanf( "%d ", &q->cheie);
    printf( "Introdu informatia= ");
    fflush(stdin);
    scanf( "%s ", q->info);
    q->urm=p;
    p=q;
}
```

Funcția de mai sus va realiza:

- (a) crearea unei liste liniare simplu înlănțuită desemnată de p cu inserare în fața listei;
- (b) crearea primului nod al listei;
- (c) crearea unei liste cu elementele în ordinea inversă a cheilor date;
- (d) inserarea unui nod în fața listei desemnată de p;
- (e) crearea unei liste cu inserare în spatele listei.

9. Se consideră descrierile în limbajul C:

```
struct nod {
    int cheie;
    char info[10];
    struct nod *urm;
};
typedef struct nod Tnod;

typedef Tnod * ref;
ref p,r;
/* p - pointerul spre inceputul listei */
int k;
```

Funcția următoare:

```
void inserare(void) {
    ref s;
    s = (ref)malloc(sizeof(Tnod));
    *s = *r; r->urm = s;
    printf( "Introdu cheia=  ");
    scanf( "%d ",&r->cheie);
    printf( "Introdu informatia=  ");
    fflush(stdin);
    scanf( "%s ", r->info);
}
```

realizează inserarea unui nou nod în fața unui nod de cheie dată-k, într-o listă liniară simplu înlănțuită, cu structura de mai sus:

- (a) chiar dacă lista este vidă;
- (b) cu excepția cazului când lista este vidă;
- (c) în fața nodului de cheie k, de adresă memorată în r, când există în listă un nod cu cheia dată;
- (d) după nodul de cheie k, de adresă memorată în r, când există în listă un nod cu cheia dată;

10. Se consideră descrierile în limbajul C:

```
struct nod {
    int cheie;
    char info[10];
    struct nod *urm;
};
typedef struct nod Tnod;

typedef Tnod * ref;
ref p,q;
/* p - pointerul spre inceputul listei */
int k;
```

Se dă funcția:

```
void cautare(void) {
    int b = 0;
    q=p;
    while ((b==0) && ( q != NULL))
    if (q->cheie ==k)
    b=1;
    else
    q=q->urm;
}
```

care realizează căutarea unui nod de cheie k într-o listă liniară simplu înlănțuită. Variabila b trebuie introdusă în această funcție pentru:

- (a) dereferențierea unui nod de adresă NULL;
 - (b) a evita dereferențierea unui nod de adresă NULL;
 - (c) a parcurge mai ușor lista;
 - (d) a putea stabili dacă nodul există sau nu în listă.
11. Fie p un pointer către primul nod al unei liste liniare simplu înlănțuită de numere întregi cu structura:

```
struct nod {
    int info;
    struct nod *urm;
};

typedef struct nod Tnod ;
```

```
typedef Tnod *ref;
ref p,r;
```

Spre al câtelea nod va pointa r în urma execuției secvenței următoare, presupunând că lista are cel puțin 7 elemente:

```
i=1;
r=p;
while(i<=5) {r=r->urm; i++;}
```

- (a) 3;
- (b) 4;
- (c) 5;
- (d) 6;
- (e) secvența este greșită

12. Fie p un pointer către primul nod al unei liste liniare simplu înlănțuită de numere întregi cu structura:

```
struct nod {
    int info;
    struct nod *urm;
};
```

```
typedef struct nod Tnod ;
```

```
typedef Tnod *ref;
ref p;
/* p - pointerul spre primul element al listei */
```

iar lista conține în ordine numerele: 3, 4, 5, 6. Ce valoare va returna apelul funcție(p) aceasta având următoarea definiție:

```
int functie (ref p) {
    int s=0;
    ref r;
    r=p;
    while(r->urm) {
        if(!(r->info%2))
            s+=r->info;
        r=r->urm;
    } return s;
}
```

- (a) 3;
- (b) 4;
- (c) 8;
- (d) 10;
- (e) 18.

13. Fie p un pointer către primul nod al unei liste liniare simplu (cu minim 3 elemente) înlănțuită de numere întregi cu structura:

```
struct nod {
    int info;
    struct nod *urm;
};

typedef struct nod Tnod ;

typedef Tnod *ref;
ref p;
```

Care din următoarele instrucțiuni afisează numărul conținut în al treilea nod din listă?

- (a) `printf("%d",p->urm->urm->info);`
 - (b) `printf("%d",p->urm->info);`
 - (c) `printf("%d",p->urm->info->info);`
 - (d) `printf("%d",p->urm->urm->urm);`
 - (e) `printf("%d",p->urm->urm->urm->info);`
14. Fie p un pointer către primul nod al unei liste liniare simplu înlănțuită de numere întregi cu structura:

```
struct nod {
    int info;
    struct nod *urm;
};

typedef struct nod Tnod ;

typedef Tnod *ref;
ref p;
```

Care este acțiunea apelului funcției următoare? void

```

func(void) {
    ref q;
    q=(ref)malloc(sizeof(Tnod));
    printf("Introduceti informatia:");
    scanf("%d",&(q->info)); q->urm=p;
    p=q;
}

```

- (a) crează un nou nod și îl inserează la sfârșitul listei;
 - (b) crează un nou nod și îl inserează la începutul listei;
 - (c) modifică conținutul primului nod al listei, adresa rămânând neschimbată;
 - (d) modifică adresa primului nod, conținutul rămâne neschimbat;
 - (e) nici una din acțiunile de mai sus.
15. Fie p un pointer către primul nod al unei liste liniare simplu înlanțuită de numere întregi cu structura:

```

struct nod {
    int info;
    struct nod *urm;
};

typedef struct nod Tnod ;

typedef Tnod *ref;
ref p,r,s;
/* p indica primul nod al listei, r indica un nod oarecare al listei*/

```

Care este acțiunea apelului funcției operatie?

```

void operatie(void)
{
    if (r->urm==NULL)
if (p==r) {
    p=NULL;
    free(r);
    } else {
    s=p;
    while (s->urm!=r) s=s->urm;
    s->urm=0;
    free(r);
    }
}

```

```
    else {  
        s=r->urm;  
        *r=*s;  
        free(s);  
    } }
```

- (a) suprimarea nodului de după nodul precizat de r;
 - (b) suprimarea nodului precizat de r;
 - (c) suprimarea unui nod diferit de ultimul nod al listei.
16. Fie p un pointer către primul nod al unei liste liniare simplu înlănțuită de numere întregi, q un pointer către ultimul nod al aceleiași liste, listă care are structura:

```
struct nod {  
    int info;  
    struct nod *urm;  
};  
  
typedef struct nod Tnod ;  
  
typedef Tnod *ref;  
ref p,q;
```

Care este efectul apelului funcției următoare?

```
void func(void)  
{  
    ref r;  
    r=(ref)malloc(sizeof(Tnod));  
    printf("Introduceti informatia:");  
    scanf("%d",&(r->info));  
    r->urm=NULL;  
    q->urm=r;  
    q=r;  
}
```

- (a) crează un nou nod și îl inserează la sfârșitul listei desemnată de p, nevidă, care are ultimul nod de adresă q;
- (b) crează un nou nod și îl inserează la începutul listei;
- (c) modifică conținutul ultimului nod al listei, adresa rămânând neschimbată;
- (d) modifică adresa ultimului nod, conținutul rămâne neschimbat;
- (e) nici una din acțiunile de mai sus.

17. Fie q un pointer către ultimul nod al unei liste liniare simplu înlănțuită de numere întregi cu structura:

```
struct nod {
    int info;
    struct nod *urm;
}

typedef struct nod Tnod ;

typedef Tnod *ref;
ref q;
```

Care este acțiunea funcției următoare?

```
void func(void)
{
    ref r;
    r=(ref)malloc(sizeof(Tnod));
    printf("Introduceti informatia:");
    scanf("%d",&(r->info)); r->urm=NULL;
    if(q!=NULL)
        q->urm=r;
    q=r;
}
```

- (a) crează un nou nod și îl inserează la sfârșitul listei;
 - (b) crează un nou nod și îl inserează la începutul listei;
 - (c) modifică conținutul ultimului nod al listei, adresa rămânând neschimbată;
 - (d) modifică adresa ultimului nod, conținutul rămâne neschimbată;
 - (e) nici una din acțiunile de mai sus.
18. Fie p un pointer către primul nod al unei liste liniare simplu înlănțuită de numere întregi cu structura:

```
struct nod {
    int info;
    struct nod *urm;
};

typedef struct nod Tnod ;
```



```
typedef Tnod *ref;  
ref p;
```

Care este acțiunea funcției următoare?

```
ref func(int k) {  
    ref r;  
    int b=0;  
    r=p;  
    while((b==0)&&(r!=NULL)) {  
        if(r->info==k)  
            b=1;  
        else r=r->urm;  
    }  
    return r;  
}
```

- (a) caută primul nod al listei și returnează valoarea sa;
- (b) caută nodul care conține valoarea k în câmpul info și returnează adresa sa;
- (c) caută nodul următor nodului ce conține valoarea k în câmpul info;
- (d) caută nodul de pe poziția k și returnează adresa sa;
- (e) nici una din acțiunile de mai sus.

19. Se consideră descrierile în limbajul C:

```
struct nod {  
    int cheie;  
    char info[10];  
    struct nod *urm;  
};  
typedef struct nod Tnod;  
  
typedef Tnod * ref;  
ref p; /* p - pointerul spre ^{\i}nceputul listei */
```

Descrierea anterioară se poate utiliza pentru a defini o structură de date de tip

- (a) listă liniară dublu înlanțuită cu elemente care au cheia de tip întreg;
- (b) listă circulară simplu înlanțuită cu elemente care au cheia de tip întreg;
- (c) listă circulară dublu înlanțuită cu elemente care au cheia de tip întreg;

- (d) arbore binar cu elemente care au cheia de tip întreg;
- (e) listă liniară simplu înlănțuită cu elemente care au cheia de tip întreg.

20. Se consideră descrierile în limbajul C:

```
struct nod {
    int info;
    struct nod *urm;
};

typedef struct nod Tnod ;

typedef Tnod *ref;
ref p,r,s;
/* p indica primul nod al listei, r indica predecesorul nodului
   care va fi suprimat, s o variabila auxiliara*/

void suprimare(void) {
    if (r->urm==NULL)
        printf( "Eroare: nodul de suprimat nu exista.\n ");
    else {
        s=r->urm;
        ...
    } }
}
```

Care din secvențele următoare pot completa descrierea corectă a funcției care va realiza suprimarea succesivului nodului precizat de r?

- (a) `r->urm=s; free(s);`
- (b) `r->urm=s->urm; free(s);`
- (c) `r->urm=s->urm; free(r);`

21. O lista liniară dublu înlănțuită, cu elemente întregi, este descrisă în C, astfel:

```
(a) struct nod {
    int element;
    struct nod *urm, *ant;
};
typedef struct nod Tnod;
typedef Tnod *ref;
ref p;
```

- (b)

```
struct nod {
    int element;
    struct nod urm, ant;
};
typedef struct nod *Tnod;
typedef Tnod ref;
ref p;
```
- (c)

```
typedef struct nod {
    int element;
    struct nod *urm, *pred;
}Tnod;
typedef Tnod *ref;
ref p;
```
- (d)

```
struct nod {
    int element;
    struct nod *urm, *pred;
};
typedef struct nod Tnod;
typedef Tnod ref;
ref p;
```

22. Se consideră descrierea în C, a unei structuri de tip listă liniară dublu înlănțuită :

```
struct nod {
    int info;
    struct nod *ant,*urm;
};
typedef struct nod Tnod;
typedef Tnod *ref;
ref p;
```

Care din afirmațiile următoare sunt greșite:

- (a) info este câmpul de informație utilă (un număr întreg);
(b) existența celor doi pointeri permite parcurgerea listei în ambele sensuri;
(c) o listă dublu înlănțuită permite inserarea unui nou element doar la unul din capetele listei;
(d) toate afirmațiile de mai sus sunt adevărate.

23. Se consideră descrierile în limbajul C:

```
struct nod {
    int element;
```

```
struct nod *urm, *ant;
};
typedef struct nod Tnod;
typedef Tnod *ref;
ref p,r;
```

Funcția următoare:

```
void inserare(void) {
    ref pred, s;
    pred = r->ant;
    s=(ref)malloc(sizeof(Tnod));
    printf( "Introdu elementul=  ");
    scanf( "%d ", &s->element);
    s->ant = pred;
    s->urm = r;
    pred->urm = s;
    r->ant = s;
}
```

realizează:

- (a) inserarea unui nou nod înaintea nodului de adresă **r**;
- (b) inserarea unui nou nod după nodul de adresă **r**;
- (c) inserarea unui nou nod înaintea nodului de adresă **r**, cu **r->ant!=0**;
- (d) inserarea unui nou nod înaintea nodului ***r**;

24. Se consideră descrierile în limbajul C:

```
struct nod {
    int element;
    struct nod *urm, *ant;
};
typedef struct nod Tnod;
typedef Tnod *ref;
ref p,r;
/* p desemneaza primul nod al listei, iar r nodul care trebuie sters*/
void sterg_nod(void) {
    ref pred,suc;
    pred=r->ant;
    suc=r->urm;
    ...
}
```

```

if (r==p) / * daca nodul de suprimat este primul nod */
p=p->urm;
free(r);
}

```

Cu care din secvențele următoare trebuie completată funcția `sterg_nod` astfel încât să realizeze suprimarea unui nod de adresă `r`, dintr-o listă liniară dublu înlănțuită cu structura nodurilor definită mai sus:

- (a) `if (r->urm!=NULL) suc->ant=pred;`
`if (r->ant!=NULL) pred->urm=suc;`
- (b) `if (r->ant!=NULL) pred->urm=suc;`
`if (r->urm!=NULL) suc->urm=pred;`
- (c) `if (r->urm!=NULL) pred->urm=suc;`
`if (r->ant!=NULL) suc->ant=pred;`

25. O stivă este:

- (a) listă liniară în care inserările se fac la un capăt al listei și suprimările la celalalt capăt al listei;
- (b) o listă liniară de tipul LIFO (Last In First Out);
- (c) o listă liniară cu restricție la intrare;
- (d) o listă liniară în care se manipulează mereu elementul cel mai recent introdus.

26. Se considera o stivă implementată prin tipul tablou:

```

#define lung_max 100
typedef int nod;
typedef struct {
    int varf;
} nod elemente [lung_max];
} stiva;
stiva s;

```

Care din următoarele funcții realizează inițializarea stivei `s` (inițializarea constă în a face stiva vidă), stiva crescând, în ordinea descreșterii indexului în tablou.

- (a) `void initializare (stiva s) {`
`s->varf = lung_max; }`
- (b) `void initializare(stiva *s) {`
`s->varf = NULL;`
`}`

- (c) `void initializare(stiva *s) {
 s->varf = lung_max;
}`
- (d) `void initializare () {
 s.varf = lung_max;
}`

27. Se consideră descrierile în limbajul C:

```
typedef int tipelem;  
typedef struct nod {  
    tipelem elem;  
} Tnod;  
  
struct nod *urm;  
} Tnod;  
  
typedef Tnod *ref;  
ref varf; /* virful stivei */  
ref r; /* variabila auxiliara */  
void adauga(void) {  
    r=malloc(sizeof(Tnod));  
    printf( "Introduceti informatia(nr.intreg,max 5 cifre): ");  
    scanf( "%d ",&r->elem);  
    ...  
}
```

Care din secvențele următoare completează corect funcția `adauga` astfel încât ea să realizeze adăugarea unui element în stiva `s`?

- (a) `r.urm=varf;`
`varf=r;`
- (b) `r->urm=varf;`
`varf=r;`
- (c) `r->urm=varf;`
28. O structură de date de tip coadă este:
- (a) structură de tip listă cu restricție la intrare;
- (b) o listă liniară de tipul FIFO (First In First Out);
- (c) o lista liniară în care toate operațiile se efectuează doar la unul din capetele listei;
- (d) o listă liniară în care inserările se efectuează la un capăt al listei, iar suprimările și ori ce alt acces se efectuează la celălalt capăt al listei.

29. Se consideră descrierile în limbajul C:

```
typedef int tipelement;

typedef struct nod {
tipelement element;
struct nod *urm;
}Tnod;

typedef Tnod *ref;

typedef struct {
ref fata,spate;
}Coadă;
Coadă c;
void initializare(Coadă* c) { /* creeaza nodul fictiv*/
c->fata=(ref)malloc(sizeof(Tnod));
c->fata->urm=0;
/* nodul fictiv este primul si ultimul nod*/
c->spate=c->fata;
} /* initializare */
```

Pentru inițializarea cozii c, cu structura de mai sus, cu element fictiv cap de listă, se va apela funcția inițializare astfel:

- (a) initializare(c);
- (b) initializare(&c);
- (c) initializare(* c).

30. Se consideră descrierile în limbajul C:

```
typedef int tipelement;
typedef struct nod {
tipelement element;
struct nod *urm;
} Tnod;

typedef Tnod *ref;

typedef struct {
ref fata,spate;
} Coadă;
```

```
Coadă c;  
void adauga(tipelement x, Coadă * c) {  
c->spate->urm=malloc(sizeof(Tnod));  
...  
c->spate->element=x;  
c->spate->urm=NULL;  
} / * adauga */
```

Cu care din secvențele următoare trebuie completată funcția `adauga` pentru a realiza adăugarea unui element de cheie dată x într-o coadă implementată prin structura de mai sus, cu element cap de listă.

- (a) `c->fata= c->spate->urm;`
- (b) `c->spate=c->spate->urm;`
- (c) `c->spate->urm=NULL;`

4 Teoria grafurilor și combinatorică

1. Ce rang are permutarea $\langle 6, 1, 3, 2, 5, 4 \rangle$ în enumerarea lexicografică a permutărilor?
 - (a) 411
 - (b) 412
 - (c) 605
 - (d) 607
2. Câte permutări ale literelor ABCDEFGH conțin subșirul BCD?
 - (a) $6!$
 - (b) $8!$
 - (c) $8!/3$
 - (d) 2^5
3. Câte șiruri se pot obține rearanjând literele șirului SUCCESS?
 - (a) $7!$
 - (b) 2^7
 - (c) 420
 - (d) 12
4. Câte numere întregi cuprinse între 1 și 1000 se divid cu 7, dar nu se divid cu 3?
 - (a) 93
 - (b) 95
 - (c) 92
 - (d) 136
5. Într-un sertar sunt opt ciorapi maro și 12 ciorapi negri. Un copil scoate la întâmplare pe întuneric ciorapi din sertar. Câți ciorapi trebuie să scoată copilul pentru a fi sigur că a scos cel puțin 2 ciorapi negri?
 - (a) 3
 - (b) 10
 - (c) 14
 - (d) 95
6. Ce coeficient are $x^8 y^2$ în $(x + 2 \cdot y)^{10}$?
 - (a) 1

- (b) 45
- (c) 180
- (d) 5120
- (e) 960

7. Câte submulțimi cu mai mult de 2 elemente are o mulțime cu 6 elemente?

- (a) 32
- (b) 61
- (c) 42
- (d) 21

8. Care este forma generală a relației de recurență

$$a_n = -3a_{n-1} - 3a_{n-2} - a_{n-3}$$

cu condițiile inițiale $a_0 = 1, a_1 = -2$ și $a_2 = -1$?

- (a) $a_n = 2n^2 - 5n + 1$
- (b) $a_n = -\frac{1}{2}n^3 + \frac{7}{2}n^2 - 6n + 1$
- (c) $a_n = -4 \cdot (-2)^n + (5n + 5)(-1)^n$
- (d) $a_n = (1 + 3n - 2n^2) \cdot (-1)^n$

9. Un mesaj transmis pe un canal de comunicare este o secvență de 2 tipuri de semnale: semnale de tip A care durează 1 microsecundă, și 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

10. În internet, format din rețele interconectate de calculatoare, fiecărei conexiuni de rețea dintr-un calculator i se atribuie o adresă internet. Protocolul IPv4 prevede că o adresă internet este un șir de 32 biți, format din un număr de rețea (*netid*) urmat de un număr de gazdă (*hostid*). Sunt 3 tipuri de adrese internet:

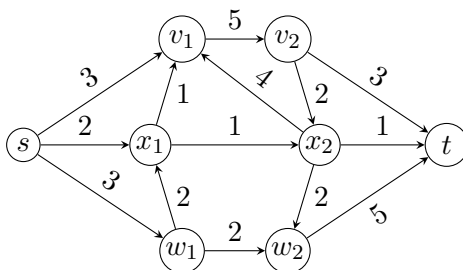
- (a) Clasa A: acestea sunt de forma $0 \underbrace{b_1 b_2 b_3 b_4 \dots b_7}_{\text{netid}} \underbrace{b_8 b_9 \dots b_{30} b_{31}}_{\text{hostid}}$
- (b) Clasa B: acestea sunt de forma $10 \underbrace{b_2 b_3 b_4 \dots b_{15}}_{\text{netid}} \underbrace{b_{16} b_{17} \dots b_{30} b_{31}}_{\text{hostid}}$

(c) Clasa C: acestea sunt de forma $110 \underbrace{b_3 b_4 b_5 \dots b_{23}}_{\text{netid}} \underbrace{b_{24} b_{25} \dots b_{30} b_{31}}_{\text{hostid}}$

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

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

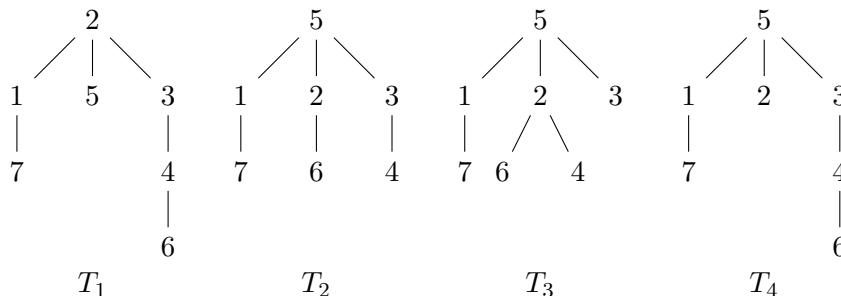
11. Fie rețeaua de transport G cu sursa s și destinația t :



Care este valoarea fluxului maxim în G ?

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

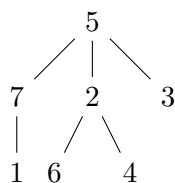
12. Care din arborii următori are secvența Prüfer 5,4,3,5,1?



- (a) T_1
- (b) T_2

- (c) T_3
- (d) T_4

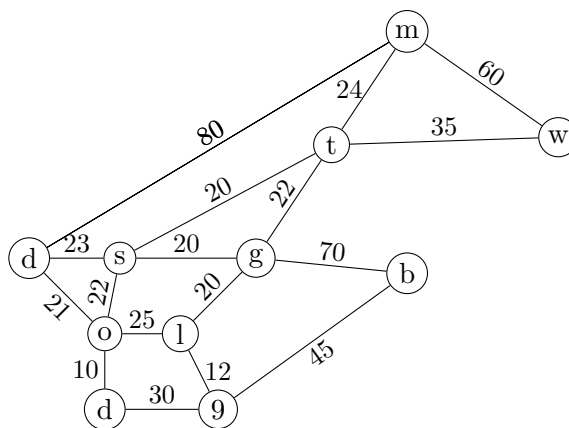
13. Care este secvența Prüfer a arborelui următor:



T_3

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

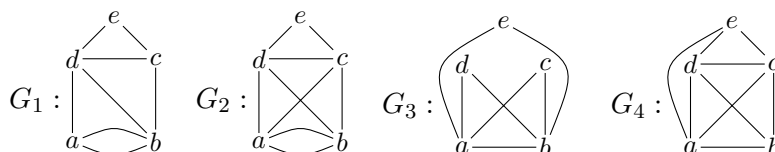
14. Fie graful ponderat



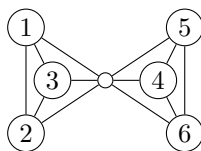
Ce greutate totală are arborele minim de acoperire al acestui graf?

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

15. Care este semnificația lui $\binom{n}{k}$?
- Numărul de submulțimi cu k elemente al unei mulțimi cu n elemente.
 - Numărul de k -permutări al unei mulțimi cu n elemente.
 - Numărul de feluri în care se poate partiționa o mulțime cu n elemente în exact k submulțimi nevide și disjuncte.
 - Numărul de feluri în care pot fi puse n persoane la k mese rotunde identice, astfel încât nici o masă rotundă să nu rămână neocupată.
16. Care din grafurile următoare este eulerian?

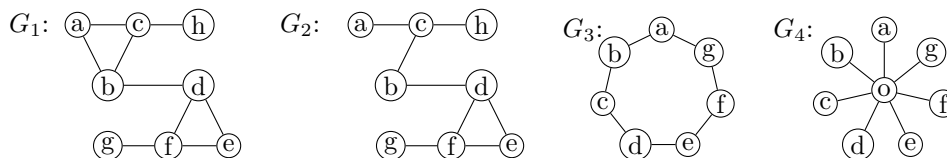


- G_1, G_4
 - G_2, G_3
 - nici unul
 - G_4
 - G_1, G_3
17. Fie configurația următoare:



- Câte colorări diferite cu 3 culori are această configurație?
- 24
 - 1120
 - 120
 - 64
 - 216

18. Care din grafurile următoare are un cuplaj perfect?



- (a) G_1
(b) G_2
(c) G_3
(d) G_4
(e) 216
19. Care din familiile următoare de mulțimi are un sistem de reprezentanți distincți:
- (a) $\{1\}, \{1, 2\}, \{2\}, \{3, 4, 5\}, \{1, 2, 3, 4, 5\}$
(b) $\{1, 2\}, \{2, 3\}, \{1, 2, 3\}, \{1, 3\}, \{1, 2, 3, 4\}, \{1, 2, 4, 5\}$
(c) $\{1, 2, 3\}, \{2, 3, 4\}, \{3, 4, 5\}, \{4, 5\}, \{1, 2, 5\}$
(d) $\{1, 2, 5\}, \{1, 5\}, \{1, 2\}, \{2, 5\}, \{1, 3\}, \{6\}, \{4, 6, 7\}$
20. Câți arbori diferiți cu 5 noduri, numerotate de la 1 la 5, există?
- (a) 10
(b) 273
(c) 32
(d) 120
(e) 125
21. Fie G un graf cu n noduri. Să se indice toate afirmațiile care sunt echivalente cu faptul că G este un arbore:
1. G este un graf conectat fără cicluri.
 2. G are $n - 1$ muchii.
 3. G nu are cicluri, iar dacă se adaugă o muchie la G , se crează exact un ciclu.
 4. Fiecare pereche de noduri din G este conectată cu exact o cale simplă.
- (a) 1, 2, 3, 4
(b) 1, 2, 3
(c) 1, 3, 4
(d) 1

5 Limbaje formale și teoria automatelor

1. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$; $V_T = \{0, 1, 2\}$; $P = \{S \rightarrow 0S0|1S1|2S2|0\}$, este:
 - (a) $L = \{0^n 1^n 2^n | n \geq 1\}$;
 - (b) $L = \{w \in \{0, 1, 2\}^* | w \text{ palindrom centrat în } 0\}$;
 - (c) $L = \{0^n 1^m 2^p | n, m, p \in N\}$;
 - (d) $L = \{w0x | w, x \in \{0, 1, 2\}^*, x \text{ oglinditul lui } w\}$;
2. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$; $V_T = \{a, b\}$; $P = \{S \rightarrow aSb|ab\}$, este:
 - (a) $L = \{a^n b^m | n, m \geq 1\}$;
 - (b) $L = \{w \in \{a, b\}^* | w \text{ conține infixul } ab\}$;
 - (c) $L = \{a^n b^n | n \geq 0\}$;
 - (d) $L = \{a^n b^n | n \geq 1\}$;
3. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$; $V_T = \{a, b\}$; $P = \{S \rightarrow bSb|bAb, A \rightarrow aA|aa\}$, este:
 - (a) $L = \{b^n aab^n | n \geq 1\}$;
 - (b) $L = \{b^n aaa^m b^n | n \geq 1, m \geq 0\}$;
 - (c) $L = \{b^n (aa)^m b^n | n, m \geq 1\}$;
 - (d) $L = \{b^n a^m b^n | n \geq 1, m \geq 2\}$;
4. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$; $V_T = \{a, b\}$; $P = \{S \rightarrow aSb|aAb, A \rightarrow aA|\lambda\}$, 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) $L = \{a^{n+i} b^n | n \geq 1, i \geq 0\}$;
5. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S, A, B\}$, $V_T = \{a, b, c\}$, $P = \{S \rightarrow abc|aAbc, Ab \rightarrow bA, Ac \rightarrow Bbcc, bB \rightarrow Bb, aB \rightarrow aaA|aa\}$, este:
 - (a) $L = \{a^n b^m c^p | n, m, p \geq 1\}$;
 - (b) $L = \{w \in \{a, b, c\}^* | w \text{ conține cel puțin trei litere }\}$;
 - (c) $L = \{w \in \{a, b, c\}^* | w \text{ palindrom centrat în } abc\}$;
 - (d) $L = \{a^n b^n c^n | n \geq 1\}$;

6. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S, A\}$, $V_T = \{a, b, c\}$, $P = \{S \rightarrow A|aS|bS|cS, A \rightarrow Aa|Ab|Ac|\lambda\}$, este :
- $L = \{a^n b^m c^p | n, m, p \geq 0\}$;
 - $L = \{w \in \{a, b, c\}^* | w \text{ conține cel puțin o literă } \}$;
 - $L = \{a, b, c\}^*$;
 - $L = \{a^n b^n c^n | n \geq 1\}$;
7. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S, A, B, C, D, E, F\}$, $V_T = \{a, b, \dots, z\}$, $P = \{S \rightarrow iA|aB, A \rightarrow oC, B \rightarrow nD, C \rightarrow n, D \rightarrow a\}$, este :
- $L = \{ion, ana\}$;
 - $L = \{ion, ani, ina, aia, iao\}$;
 - $L = \{w \in \{a, b, \dots, z\}^* | w \text{ începe cu a sau i și se termină cu a sau n } \}$;
8. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$, $V_T = \{a, b\}$, $P = \{S \rightarrow aS|bS|a\}$, este :
- $L = \{a^n a, b^n a | n \geq 0\}$;
 - $L = \{wa | w \in \{a, b\}^*\}$;
 - $L = \{w \in \{a, b\}^* | w \text{ se termină cu a } \}$;
 - $L = \{w \in \{a, b\}^* | w \text{ începe cu a } \}$;
9. Limbajul generat de gramatica $G = (V_N, V_T, S, P)$, unde $V_N = \{S, X\}$, $V_T = \{a, b, c\}$, $P = \{S \rightarrow aS|bS|cS|abbaX, X \rightarrow aX|bX|cX|\lambda\}$, este:
- $L = \{w \in \{a, b, c\}^* | w \text{ conține abba } \}$;
 - $L = \{wabbaw | w \in \{a, b, c\}^*\}$;
 - $L = \{w \in \{a, b, c\}^* | w \text{ se termină cu abba } \}$;
 - $L = \{wabbax | w, x \in \{a, b, c\}^*\}$;
 - $L = \{w \in \{a, b, c\}^* | w \text{ începe cu abba } \}$;
10. Regulile gramaticii $G = (V_N, V_T, S, P)$, unde $V_N = \{S\}$, $V_T = \{PCR, PDAR, UDMR\}$, $P = \{S \rightarrow PCR|PDAR|UDMR\}$, respectă restricțiile impuse gramaticilor:
- regulate (tip 3);
 - independente de context (tip 2);
 - dependente de context (tip 1);
 - de tipul 0, 1, 2, 3;
11. Regulile gramaticii $G = (V_N, V_T, S, P)$, unde $V_N = \{S, X\}$, $V_T = \{a, b, c\}$, $P = \{S \rightarrow aS|bS|cS|abbaX, X \rightarrow aX|bX|cX|\lambda\}$, respectă restricțiile 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;
12. Regulile gramaticii $G = (V_N, V_T, S, P)$, unde $P = \{S \rightarrow abc|aAbc, Ab \rightarrow bA, Ac \rightarrow Bbcc, bB \rightarrow Bb, aB \rightarrow aaA|aa\}$, respectă restricțiile impuse gramaticilor:
- (a) regulate (tip 3);
 (b) independente de context (tip 2);
 (c) dependente de context (tip 1);
 (d) de tipul 0;
13. Regulile gramaticii $G = (V_N, V_T, S, P)$, unde $P = \{S \rightarrow aSb|\lambda\}$, respectă restricțiile impuse gramaticilor:
- (a) regulate (tip 3);
 (b) independente de context (tip 2);
 (c) dependente de context (tip 1);
 (d) de tipul 0;
14. Regulile gramaticii $G = (V_N, V_T, S, P)$, unde $P = \{S \rightarrow aS|bS|\lambda\}$, respectă restricțiile impuse gramaticilor:
- (a) regulate (tip 3);
 (b) independente de context (tip 2);
 (c) dependente de context (tip 1);
 (d) de tipul 0;
15. Expresia regulată ce notează limbaajul $L = \{w \mid \text{șiruri de 0 și 1 terminate cu 1}\}$, este:
- (a) $(0|1)^*1$;
 (b) $1|(0|1)^*$;;
 (c) $(0^*|1^*)^*1$;
 (d) $(1|0|1)^*1$;
16. Expresia regulată ce notează limbaajul $L = \{w \mid \text{șiruri de 0 și 1 ce conțin cel puțin un simbol 1}\}$, este:
- (a) $(0|1)^*1$;
 (b) $1|(0|1)^*1(1|0)^*$;

- (c) 0^*11^* ;
 (d) $(0|1)^*1(0|1)^*$;
17. Expresia regulată ce notează limbaajul $L = \{w \mid \text{șiruri de 0 și 1 ce conțin cel puțin un simbol}\}$, este:
- (a) $(0|1)^*1$;
 (b) $1|0|(0|1)^*$;
 (c) $(0^*|1^*)^*|1|0$;
 (d) $(1|0)(0|1)^*$;
18. Expresia regulată ce notează limbaajul $L = \{ana, ani, ina, ini\}$, este:
- (a) $ana|ani|ina|ini$;
 (b) $(a|i)n(a|i)$;
 (c) a^*ni^* ;
 (d) $(a|i)^*n(a|i)^*$;
19. Expresiile regulate notează:
- (a) limbajele regulate;
 (b) limbajele de tipul 0 și 3;
 (c) limbajele recunoscute de automate finite deterministe;
 (d) numai limbajele finite;
20. Limbaajul L notat de expresia regulata $(a|i)n(a|i)$ este:
- (a) $L = \{ana, ani, ina, ini\}$;
 (b) $L = \{w \in \{a, n, i\}^* \mid w \text{ începe și se termină cu a sau i}\}$;
 (c) $L = \{w \in \{a, n, i\}^* \mid w \text{ începe cu a sau i și se termină cu na sau ni}\}$;
 (d) $L = \{w \in \{a, n, i\}^* \mid w \text{ începe și se termină cu a sau i, iar litera n este în mijlocul cuvântului } w \}$;
21. Limbaajul L notat de expresia regulata $(0|1)(0|1)^*$ este:
- (a) $L = \{0, 1, 00, 01, 10, 11, 000, \dots\}$;
 (b) $L = \{w \in \{0, 1\}^* \mid w \text{ începe cu 0 sau 1}\}$;
 (c) $L = \{0, 1\}^*$;
22. Limbaajul L notat de expresia regulata $01^*|1$; este:
- (a) $L = \{0, 1, 00, 01, 10, 11, 000, \dots\}$;

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

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

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

23. Limbajul L notat de expresia regulata $(11)^*1$ este:

(a) $L = \{1, 11, 111, 1111, \dots\}$;

(b) $L = \{w \in \{0, 1\}^* | w \text{ are un număr impar de simboluri } 1\}$;

(c) $L = \{1w | w = 1^{2n}, n \in N\}$;

(d) $L = \{1^{2n+1}, n \in N\}$;

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

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

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

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

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

6 Răspunsuri

Algoritmica

- | | |
|-----------------|-----------------|
| 1. 1c | 16. 16b,16c |
| 2. 2d | 17. 17a,17b |
| 3. 3d | 18. 18c |
| 4. 4c,4d | 19. 19d |
| 5. 5a,5c | 20. 20c |
| 6. 6c | 21. 21b |
| 7. 7c | 22. 22b,22c |
| 8. 8b | 23. 23b |
| 9. 9a,9d | 24. 24b,24d |
| 10. 10d | 25. 25a,25b,25d |
| 11. 11b,11e | 26. 26b,26e |
| 12. 12d | 27. 27d,27f |
| 13. 13b | 28. 28d |
| 14. 14c,14d,14e | 29. 29b,29d,29f |
| 15. 15b | 30. 30b |

Bazele informaticii

- | | |
|----------------|---------------------|
| 1. 1a,1b,1c | 16. 16a |
| 2. 2a,2b | 17. 17a |
| 3. 3a,3b,3c | 18. 18a |
| 4. 4a,4b,4c,4d | 19. 19a |
| 5. 5b | 20. 20a |
| 6. 6a | 21. 21a |
| 7. 7a | 22. 22a |
| 8. 8a,8b | 23. 23b |
| 9. 9a | 24. 24b |
| 10. 10a | 25. 25a |
| 11. 11a | 26. 26a |
| 12. 12b | 27. 27a |
| 13. 13b | 28. 28a,28b,28c,28d |
| 14. 14b | 29. 29a |
| 15. 15c | 30. 30b |

Structuri de date

- | | |
|----------------|-------------|
| 1. 1b,1c,1e | 16. 16a |
| 2. 2b | 17. 17a |
| 3. 3a,3c,3d,3e | 18. 18b |
| 4. 4a | 19. 19b,19e |
| 5. 5b | 20. 20b |
| 6. 6c | 21. 21a,21c |
| 7. 7a,7c | 22. 22c,22d |
| 8. 8a,8b,8d | 23. 23c |
| 9. 9c | 24. 24a |
| 10. 10b | 25. 25b,25d |
| 11. 11d | 26. 26d |
| 12. 12b | 27. 27b |
| 13. 13a | 28. 28b,28d |
| 14. 14b | 29. 29b |
| 15. 15b | 30. 30b |

Teoria grafurilor și combinatorică

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

Limbaje formale și teoria automatelor

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