Universitatea Tehnică "Gheorghe Asachi" din Iași
Facultatea de Automatică și Calculatoare
Admitere – sesiunea iulie 2020
Domeniile: Calculatoare și tehnologia informației
Ingineria sistemelor (Automatică și informatică aplicată)

*

# Subiecte la testul grilă de Informatică

1. Fie programul de mai jos:

```
void Print(int i, int limit)
{
    do
    {
        if (i++ < limit)
        {
            cout<<"Queen";
            continue;
        }
    }while (0);
}
int main() {
    Print(1, 4);
    return 0;
}
```

De câte ori va fi afișat pe ecran cuvântul Queen?

(a) 3 (b) 1 (c) 4 (d) 0

2. Fie următoarea secvență de cod:

```
if (test)
    cout << "Da" << endl;
else
    cout << "Nu" << endl;
```

Care trebuie să fie forma expresiei *test*, din secvența de cod dată, pentru ca, pe monitor să se afișeze *Da* numai dacă valoarea variabilei $x$ este pară și strict pozitivă?

(a) (x % 2 == 0) && (x <= 0)

(b) (x % 2 == 0) || (x > 0)

(c) !((x % 2 != 0) || (x <= 0))

(d) !((x % 2 != 0) && (x <= 0))

3. Se consideră următoarele funcții, cu A și B numere naturale:

```
int f1(int A, int B)
{
    if (!B)
        return A;
    return f1(B, A%B);
}
```

```
int f2(int A, int B)
{
    while (A != B)
    {
        if (A > B)
            A = A - B;
        if (B > A)
            B = B - A;
    }
    return A;
}
```

Care dintre cele două funcții implementează algoritmul lui Euclid?

(a) Numai $f1$ (b) Numai $f2$ (c) $f1$ și $f2$ (d) Nici $f1$ nici $f2$

4. Ce se va afișa în urma apelului $f(6)$ a funcției de mai jos, dacă $a$ este un vector cu $n$ elemente ce conține valorile 1,2,3,4,5,6,7,8,9,10, date în această ordine? Indicele primului element al vectorului este 1.

**funcția** f($n$)
- **pentru** $i \leftarrow 1, n/2$ **execută**
  - $aux \leftarrow a[i]$
  - $a[i] \leftarrow a[n-i]$
  - $a[n-i] \leftarrow aux$
- **pentru** $i \leftarrow 1, n$ **execută**
  - afișează $a[i]$

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

5. Care este rezultatul apelului funcției $f$ de mai jos, dacă tabelul bidimensional transmis ca parametru este matricea din figură?

```
int f(int a[7][7]) {
    int i, j, c = 0;
    for (i = 1; i<6; ++i)
        for (j = 1; j<6; ++j)
        {
            int x = (a[i][j]+1)%3;
            int y = (3+a[i][j]-1)%3;
            if (x == a[i][j+1] && x == a[i][j-1] &&
                    y == a[i+1][j] && y == a[i-1][j])
                c = c + 1;
        }
    return c;
}
```

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 0 | 2 | 0 |
| 2 | 2 | 1 | 2 | 1 | 2 | 2 |
| 0 | 1 | 0 | 0 | 0 | 2 | 2 |
| 2 | 2 | 0 | 1 | 1 | 1 | 2 |
| 1 | 0 | 1 | 0 | 0 | 2 | 0 |
| 2 | 2 | 2 | 1 | 2 | 1 | 2 |
| 1 | 0 | 1 | 0 | 1 | 2 | 1 |

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

6. Care este valoarea întoarsă de funcție și ce se afișează după apelul functie(10) ?

```
int functie(int n)
{
    if (n < 1)
        cout << n << endl;
    else {
        cout << n << " ";
```

```
        return n + functie(n-1);
        cout << n << " ";
    }
    return 1;
}
```

(a) Valoarea întoarsă de funcție este 112 și se afișează:
1 2 3 4 5 6 7 8 9 10
10 9 8 7 6 5 4 3 2 1 0
(b) Valoarea întoarsă de funcție este 110 și se afișează:
10 9 8 7 6 5 4 3 2 1 0
1 2 3 4 5 6 7 8 9 10
(c) Valoarea întoarsă de funcție este 55 și se afișează:
10 9 8 7 6 5 4 3 2 1
(d) Valoarea întoarsă de funcție este 56 și se afișează:
10 9 8 7 6 5 4 3 2 1 0

7. Următoarea instrucțiune afișează pe ecran:

```
int a[4] = {1};
for (int k = 0; k <= 3; k++)
    cout << a[k] << " ";
```

(a) 0 0 0 0 (b) 1 0 0 0 (c) 1 1 1 1 (d) eroare la compilare

8. Se dă un labirint sub forma unei matrice pătratice cu $n$ linii și $n$ coloane, în care locațiile accesibile sunt notate cu **0** iar locațiile inaccesibile (zidurile labirintului) cu **1**. Coordonatele locațiilor la care se află recompensele sunt $(3,3)$, $(1,4)$ și $(3,1)$. Dându-se harta labirintului de mai jos, să se determine lungimea minimă a unui drum care pleacă din poziția $(1,1)$, trece obligatoriu prin locațiile la care se află recompensele (nu contează în ce ordine) și apoi ajunge în poziția $(n,n)$. Un pas din drum constă din deplasarea dintr-un punct $(i,j)$ într-unul din cele patru învecinate pe linie și coloană, adică într-unul din punctele $(i-1,j)$, $(i,j-1)$, $(i+1,j)$, $(i,j+1)$. Deplasarea se face doar prin locațiile accesibile (notate cu 0).

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 |

(a) 15 (b) 11 (c) 10 (d) 12

9. Se dă o matrice cu $n$ linii și $n$ coloane care conține inițial numai valori de 0. Se definesc trei tipuri de operații care pot fi executate asupra acesteia: **ADD(X,Y,Z)** - la valoarea existentă pe poziția (X,Y) în matrice adună valoarea Z; **UNDO(X)** - elimină ultimele X operații de **UNDO** și/sau **ADD** și **SUM(X,Y)** - determină suma elementelor din submatricea determinată de colțul din stânga sus (1, 1) și colțul dreapta jos (X,Y). Dându-se $n = 5$ și seria de operații de mai jos, determinați răspunsul pentru fiecare operație de tip **SUM(X,Y)**.

**ADD**(1,1,8), **ADD**(2,3,10), **SUM**(1,3), **ADD**(3,2,6), **ADD**(2,2,4), **SUM**(3,2), **UNDO**(2), **ADD**(1,1,7), **SUM**(5,5), **UNDO**(3), **SUM**(4,4)

(a) 8, 18, 25, 24 (b) 8, 18, 25, 0 (c) 18, 18, 25, 0 (d) 8, 28, 25, 24

10. Se implementează un algoritm care caută secvența de sumă maximă dintr-un vector de numere întregi și calculează valoarea acestei sume. De exemplu, pentru un vector cu valorile 2, 3, -6, 4, 6, 8, 11, -9, 1, 2 secvența de sumă maximă este 4, 6, 8, 11 iar suma maximă este 29. Algoritmul parcurge vectorul și la fiecare pas verifică semnul secvenței curente (**sc**), modifică corespunzător valoarea **sc** și indexul de inceput al secvenței curente (**pc**) si actualizează secvența de suma maximă (**smax**) cu păstrarea indicilor de inceput (**p**) și sfârșit (**u**) ai secvenței.

```
int smax = a[0], sc = a[0];
int p = 0, u = 0, pc = 0;
    for (i = 1; i <= N; i++)
    {
        if (sc > 0) sc += a[i];
        else
        {
            S;
        }
        if (sc > smax)
        {
            p = pc; u = i; smax = sc;
        }
    }
```

Care sunt instrucțiunile aferente secvenței S?

(a) $sc = smax + a[i]; pc = i;$

(b) $sc = smax + a[i]; pc = i + 1;$

(c) $sc = a[i]; pc = i;$

(d) $sc = a[i-1]; pc = pc++;$

11. Fie următoarea funcție C++ care returnează 1 dacă numărul dat ca parametru este prim, respectiv 0 dacă nu este prim.

```
int isPrime(int n) {
    for(int i=2; i<=X; i++) {
        if(!(n%i)) {
            return 0;
        }
    }
    return 1;
}
```

Care este valoarea minimă pe care o poate lua X astfel încât funcția să returneze rezultate corecte pentru $n$ oricât de mare, unde $n > 1$?

(a) $(int)(n/2)$ (b) $n$ (c) $log(n)$ (d) $(int)(sqrt(n))$

12. Fie următoarea funcție ce are ca parametri de intrare un vector de numere întregi și un numar natural nenul:

**funcția** `f`($v$, $n$)
- $h := n$
- $sum := 0$
- **cât timp** $h > 0$ **execută**
  - **pentru** $i := 0$ **la** $n - 1$ **execută**
    - $sum := sum + v[i]$
  - $h := h/2$
- **returnează** $sum$

Care este ordinul de complexitate al functiei `f`?

(a) O($N$) (b) O($N^2$) (c) O($N$log$N$) (d) O(log$N$)

13. Se dorește căutarea valorii 70 în vectorul de elemente: -6, 0, 4, 7, 11, 13, 22, 23, 40, 44, 47, 66, 68, 70, 71.

Câte operații de comparație între valoarea 70 și elemente din vector se realizează în urma aplicării algoritmului de căutare binară?

(a) 12 (b) 3 (c) 2 (d) 4

14. Se consideră problema următoare: dată fiind $S = \{1, 2, 3, 4\}$ mulțimea a 4 regine şi o tablă de tip şah (4 linii si 4 coloane, numerotate de la 1 la 4), să se plaseze cele 4 regine astfel încât să nu existe două regine pe aceeaşi linie, coloană sau diagonală. Problema este rezolvată prin metoda backtracking. Plasarea unei regine $i$ se face pe linia $i$, prin testarea de la stânga la dreapta a pozițiilor acceptabile. Se presupune că reginele 1 si 2 sunt plasate ca în figura de mai jos.

| | | | |
|---|---|---|---|
| 1 | | | |
| | | 2 | |
| | | | |
| | | | |

Se încearcă plasare reginei 3. Care dointre figurile de mai jos este figura care reprezintă următoare operație ce trebuie efectuată?

(a)

| | | | |
|---|---|---|---|
| | 1 | | |
| | | | 2 |
| | | | |
| | | | |

(b)

| | | | |
|---|---|---|---|
| 1 | | | |
| | | | 2 |
| | | | |
| | | | |

(c)

| | | | |
|---|---|---|---|
| | 1 | | |
| | | | 2 |
| 3 | | | |
| | | | |

(d)

| | | | |
|---|---|---|---|
| | | 1 | |
| 2 | | | |
| | | | |
| | | | |

15. Între cele n elemente ale unui vector se găsesc k elemente ce sunt considerate necorespunzătoare. Un element aflat pe poziția i este considerat necorespunzător dacă atât suma elementelor de pe pozițiile $1, 2, ..., i-1$, cât și suma elementelor de pe pozițiile $i+1, ..., n$ este mai mică decât elementul de pe poziția $i$. Mihai încearcă să implementeze un algoritm eficient care să rezolve această problemă. Indicați ordinul de complexitate al celui mai bun algoritm pe care îl poate găsi Mihai.

(a) $O(1)$ (b) $O(n)$ (c) $O(\log_2 N)$ (d) $O(n^2)$

16. Mihai are un număr de sarcini care trebuie duse la îndeplinire. Fiecare sarcină poate să depindă de alte sarcini, care depind la rândul lor de altele, etc. De exemplu, înainte de a se apuca de vopsit în casă, trebuie să cumpere vopsea și pensulă.

Fie N numărul total de sarcini și sarcinile S1, S2, ..., Sn. Relațiile de dependență sunt reprezentate prin intermediul matricei M, unde, dacă M[i,j]=1, atunci sarcina Si trebuie îndeplinită înaintea sarcinii Sj. De exemplu, sarcina S4 trebuie îndeplinită înaintea sarcinii S1.

| | **S1** | **S2** | **S3** | **S4** | **S5** | **S6** | **S7** | **S8** |
|---|---|---|---|---|---|---|---|---|
| **S1** | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| **S2** | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| **S3** | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| **S4** | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **S5** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| **S6** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **S7** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| **S8** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Care este o ordine corectă de îndeplinire a sarcinilor ținând cont de dependențele reprezentate în matricea de mai sus?

(a) $6, 1, 3, 4, 2, 7, 8, 5$ (b) $5, 7, 2, 3, 8, 4, 1, 6$ (c) $1, 6, 2, 3, 4, 5, 7, 8, 6$ (d) $6, 1, 2, 3, 4, 8, 5, 7$

17. Se consideră un graf $G$ cu $n$ vârfuri și $m$ muchii format din $k$ componente conexe. Câte muchii din graful $G$ trebuie eleminate pentru a transforma componentele conexe în arbori parțiali (un arbore parțial se obține prin eliminarea unor muchii din graf)?

(a) $m-n+k$ (b) $n-k$ (c) $m-k$ (d) $k$

18. Fie $D=(V,A)$ un digraf cu $n$. Un vârf $i$ se numește *groapă* dacă pentru orice alt vârf $j \neq i$ există un arc $(j,i) \in A$ și nu există un arc $(i,j) \in A$. Care este numărul maxim de *gropi* ce pot exista într-un digraf?

(a) $n$ (b) 1 (c) $n-1$ (d) 0

19. Fie $T$ un arbore binar cu $n$ noduri dintre care $f$ sunt frunze. Un nod complet este un nod care are doi fii. Cate noduri complete poate avea arborele $T$?

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

20. Numerele raționale pozitive sunt acele numere care sunt egale cu $m/n$, cu $m$ și $n$ numere naturale, prime între ele, $\leqslant 32000$. Există un mod foarte interesant de a obține toate numerele raționale: pentru început se consideră două numere: $0/1$ adică 0 și $1/0$ adică un fel de $infinit$. Pornind de la acestea putem obține un nou șir de numere raționale combinând două numere consecutive. Se aleg cele 2 numere. Fie ele de tipul $a/b$ și $c/d$. Noul rezultat este $(a+c)/(b+d)$ și se pune între cele 2. De exemplu primului șir îi urmează: $1/1$ adică $(0+1)/(1+0)$, mai apoi $1/2$ adică $(0+1)/(1+1)$ și $2/1$ adică $(1+1)/(1+0)$, șamd. Prin acest procedeu se poate genera o rețea de noduri organizată pe mai multe niveluri:

- pe nivelul 0: $0/1$ și $1/0$
- pe nivelul 1: $1/1$
- pe nivelul 2:$1/2$ și $2/1$

...

Pentru rețeaua obținută urmând procedeul de mai sus, se aplică următoarea codificare, pentru oricare două noduri situate pe niveluri consecutive **X** și **Y**: dacă valoarea din nodul **Y** este mai mică decât valoarea din nodul **X**, atunci drumul de la nodul **X** către nodul **Y** se codifică cu $\boldsymbol{S}$, în caz contrar drumul între cele două noduri se codifică cu $\boldsymbol{D}$. Indicați traseul străbătut în rețea pentru a ajunge la fracția $m/n$, pornind din vârful $1/1$. Spre exemplu: pentru a ajunge la fracția $3/1$ drumul codificat este: **D**, **D**.

(a) DSS (b) DSD (c) DDD (d) SDD

21. Fie un graf neorientat cu șase vârfuri, în care fiecare vârf are asociat câte un vector de elemente întregi. Două vârfuri au muchie între ele dacă există măcar o valoare comună în vectorii asociați vârfurilor respective. Fiecare muchie are o pondere egală cu maximul valorilor comune. Dacă nu există muchie între două vârfuri, atunci se consideră că are ponderea infinit.

Vectorii pentru fiecare vârf sunt: (4, 5, 6), (2, 3, 4), (3, 4, 5), (1, 2, 3), (5, 6, 7), (1, 4, 7).

Care este maximul lungimilor minime dintre oricare două vârfuri din graful respectiv?

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

22. Se consideră următoarea funcție recursivă fun(x,y). Care este rezultatul apelului fun(4, 3)?

```
int fun(int x, int y)
{
    if (x == 0) return y;
    return fun(x - 1, x + y);
}
```

(a) 12 (b) 13 (c) 9 (d) 10

23. Se consideră următoarea funcție recursivă:

```
int foo(int n, int r)
{
    if (n > 0)
        return (n%r + foo (n/r, r ));
    else
        return 0;
}
```

Ce valoare se obține la apelul foo(345, 10) ?

(a) 345 (b) 12 (c) 10 (d) 5

24. De câte ori va apărea pe ecran semnul # după apelul functie(4) ?

```
void functie (int n)
{
    cout<<"#";
    if(n!=0)
    {
        cout<<n<<" ";
        for(int j=0;j<n;j++)
            cout<<j;
        cout<<endl;
        functie (n-1);
        cout<<"#";
    }
    else cout<<"#";
}
```

(a) De 3 ori; (b) De 14 ori; (c) De 10 ori; (d) De 12 ori.

25. Un singur vector de dimensiune *MAXSIZE* este utilizat pentru stocarea a două stive. Cele două stive își modifică pozițiile vârfurilor (dimensiunile) pornind de la cele două capete ale vectorului. Variabilele *top1* si *top2* (top1<top2) indică pozițiile din vârfurile stivelor. Pentru o utilizare eficientă a spațiului de memorie, condiția de stivă plină este:

(a) $(top1 = MAXSIZE/2)$ *și* $(top2 = MAXSIZE/2 + 1)$

(b) $top1 = top2 - 1$

(c) top1 + top2 = MAXSIZE

(d) (top1= MAXSIZE/2) sau (top2 = MAXSIZE)