

# Grile - Modulul 1

# 1 Algoritmi și programare

1. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>
void main(){
    int a,b = 0;
    int i;
    for (i=0;i<10;i++){
        b = a;
        a = i;
    }
    printf("%d", b);
}
```

- A. 8
- B. 9
- C. 10
- D. 0123456789

2. Care din următoarele expresii sunt echivalente cu `a[i]`?

- A. `*(a+i)`
- B. `i[a]`
- C. `&a[i]`
- D. `*(a+a)`
- E. `*(a+i*sizeof(a[i]))`

3. Instrucțiunea `k++` este echivalentă cu:

- A. `k = k + 1`
- B. `k += 1`
- C. `++k`
- D. `k = k + k`
- E. `k += k`
- F. `k = k + sizeof(k)`

4. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>
int a,b;
void f1 (int *r, int *s){
    int temp;
    temp = *r;
    *r = *s;
    *s = temp;
}
void f2 (int *x, int *y){
    if (*x > *y) f1(x,y);
}
main(){
    a = 64;
    b = 42;
    f2(&a,&b);
    printf("%d,%d\n",a,b);
}
```

- A. 42,64

B. 64,42

C. 64,64

D. 42,42

E. eroare de compilare pentru că lipsește void la declararea funcției main

5. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>
int a,b;
void f1 (int r, int s){
    int temp;
    temp = r;
    r = s;
    s = temp;
}
void f2 (int x, int y){
    if (x > y) f1(x,y);
}
void main(){
    a = 064;
    b = 042;
    f2(a,b);
    printf("%d,%d\n",a,b);
}
```

**A. 52,34**

B. 64,42

C. 42,62

D. 34,52

E. 0,0

F. rezultatul nu poate fi determinat având în vedere valorile variabilelor a și b

6. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>
void main(){
    int k[ ] = {100,200,300,400,500,600,700};
    int *t=k+2;
    printf("%d ",*t);
    printf("%d ",*(t+2) + *t);
    *t = *t + 10;
    printf("%d ",*t);
    t = t+3;
    printf("%d ",*t);
    *t = *t + *(t-2);
    printf("%d ",*t);
}
```

**A. 300 800 310 600 1000**

B. 200 300 110 600 1100

C. 300 900 320 600 1000

D. 300 800 310 600 900

E. 100 900 310 600 1000

F. nici una din variante nu este corecta

7. Ce se va afișa pe ecran în urma execuției următorului program?

```

#include <stdio.h>
void main(){
    int p[] = {50,60,70,80,90,100};
    int *q[6],i;
    for(i =0;i<6;i++)
        q[i] = &p[i];

    for(i=5;i>=1;i--){
        q[i] = q[i-1];
        *q[i] = *q[i] + *p * 2;
    }
    for(i=0;i<6;i++)
        printf("%d ",p[i]);
}

```

- A. 150 160 170 180 190 100
- B. 160 160 170 180 190 100
- C. 150 150 150 150 150 150
- D. 150 160 170 180 190 190
- E. 150 160 170 180 190 200
- F. nici una din variante nu este corectă

8. Ce se va afișa pe ecran în urma execuției următorului program?

```

#include <stdio.h>
#include <stdlib.h>
void f(int *x){
    x=(int *) malloc(sizeof(int));
    *x=12;
}
void main(){
    int v=15/10;
    f(&v);
    printf("%d",v);
}

```

- A. 1
- B. 12
- C. 15
- D. 1.5
- E. adresa variabilei v
- F. adresa variabilei x

9. Ce se va afișa pe ecran în urma execuției următorului program?

```

#include <stdio.h>
void main(){
    int i, p=0x10;
    for(i=1;;i+=2){
        if (i=5)
            break;
        p+=i;
    }
    printf("%d",p);
}

```

- A. 16
- B. 10
- C. 26
- D. 25
- E. 14
- F. programul ciclează la infinit

10. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>
int a = 10, b = 5;
void f (int a, int t[]){
    b = a;
    t[2] = a % 2;
    a = (t[0]+t[1])/2;
    t[0]^=t[1];
    t[1]^=t[0];
    t[0]^=t[1];
}
void main(){
    int t[]={4,3,1};
    f(t[2],t);
    printf("%d %d %d %d %d",a,b,t[0],t[1],t[2]);
}
```

- A. 10 1 3 4 1
- B. 10 5 4 3 1
- C. 1 5 3 4 1
- D. 7 5 3 4 1
- E. 5 5 10 4 0
- F. nici una din variante nu este corectă

11. Considerăm următoarea secvență de program?

```
float x = 10;
int y = 7;
float t[]={3,4,1};
float *q = &x;
int *r = &y;
void *p;
```

Care din următoarele atribuiri sunt corecte?

- A. q = t;
- B. p = t;
- C. p = &y;
- D. p = r;
- E. x = &t[2];
- F. t = q;
- G. q = r;
- H. r = q;
- I. r = p;

12. Ce se va afișa pe ecran în urma execuției următorului program?

```

#include <stdio.h>
void main(){
    char s[]="abcdef";
    char *p,*q,c;
    p=s;
    q=p+4;
    while(p < q){
        c=*p;
        *p=*q;
        *q=++c;
        p++;q--;
    }
    printf("%s",s);
}

```

- A. edccbf
- B. fedcba
- C. feddca
- D. edcbaf
- E. bcdefg
- F. nici una din variante nu este corectă

13. Care din următoarele linii vor genera eroare la compilare?

```

1: void main() {
2:   int a[2] = {1,2};
3:   int b[3] = {3};
4:   int* x = a;
5:   int* const y = a;
6:   b = x;
7:   b = y;
8:}

```

- A. 6, 7
- B. 3, 5, 6, 7
- C. 3, 4, 6, 7
- D. 3, 5
- E. 5, 6, 7
- F. 5
- G. nici una

14. Considerăm următorul program:

```

1: #include <stdio.h>
2: void main(){
3:   int t[10];
4:   printf("sizeof(int)=%d\n",sizeof(int));
5:   printf("%p\n",t);
6:   printf("%p\n",t+3);
7:}

```

Dacă în urma execuției liniilor 4 și 5 se afișează 2 respectiv 0xF004, ce se va afișa în urma execuției liniei 6?

- A. 0xF00A
- B. 0xF010

- C. 0xF007
- D. 0xF006
- E. 0xF003

15. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>
void main(){
    int x=7;
    x = 1,2,3;
    printf("%d",x);
}
```

- A. 1**
- B. 7
- C. 2
- D. 3
- E. 6
- F. nimic (programul nu se compilează)

**Solution:** Operatorul virgulă se evaluează de la stânga la dreapta. Rezultatul este dat de ultima valoare a expresiei. Însă, deoarece operatorul = are prioritatea mai mare decât a operatorului virgulă mai întâi se atribuie lui x valoarea 1 și apoi se evaluează expresia x,2,3, rezultat care nu este memorat în nicio variabilă.

16. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>
void main() {
    int array[3][ ] = {{0, 1, 2}, {3, 4, 5}, {6, 7, 8}};
    int s = 0, i ,j;
    for (i = 0; i < 3 ; ++i){
        for (j = 2; j < 3 ; j++){
            s += array[i][j];
        }
    }
    printf("%d",s);
}
```

- A. nimic (programul nu se compilează)**
- B. 15
- C. 0
- D. 27
- E. 36

**Solution:** La declararea tablourilor multidimensionale numai prima dimensiune poate fi omisă. În acest caz, se generează eroare de compilare deoarece nu este specificat numărul de coloane.

17. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <stdio.h>

int f(int y);

void main(){
    int x = 3;
    int y = 6;
    printf("%d", f(x));
}

int f(int x){
    return x+1;
}
```

**A. 4**

B. 7

C. 3

D. 6

E. nimic (eroare de compilare deoarece funcția **f** a fost apelată înainte de a fi definită)

F. nimic (eroare de compilare deoarece numele argumentului funcției **f** este **y** la declarare și **x** la definire)

18. Câte elemente ale vectorului **a** vor avea valoarea 9 după execuția programului de mai jos?:

```
void main(){
    int a[] = {0, 1, 2, 3, 0, 4, 5, 6};
    int i = 0, x = 9;

    do{
        a[i++] = x;
    }while(i<6&& a[i]);
}
```

A. nici unul

B. unu

C. două

**D. patru**

E. toate

19. Fie urmatorul program:

```
void main (){
    int v[20], i, n, D;

    scanf("%d", &n);
    for(i=0;i<n;i++)
        v[i]=i%2?-i;
    for(D=1,i=0;i<n;D*=v[i++]);
    D++;
    printf("%d",D);
}
```

În urma execuției sale sunt posibile următoarele situații:

A. Expresia condițională din primul ciclu **for** este eronată din punct de vedere sintactic.



**B. Dacă variabila n primește la citire valoarea 6, atunci elementele vectorului v vor fi, în ordine (0,1,-2,3,-4,5).**

C. Prezența caracterului " ; " după al doilea ciclu for constituie o eroare

**D. Dacă variabila n primește la citire valoarea 5, atunci programul afișează 1**

E. Programul funcționează corect pentru orice valoare întregă a lui n mai mică sau egală cu MAXINT.

20. Fie programul :

```
void main(){
    int v[]={0, 1, 2, 3, 4, 5, 0};
    int i=0, n=0;

    do{
        if (i == v[i])
            n++;
    }while(i<6 && v[i++]);
}
```

În urma execuției programului sunt posibile următoarele situații:

A. variabila n va avea valoarea 0

**B. variabila n va avea valoarea 1**

C. programul va intra într-un ciclu infinit

D. variabila n va avea valoarea 5

E. variabila n va avea valoarea 2

21. Se consideră secvența următoare, în care valorile lui n și x se presupun cunoscute, v este un vector cu elementele(v[0],v[1],...,v[n-1])

```
p=n;
for(i=0;i<n;i++)
    if (v[i]==x)
        p=i;
for(i=p+1; i<n; i++)
    v[i-1]=v[i];
for(i=0; i<n-1; i++)
    printf("%3d",v[i]);
```

Precizați care dintre următoarele afirmații sunt adevărate:

A. Pentru n=5, x=3 și v=(5,6,2,7,1), se afișează ultimele patru elemente nemodificate ale vectorului:6 2 7 1.

B. Pentru n=5, x=1 și v=(2,1,3,1,4), se afișează: 2 3 1 4;

C. Secvența conține erori de sintaxa

D. Algoritmul șterge din vector elementul cu valoarea x, prin mutarea cu o poziție mai la dreapta a elementelor aflate înaintea lui.

**E. Algoritmul șterge din vector elementul cu valoarea x, prin mutarea cu o poziție mai la stânga a elementelor aflate după el.**

22. În programul următor, care dintre secvențele de instrucțiuni (I), (II), (III) realizează corect citirea unui șir de caractere de la tastatură și afișarea acestuia?

```
void main(){
    char s1[10],s2[10],s3[10];
    scanf("%s", &s3[3]);
```

```

printf("%s", s3[3]);           //(I)
scanf("%s", s2);
printf("s2=%s", s2);         //(II)
scanf("%s",&s1);
printf("%s", s1[10]);        //(III)
}

```

- A. numai (I)
- B. toate
- C. (I) și (II)
- D. (I) și (III)
- E. numai (II)**

23. Pentru programul următor, analizați corectitudinea afirmațiilor de mai jos:

```

#include <stdlib.h>
#include <string.h>
#include <stdio.h>

void main(){
    char s1[4], s2[4];
    long x;

    scanf("%s %s", s1, s2) ;
    if (strcmp(s1, s2)>0)
        x=atol(s1);
    else
        if(strcmp(s1, s2)==0)
            x = 0;
        else x = atol(s2);
    printf("%d", x);
}

```

- A. Condițiile din cele doua linii if sunt greșite.
- B. Apelurile funcției atol sunt corecte**
- C. Dacă de la tastatura se introduc șirurile "98" și "123" atunci se va afișa 98.**
- D. Dacă de la tastatura se introduc șirurile "123" și "121", atunci programul va afișa șirul "123".**
- E. Dacă de la tastatura se introduc șirurile "ab" și "ac", atunci se va semnala un mesaj de EROARE.

24. În conformitate cu standardul ASCII, codurile literelor mari sunt succesive începând cu 65, ce va afișa programul de mai jos?

```

#include <stdlib.h>
#include <string.h>
#include <stdio.h>

void main(){
    int x = 20, e;
    char s[15] ="ABC", t[15], u[15];

    e = s[1] + s[2];
    itoa(e, t, 10);
    strcpy(u, t);
}

```

```

    strcat(s, u);
    printf("%s ", s);
}

```

- A. Nimic, șirul s fiind vid
  - B. ABC13
  - C. AB13
  - D. ABC133**
  - E. ABC131
25. Care dintre specificatorii modului de acces al fișierelor binare sau text, pentru funcția fopen, sunt corecți:
- A. ab**
  - B. rb**
  - C. ap
  - D. wt**
  - E. at**
  - F. ba
  - G. toate variantele sunt corecte
26. Știind că fișierul f1.txt există în directorul curent iar f2.txt nu este creat, care dintre următoarele secvențe de instrucțiuni este greșită?
- A. f1=fopen("f1.txt","w");
  - B. f1=fopen("f1.txt","r+");
  - C. f2=fopen("f2.txt","r+");**
  - D. f2=fopen("f2.txt","w+");
  - E. toate variantele anterioare sunt greșite
27. Care este efectul subprogramului alăturat?

```

void X(char *a, char *b){
    FILE *f,*g;
    char s[255];

    f=fopen(a,"a");
    g=fopen(b,"r");
    while(!feof(g)){
        fgets(s,255,g);
        fputs(s,f);
    }
    fclose(f);
    fclose(g);
}

```

- A. copiază conținutul fișierului g peste conținutul fișierului f
- B. citește informațiile din cele două fișiere ale căror nume se transmit ca parametri
- C. concatenează două fișiere, rezultatul concatenării fiind pus în fișierul f**

D. concatenează două fișiere, rezultatul concatenării fiind pus în fișierul g

E. toate variantele anterioare sunt greșite

28. Care dintre următoarele afirmații sunt adevărate?

A. Pentru a închide un fișier se folosește funcția unlink;

B. Pentru redenumirea unui fișier în cadrul programului se folosește funcția remove

**C. Pentru a deschide un fișier se folosește funcția fopen**

D. Nu pot fi adăugate informații într-un fișier

E. Toate celelalte variante sunt eronate

29. Ce valori se vor găsi în fișierul numere.txt după execuția următorului program?

```
#include <conio.h>
#include <stdio.h>

FILE *f;
int i=0, a[10]={20,11,17,4,5,10,14,34,23,11};

void main(){
    f=fopen("c:\\numere.txt","w");
    for(i=0; i<5; i++){
        if(a[i]%2!=0)
            fprintf(f,"%d \n",a[i]);
    }
    fclose(f);
}
```

A. 11 17 5 23 11

B. 20 4 10 14 34

**C. 11**

**17**

**5**

D. 11

17

5

23

11

30. Ce se întâmplă în urma execuției următorului program dacă fișierul nr.txt conține valorile 7 14 6 3 8 10 ?

```
FILE *f, *g;
int x;
void main(){
    f=fopen("nr.txt","r+");
```

```

g=fopen("nr2.txt","w+");
while(!feof(f)){
    fscanf(f,"%d", &x);
    if(x%2==0)
        fprintf(g,"%d \n",x);
}
fclose(f);
fclose(g);
}

```

A. Fișierul nr2.txt va conține valorile 14 6 8 10

**B. Fișierul nr2.txt va conține valorile 14 6 8 10, fiecare pe altă linie**

C. Fișierul nr.txt va conține valorile 7 14 6 8 10

D. Fișierul nr.txt va conține valorile 7 14 6 8 10, fiecare pe altă linie

## 2 Structuri de date și tehnici de elaborare a algoritmilor

1. Se da următorul algoritm:

---

**Algorithm 1** Algoritm

---

```

procedure ALGORITM( $a, n$ )
  for  $i \leftarrow 1, n - 1$  do
    for  $j \leftarrow n, i + 1$  STEP  $- 1$  do
      if  $a_j < a_{j-1}$  then  $a_{j-1} \leftrightarrow a_j$ 
      end if end if
    end forend for  $j$ 
  end forend for  $i$ 
end procedure end procedure

```

---

Care vor fi valorile vectorului  $a$  după terminarea pasului  $i = 5$ ,  $a = (8, 6, 4, 2, 3, 5, 7)$ ?

**A. (2, 3, 4, 5, 6, 8, 7)**

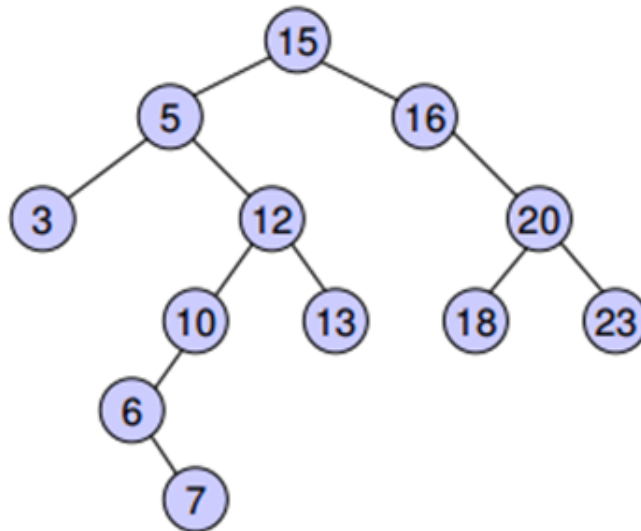
B. (2, 3, 4, 5, 8, 7, 6)

C. (2, 3, 4, 8, 7, 6, 5)

D. (8, 7, 6, 5, 4, 2, 3)

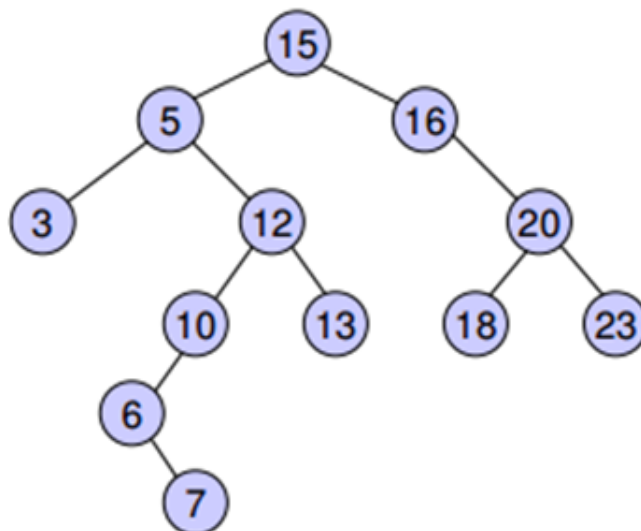
E. (2, 3, 4, 5, 6, 7, 8)

2. O procedură ce parcurge următorul arbore în inordine va afișa:



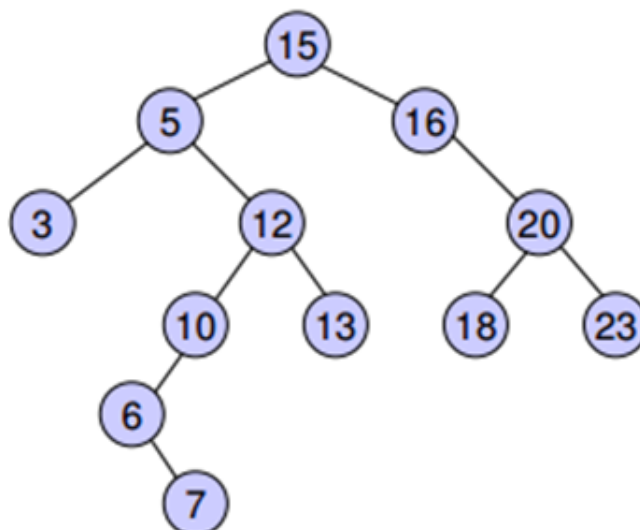
- A. 3, 5, 6, 7, 10, 12, 13, 15, 16, 18, 20, 23
- B. 15, 5, 3, 12, 10, 6, 7, 13, 16, 20, 18, 23
- C. 3, 7, 6, 10, 13, 12, 5, 18, 23, 20, 16, 15
- D. 3, 6, 5, 7, 10, 12, 13, 15, 16, 18, 20, 23
- E. 3, 5, 6, 7, 10, 12, 13, 16, 15, 18, 20, 23

3. O procedură ce parcurge următorul arbore în postordine va afișa:



- A. 3, 5, 6, 7, 10, 12, 13, 15, 16, 18, 20, 23
- B. 15, 5, 3, 12, 10, 6, 7, 13, 16, 20, 18, 23
- C. 3, 7, 6, 10, 13, 12, 5, 18, 23, 20, 16, 15
- D. 3, 6, 5, 7, 10, 12, 13, 15, 16, 18, 20, 23
- E. 3, 5, 6, 7, 10, 12, 13, 16, 15, 18, 20, 23

4. O procedură ce parcurge următorul arbore în preordine va afișa:



- A. 3, 5, 6, 7, 10, 12, 13, 15, 16, 18, 20, 23
- B. 15, 5, 3, 12, 10, 6, 7, 13, 16, 20, 18, 23**
- C. 3, 7, 6, 10, 13, 12, 5, 18, 23, 20, 16, 15
- D. 3, 6, 5, 7, 10, 12, 13, 15, 16, 18, 20, 23
- E. 3, 5, 6, 7, 10, 12, 13, 16, 15, 18, 20, 23

5. Care dintre următoarele conditii nu este conditia necesară pentru un algoritm de căutare binară:
- A. lista trebuie să fie sortată
  - B. ar trebui să existe acces direct către elementul din mijlocul fiecărei subliste
  - C. ar trebui să existe un mecanism de stergere sau/si de inserare a elementelor în listă**
  - D. niciuna de mai sus
6. Care dintre următoarele nu este afirmatii nu este adevarată in cazul unui algoritm de cautare binara?
- A. trebuie folosit un array sortat
  - B. cerinta de a avea un array sortat necesita multa memorie si timp atunci cand sunt necesare multe stingeri si inserari
  - C. trebuie sa existe un mecanism care sa permita accesul direct la elementul din mijloc
  - D. algoritmul de cautare binar nu este eficient atunci cand avem mai mult de 1000 de elemente**
7. Ce va afisa urmatorul program?

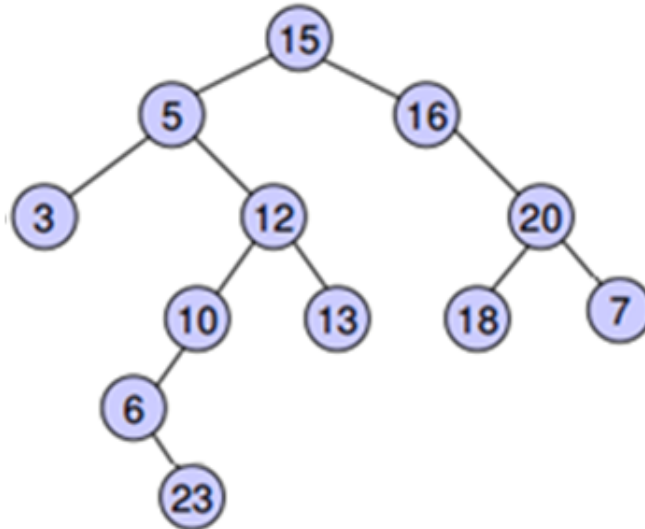
```

String name;
int i;
boolean startWord;
name = "Franklin D. Roosevelt";
startword = true;
for (i = 0; i < name.length(); i++) {
    if (startWord)
        System.out.println(name.charAt(i));
    if (name.charAt(i) == ' ')
        startWord = true;}
    else
        startWord = false;
}
  
```

**A. FDR**

- B. Franklin
- C. D
- D. Roosevelt
- E. Franklin D. Roosevelt

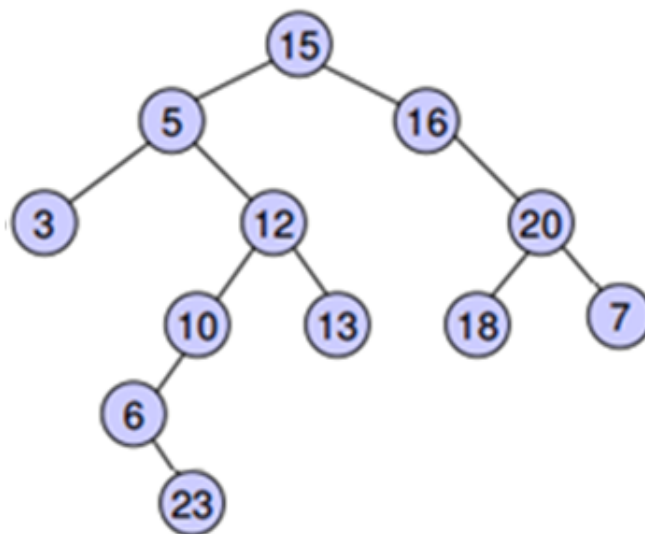
8. O procedură ce parcurge următorul arbore în inordine va afișa:



**A. 3, 5, 6, 23, 10, 12, 13, 15, 16, 18, 20, 7**

- B. 15, 5, 3, 12, 10, 6, 7, 13, 16, 20, 18, 23
- C. 3, 7, 6, 10, 13, 12, 5, 18, 23, 20, 16, 15
- D. 3, 6, 5, 7, 10, 12, 13, 15, 16, 18, 20, 23
- E. 3, 5, 6, 7, 10, 12, 13, 16, 15, 18, 20, 23

9. O procedură ce parcurge următorul arbore în postordine va afișa:

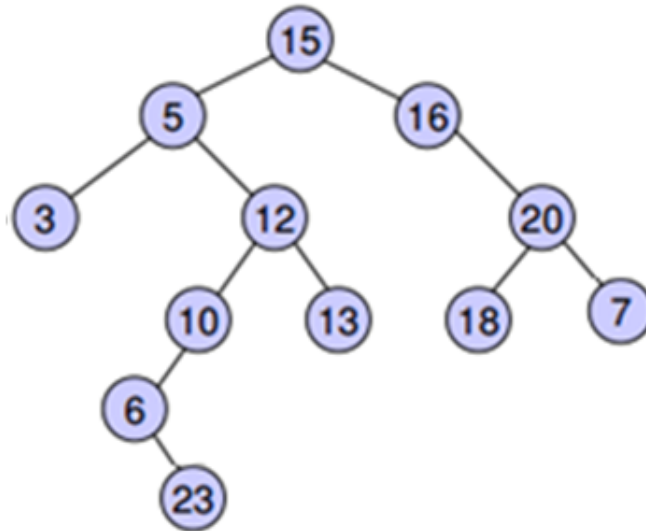


**A. 3, 5, 6, 7, 10, 12, 13, 15, 16, 18, 20, 23**



- B. 15, 5, 3, 12, 10, 6, 7, 13, 16, 20, 18, 23
- C. 3, 23, 6, 10, 13, 12, 5, 18, 7, 20, 16, 15**
- D. 3, 6, 5, 7, 10, 12, 13, 15, 16, 18, 20, 23
- E. 3, 5, 6, 7, 10, 12, 13, 16, 15, 18, 20, 23

10. O procedură ce parcurge următorul arbore în preordine va afișa:



- A. 3, 5, 6, 7, 10, 12, 13, 15, 16, 18, 20, 23
- B. 15, 5, 3, 12, 10, 6, 23, 13, 16, 20, 18, 7**
- C. 3, 7, 6, 10, 13, 12, 5, 18, 23, 20, 16, 15
- D. 3, 6, 5, 7, 10, 12, 13, 15, 16, 18, 20, 23
- E. 3, 5, 6, 7, 10, 12, 13, 16, 15, 18, 20, 23

11. Ce metoda de sortare este descrisă în pseudo-codul următor:

```

procedure sort(A)
  n = length(A)
  repeat
    swapped = false
    for i = 1 to n-1 do
      if (A[i-1]>A[i]) then
        swap (A[i-1],A[i])
        swapped = true
      end if
    end for
  until not swapped
end procedure
  
```

- A. HeapSort
- B. MergeSort
- C. BubbleSort**
- D. SwapSort

12. Coloana din stanga reprezinta un input de stringuri ce trebuie sortate; coloana din dreapta reprezinta stringurile sortate; celelalte coloane reprezinta un pas intermediar al unuia dintre algoritmi: quicksort, mergesort top-down, merge sort botton-up, heapsort. Care este ordinea algoritmilor tinand cont de coloanele afisate?

navy wine mist blue blue bark  
 plum teal coal gray coal blue  
 coal silk jade rose gray cafe  
 jade plum blue mint jade coal  
 blue sage cafe lime lime corn  
 pink pink herb navy mint dusk  
 rose rose gray jade navy gray  
 gray jade leaf teal pink herb  
 teal navy dusk coal plum jade  
 ruby ruby mint ruby rose leaf  
 mint pine lime plum ruby lime  
 lime palm bark pink teal mint  
 silk coal corn silk bark mist  
 corn corn navy corn corn navy  
 bark bark wine bark dusk palm  
 wine gray silk wine leaf pine  
 dusk dusk ruby dusk silk pink  
 leaf leaf teal leaf wine plum  
 herb herb sage herb cafe rose  
 sage blue rose sage herb ruby  
 cafe cafe pink cafe mist sage  
 mist mist pine mist palm silk  
 pine mint palm pine pine teal  
 palm lime plum palm sage wine  
 0 - - - 1

**A. Input, quicksort, heapsort, merge bottom-up, merge top-down, output**

- B. Input, heapsort, merge top-down, merge bottom-up, quicksort, output
- C. Input, quicksort, heapsort, merge top-down, merge bottom-up, output
- D. Input, merge top-down, merge bottom-up, quicksort, heapsort, output
- E. Input, merge top-down, quicksort, heapsort, merge bottom-up, output

13. Analizand algoritmul de mai jos precizati ce timp de executie are.

```

public static int f1(int N){
  int x = 0;
  for (int i = 0; i < N, i++)
    x++;
  return x;
}

```

- A.  $\log N$
- B.  $N$**
- C.  $N \log N$
- D.  $N^2$
- E.  $2^N$
- F.  $N!$

14. Analizand algoritmul de mai jos precizati ce timp de executie are.

```

public static int f2(int N){
  int x = 0;
  for (int i = 0; i < N, i++)
    for (int j = 0; j < i; j++)
      x++;
  return x;
}

```

- A.  $\log N$
- B.  $N$
- C.  $N \log N$
- D.  $N^2$**
- E.  $2^N$
- F.  $N!$

15. Analizand algoritmul de mai jos precizati ce timp de executie are.

```
public static int f3(int N){
    if (N==0) return 1;
    int x=0;
    for (int i = 0; i < N; i++)
        x+=f3(N-1);
    return x;
}
```

- A.  $\log N$
- B.  $N$
- C.  $N \log N$
- D.  $N^2$
- E.  $2^N$
- F.  $N!$**

16. Analizand algoritmul de mai jos precizati ce timp de executie are.

```
public static int f1(int N){
    int x = 0;
    for (int i = 0; i < N, i++)
        x++;
    return x;
}

public static int f4(int N){
    if (N==0) return 0;
    return f4(N/2) + f1(N) + f4(N/2);
}
```

- A.  $\log N$
- B.  $N$
- C.  $N \log N$**
- D.  $N^2$
- E.  $2^N$
- F.  $N!$

17. Analizand algoritmul de mai jos precizati ce timp de executie are.

```
public static int f1(int N){
    int x = 0;
    for (int i = 0; i < N, i++)
        x++;
    return x;
}

public static int f5(int N){
    int x = 0;
```

```

for (int i = N; i > 0; i = i/2)
    x+=f1(i);
return x;
}

```

- A.  $\log N$
- B.  $N$**
- C.  $N \log N$
- D.  $N^2$
- E.  $2^N$
- F.  $N!$

18. Analizand algoritmul de mai jos precizati ce timp de executie are.

```

public static int f6(int N){
    if (N==0) return 1;
    return f6(N-1) + f6(N-1);
}

```

- A.  $\log N$
- B.  $N$
- C.  $N \log N$
- D.  $N^2$
- E.  $2^N$**
- F.  $N!$

19. Analizand algoritmul de mai jos precizati ce timp de executie are.

```

public static int f7(int N){
    if (N==1) return 0;
    return 1+f7(N/2);
}

```

- A.  $\log N$**
- B.  $N$
- C.  $N \log N$
- D.  $N^2$
- E.  $2^N$
- F.  $N!$

20. Algoritmul Quicksort este cel mai potrivit pentru

- A. Sortarea unui liste mari de chei care sunt in ordine aleatoare**
- B. Sortarea unei liste de chei care sunt aproape ordonate
- C. Gasirea celui mai scurt drum
- D. Sortarea unei liste scurte de chei

### 3 Programare orientată obiect

1. Un program orientat obiect reprezintă:

- A. o colecție de obiecte care cooperează prin intermediul mesajelor în vederea realizării unui obiectiv comun**
- B. o variantă de program procedural
- C. un program care poate fi considerat un obiect de către alte programe

- D. un program specific mediului de programare Windows  
E. un program format din mai multe fișiere sursa și header
2. O clasă în modelul de programare orientat obiect reprezintă:
- A. o colecție de obiecte cu aceleași caracteristici și cu un comportament comun**
  - B. o colecție de caracteristici
  - C. un obiect din program
  - D. o interfață
  - E. un grup de comportamente specific unui obiect
3. Obiectele aparținând unei aceleiași clase se diferențiază prin:
- A. starea lor**
  - B. valorile datelor membru declarate în cadrul clasei**
  - C. valorile caracteristicilor clasei**
  - D. caracteristicile diferite specifice fiecărui obiect
  - E. comportamentul diferit specific fiecărui obiect
4. Considerăm următorul program C++

```
#include <iostream>
using namespace std;
class C
{
private:
int _i;
public:
    C(int i):_i(i){}
    C():_i(3){}
    ~C(){cout<<_i<<endl;}
};
void main()
{
    C c1(4);
    C c2;
}
```

Care din următoarele afirmații sunt adevărate:

- A. programul nu conține erori de sintaxă**
  - B. există erori de sintaxă în definițiile metodelor constructor
  - C. destructorul nu poate conține instrucțiuni de afișare
  - D. în cadrul programului principal sunt create două instanțe ale clasei C**
  - E. programul afișează la consolă numerele 3 și 4**
5. Care din următoarele afirmații referitoare la moștenire sunt adevărate relativ la limbajul de programare C++:
- A. Relația de moștenire în care clasa de bază și clasa derivată au aceeași interfață se numește moștenire pură**
  - B. Dintr-o clasă de bază se poate obține prin intermediul moștenirii cel mult o clasă derivată
  - C. O clasă derivată are exact o clasă de bază
  - D. Obiectele aparținând unei clase derivate pot fi convertite la tipul unei clase de bază din care clasa derivată provine**
  - E. Specificatorul protected este folosit pentru a arăta că elementele care îl urmează pot fi accesate și din cadrul claselor derivate**

6. Ce se va afișa pe ecran în urma execuției următorului program:

```
#include <iostream.h>
class Test {
    int n;
public:
    Test(int x):n(x){}
    int f(int n, int &x, int *p){
        n=this->n;
        x=n++;
        *p=this->n + n;
        return n;
    }
};
void main(){
    Test t(2);
    int n=6,x=0x10,p=1;
    cout<<t.f(n,x,&p)<<" ";
    cout<<n<<" "<<x<<" "<<p;
}
```

**A. 3 6 2 5**

B. 3 6 16 1

C. 6 6 10 1

D. 3 7 10 1

E. Eroare de compilare deoarece constructorul nu conține nici o instrucțiune

7. Ce se va afișa pe ecran în urma execuției următorului program:

```
#include <iostream.h>
class B1 {
public:
    B1(){cout<<"B1 ";}
    ~B1(){cout<<"DB1 ";}
};
class B2 {
public:
    B2(){cout<<"B2 ";}
    ~B2(){cout<<"DB2 ";}
};
class D: public B1, public B2{
public:
    D():B2(),B1(){cout<<"D ";}
    ~D(){cout<<"DD ";}
};
void main(){
    D d1;
    D d2=d1;
}
```

**A. B1 B2 D DD DB2 DB1 DD DB2 DB1**

B. B2 B1 D DD DB2 DB1 DD DB2 DB1

C. B1 B2 D B1 B2 D DD DB2 DB1 DD DB2 DB1

D. B1 B2 D DD DB1 DB2 DD DB1 DB2

E. D DD

8. Ce se va afișa pe ecran în urma execuției următorului program:

```

#include <iostream.h>
class B1 {
public:
    B1(){cout<<"B1 ";}
    ~B1(){cout<<"DB1 ";}
};
class B2 {
public:
    B2(){cout<<"B2 ";}
    ~B2(){cout<<"DB2 ";}
};
class D: public B1, public B2{
    static int idGen;
    int id;
public:
    D():B2(),B1(){id=idGen++;cout<<"D("<<id<<" ") ";}
    D(const D &o){id=idGen++;cout<<"CD("<<id<<" ") ";}
    ~D(){cout<<"DD ";}
};
int D::idGen = 0;

void main(){
cout<<endl;
    D d1;
    D d2=d1;
}

```

**A. B1 B2 D(0) B1 B2 CD(1) DD DB2 DB1 DD DB2 DB1**

B. B1 B2 D(0) DD DB2 DB1 DD DB2 DB1

C. B1 B2 D(0) DD DB2 DB1

D. B2 B1 D(0) B1 B2 CD(1) DD DB2 DB1 DD DB2 DB1

E. B1 B2 D(0) B2 B1 CD(1) DD DB2 DB1 DD DB2 DB1

F. rezultatul executiei nu poate fi determinat deoarece depinde de continul memoriei

9. Ce se va afișa pe ecran în urma execuției următorului program:

```

#include <iostream.h>
class B1 {
public:
    B1(){cout<<"B1 ";}
    ~B1(){cout<<"DB1 ";}
};
class B2 {
public:
    B2(){cout<<"B2 ";}
    ~B2(){cout<<"DB2 ";}
};
class D: public B1, public B2{
public:
    D():B2(),B1(){cout<<"D ";}
    D(const D &o){cout<<"CD ";}
    ~D(){cout<<"DD ";}
};
void main(){
    D *d1 = new D;
    D *d2 = d1;
    delete d1;
}

```

**A. B1 B2 D DD DB2 DB1**

- B. B1 B2 D DD DB2 DB1 DD DB2 DB1
- C. B2 B1 D DD DB2 DB1
- D. B2 B1 D DD DB2 DB1 DD DB2 DB1
- E. B1 B2 D B1 B2 D DD DB2 DB1 DD DB2 DB1
- F. B1 B2 D B1 B2 D DD DB2 DB1

10. Ce se va afișa pe ecran în urma execuției următorului program:

```
#include <iostream.h>
class A {
public:
    A(){cout<<"A()";}
    A(int n,int m=0){cout<<"A(int,int)";}
    A(const A &o){cout<<"A(const A&)";}
    ~A(){cout<<"~A";}
};

void main(){
cout<<endl;
A a1;
A a2=a1;
A a3=1;
A a4(1);
A a5(1,1);
}
```

**A. A()A(const A&)A(int,int)A(int,int)A(int,int)~A~A~A~A~A**

- B. A(const A&)A(const A&)A(int)A(int,int)A(int,int)~A~A~A~A~A
- C. A()A(const A&)A(int,int)A(int,int)A(int,int)~A~A~A~A
- D. A()A(const A&)A(int)A(int)A(int,int)~A~A~A~A~A
- E. A()A()A(int,int)A(int)A(int,int)~A~A~A~A~A
- F. Eroare de compilare la declararea obiectului a2 deoarece nu e supraîncărcat operatorul =.

11. Ce se va afișa pe ecran în urma execuției următorului program:

```
#include <iostream.h>
class B {
public:
    virtual void f() { cout<<"B::f() ";}
    void g() { cout<<"B::g() ";}
};
class D1: public B{
public:
    void f() { cout<<"D1::f() ";}
    void g() { cout<<"D1::g() ";}
};
class D2: public B{
public:
    void g() { cout<<"D2::g() ";}
};
void main(){
int i;
B *b[] = {new B(), new D1(), new D2()};
}
```



```

for (i=2;i>=0;i--) {b[i]->f();}
for (i=0;i<=2;i++) {b[i]->g();}
}

```

- A. B::f() D1::f() B::f() B::g() B::g() B::g()
- B. D2::f() D1::f() B::f() B::g() B::g() B::g()
- C. B::f() D1::f() D::f() B::g() D1::g() D2::g()
- D. B::f() D1::f() B::f() B::g() D1::g() D2::g()
- E. B::f() B::f() D2::f() B::g() B::g() D2::g()

12. Ce se va afișa pe ecran în urma execuției următorului program:

```

#include <iostream.h>
class B {
public:
    virtual void f() { cout<<"B::f() ";}
};
class D1: public B{
public:
    void f() { cout<<"D1::f() ";}
};
class D2: public B{
public:
    void f() { cout<<"D2::f() ";}
};
void main(){
    int i;
    B b[] = { B(), D1(), D2()};
    for (i=2;i>=0;i--) {b[i].f();}
}

```

- A. B::f() B::f() B::f()
- B. D2::f() D1::f() B::f()
- C. B::f() D1::f() D2::f()
- D. B::f() D1::f() D2::f()

13. Ce se va afișa pe ecran în urma execuției următorului program:

```

#include <iostream.h>
class B {
protected:
    int x;
public:
    B(int x=0){this->x=x;}
    virtual void f() {x=x+1;}
    void print(){cout<<x<<" ";}
};
class D1: public B{
public:
    void f() {x=x+5;}
};
class D2: public B{
public:
    void f() {x=x+2;}
};
void main(){

```

```

int i;
B *b[] = { new B(2), new D1(), new D2(), new B};
for (i=3;i>0;i--) (*b[i]).f();
for (i=0;i<=3;i++) b[i]->print();
}

```

- A. 2 5 2 1
- B. 2 0 0 0
- C. 3 5 2 1
- D. 2 1 1 1
- E. 2 5 2 0
- F. 2 5 2
- G. Eroare de compilare

14. Ce se va afișa pe ecran în urma execuției următorului program:

```

#include <iostream.h>
class B {
    int x;
public:
    B(int x=0){this->x=x;}
    virtual void f() {x=x+1;}
    void print(){cout<<x<<" ";}
};
class D1: public B{
public:
    void f() {x=x+5;}
};
class D2: public B{
public:
    void f() {x=x+2;}
};
void main(){
    int i;
    B *b[] = { new B(2), new D1(), new D2(), new B};
    for (i=3;i>0;i--) (*b[i]).f();
    for (i=0;i<=3;i++) b[i]->print();
}

```

- A. 2 5 2 1
- B. 2 0 0 0
- C. 3 5 2 1
- D. 2 1 1 1
- E. 2 5 2 0
- F. 2 5 2
- G. Eroare de compilare

15. Ce se va afișa pe ecran în urma execuției următorului program:

```

#include <iostream.h>
class B {
    int x;
    static int y;
public:
    B(int x=0){this->x=x;}
    void setX(int vx) {x=vx;}
}

```

```

    void setY(int vy) {y=vy;}
    void print(){cout<<x<<" "<<y<<" "};
};
int B::y=0;
void main(){
    B b1=3;
    B b2(4);
    b1.print(); b2.print();
    b1.setX(5); b2.setY(2);
    b1.print(); b2.print();
}

```

- A. 3 0 4 0 5 2 4 2
- B. 3 0 4 0 5 0 4 2
- C. 3 4 5 2 4 2
- D. 3 3 4 4 5 2 4 2
- E. 3 0 4 0 5 2 5 2
- F. 3 0 4 0 5 0 2 0

16. Ce se va afișa pe ecran în urma execuției următorului program?

```

#include <iostream.h>
struct A{
    int x;
    operator double() {
        return 21.4;
    }
};
void main(){
    A a;
    a.x = 11;
    cout << (0?3:a);
}

```

- A. 21
- B. 21.4
- C. 3
- D. 11
- E. programul nu se compilează

17. Ce se va afișa pe ecran în urma execuției următorului program?

```

#include <iostream.h>
struct A {
    A(int d) : x(d) {}
    int x;
};
void main(){
    double x = 3.14;
    A f( int(x) );
    cout << f.x << endl;
}

```

- A. nimic (eroare de compilare)
- B. 0
- C. 3

D. 3.14

E. depinde de implementarea compilatorului

18. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <iostream.h>

int f(int x, int y = x) {
    return x+y+1;
}

void main(){
    cout << f(2);
}
```

**A. nimic (eroare de compilare)**

B. 1

C. 3

D. 5

E. depinde de implementarea compilatorului

19. Ce se va afișa pe ecran în urma execuției următorului program?

```
#include <iostream.h>
struct A {
    virtual int f(int x = 5){
        return x * 2;
    }
};

struct B : public A{
    int f(int x = 10) {
        return x * 3;
    }
};

void main(){
    A* a = new B;
    cout << a->f();
}
```

**A. 15**

B. 10

C. 20

D. 3

E. nimic (eroare de compilare)

20. Fie secvența de program:

```
class A {
    private:
        int x,y;
    public:
        A(){x=0;y=0; }
        A(int xi,int yi){
            x=xi;y=yi;
```

```

    }
    void afis(){
        cout <<x " " << y;
    }
};
void main(){
    A A1,A2(10,20);
    A1.afis();
    A2.afis();
}

```

Care din urmatoarele afirmații sunt adevarate:

- A. Metoda afis() este incorect definită
- B. Constructorul este incorect definit
- C. Declarație incorectă pentru obiectul A1
- D. Programul afisează valorile (10 20 0 0 )

**E. Programul afisează valorile (0 0 10 20 )**

21. Fie secvența de program:

```

class C{
    public:
        C(){n++;}
        static int index(){return n;}
    private:
        static int n;
};
int C :: n = 0;
class A{
    private:
        C c;
        int a;
};
class B{
    public:
        B(int i = 0) :b(i)
        {cout <<C :: index() <<" " ;}
    private:
        A a;
        C c;
        int b;
};
void main(){
    A a1;
    cout <<C :: index() <<' ';
    B b1[3];
}

```

Rezultatul execuției programului este:

- A. 1 2 3 4
- B. 1 3 4 5
- C. 1 7 3 4
- D. 1 3 5 7**
- E. 0 1 2 3

22. Se consideră secvența de program:

```
class C {
    private:
        int x,y;
    public:
        C(int xi,int yi){ x=xi;y=yi; }
        C(const C &a){
            x=a.x;y=a.y;
        }
};
```

În care din următoarele situații se realizează copierea unui obiect în altul:

- A. C c1(4,5)
- B. C c2(0.0, 0,0)
- C. C c3=c1**
- D. C c4(1)
- E. C c5(c1)**

23. Se considera secvența de program:

```
class Punct {
    private:
        double x,y;
    public:
        Punct(double xi,double yi){
            x=xi;y=yi;
        }
};
```

În care din următoarele situații se realizează copierea unui obiect:

- A. Punct P1(10,20)
- B. Punct P2(P1(3))
- C. Punct P3
- D. Punct P4=P1**
- E. Punct P5(P1)**

24. Se dă programul :

```
class P {
    double x,y;
    public:
        P(double x1 = 0,double y1 = 0){
            x=x1;y=y1;
        }
        void afis(){
            cout <<x <<" " << y;
```

```

    }
    P operator++ (){
        ++x;++y;
        return *this;
    }
};
void main(){
    P p1(1,2),p2;
    p2=++p1;
    p2.afis();
}

```

Care din afirmațiile următoare sunt corecte:

- A. Obiectul *p2* nu poate fi instanțiat
- B. Asignarea  $p2 = ++p1$  este incorectă

**C. În program se realizează supraîncărcarea operatorului prefix de incrementare**

**D. Programul va afișa valorile: 23**

- E. Programul afișează valorile lui *p1*

25. Se dă programul :

```

class A {
    int x,y;
public:
    A(int xi = 0, int yi=0){
        x=xi;y=yi;
    }
    void afis(){
        cout <<" x" << y;
    }
    A operator+ (A o2);
    friend A operator++ (A &a);
};
A A::operator+ (A o2){
    A temp;
    temp.x=x+o2.x;
    temp.y=y+o2.y;
    return temp;
}
A operator++ (A &a){
    ++a.x;++a.y;
    return a;
}
void main(){
    A a1(20,30),a2(2,2),a3;
    a3=++a1+a2;
    a3.afis();
}

```

Care din afirmațiile următoare sunt corecte:

- A. Utilizarea cuvântului predefinit " friend " nu este permisă în acest context
- B. Funcția operatorului ++ nu este definită corect
- C. Programul va afișa valorile: 23 33**
- D. Atribuirea a3=++a1+a2 este incorectă
- E. programul va afișa valorile: 22 32
26. Supraîncărcarea unor operatori se poate realiza prin funcții operator sau funcții friend. Diferența constă în:
- A. Precedența operatorilor
- B. Asociativitatea operatorilor
- C. Obiectul returnat
- D. Lista de parametri**
- E. Numărul parametrilor din lista funcției**
27. Se considera secvența:
- ```
class A {
    int a[3];
public:
    A(int xi, int yi, int zi){
        a[0]=xi;a[1]=yi;a[2]=zi;
    }
    int &operator[](int i){
        return a[i];
    }
};

void main() {
    A o(1, 2, 3);
    cout<<o[0];
    o[1]=10;
    cout<<o[1];
}
```
- Ce se poate afirma despre operator[]()?
- A. Supraîncarcă operatorul()
- B. Supraîncarcă un operator unar
- C. Este o metoda oarecare a clasei care nu produce supraîncărcarea unor operatori
- D. Este un constructor
- E. Supraîncarcă operatorul[]**
28. Se consideră secvența:
- ```
class B{
    int a;
protected:
    int b;
public:
    int c;
    void set_a(int x){a = x;}
    void set_b(int y){b = y;}
    void set_c(int z){c = z;}
}
```



```

};
class D : public B{    int d;
    public:
        void set_b(int y) {b = y;}
        void set_c(int z) {c = z;}
};
void main(){
D o;
    o.a = 1;        //(1)
    o.B::set_a(2);  //(2)
    o.b = 3;        //(3)
    o.B::set_b(4); //(4)
    o.c = 5;        //(5)
}

```

Care din instrucțiunile (1)-(5) accesează corect membrii claselor:

- A. Toate
- B. (1), (2) și (5)
- C. (1), (3) și(4)
- D. (1), (2), (3), (4)
- E. (2), (4) și (5)**

29. În programul următor, care din instrucțiunile (1)-(5) accesează corect membrii claselor:

```

class B{
    int a;
    protected:
        int b;
    public:
        int c;
        void set_a(int x){a = x;}
        void set_b(int y){b = y;}
        void set_c(int z){c = z;}
};
class D : protected B{
    int d;
    public:
        void set_b(int y) {b = y;}
        void set_c(int z) {c = z;}
};
void main(){
D o;
    o.a = 1;        //(1)
    o.B::set_a(2);  //(2)
    o.b = 3;        //(3)
    o.B::set_b(4); //(4)
    o.c = 5;        //(5)
}

```

- A. Toate
- B. Nici una**
- C. Numai (5)
- D. (2), (4) și (5)

30. În programul următor, care din instrucțiunile (1)-(6) accesează corect membrii claselor:

```
class B{
    int a;
protected:
    int b;
public:
    int c;
    void set_a(int x){a = x;}
    void set_b(int y){b = y;}
    void set_c(int z){c = z;}
};
class D : private B{
    int d;
public:
    void set_b(int y) {b = y;}
    void set_c(int z) {c = z;}
};
void main(){
D o;
    o.a = 1;        //(1)
    o.B::set_a(2);  //(2)
    o.b = 3;        //(3)
    o.B::set_b(4); //(4)
    o.c = 5;        //(5)
    o.set_c(6);    //(6)
}
```

- A. (1) și (4)
- B. (2), (3), (4)
- C. Toate
- D. (6)**

## 4 Tehnologii Java

1. Fie următoarea declarație Java:

```
public private int h;
```

Care afirmații sunt adevărate:

- A. Variabila h va fi accesată în mod public, deoarece se ia în considerare primul modificador de acces;
- B. Variabila h va fi accesată în mod private, deoarece se ia în considerare ultimul modificador de acces;

**C. Va fi eroare la compilare deoarece o variabilă nu poate fi în același timp accesată public și private;**

D. Nici una din variantele de mai sus;

2. Fie următorul cod Java:

```
int x=0;
if (Double.isInfinite(2/x))
    System.out.println("Infinit");
else
    System.out.println("2/0");
```

Ce puteți spune despre acest cod, dacă este integrat în cadrul unui program Java?

A. Va produce eroare la compilare din cauza împărțirii la 0;

**B. Va produce eroare la execuție din cauza împărțirii la 0 (se aruncă o excepție: "ArithmeticExpetion");**

C. Codul este corect și va afișa Infinit;

D. Codul este corect și va afișa NaN;

3. Fie următorul program Java:

```
public class Program{
    static void f(int k){
        switch(k){
            default: System.out.print("i "); break;
            case 1: System.out.print("1 "); break;
            case 2: case 3: System.out.print("23 "); break;
            case 4: case 5: System.out.print("45 ");
        }
    }
    public static void main(String []args){
        for(int i=0;i<6;i++)
            f(i);
    }
}
```

Care afirmații sunt adevărate?

A. Programul produce eroare la compilare;

**B. Programul se compilează și la execuție afișează  
i 1 23 23 45 45 ;**

C. Programul se compilează și la execuție afișează  
i 1 23 45 ;

D. Programul se compilează și la execuție afișează  
i 1 23 23 45 45 i;

4. Fie următorul cod Java:

```
byte b=-7 >>> 1;
System.out.println(b);
```

Ce se poate spune despre acest cod, dacă este integrat într-un program Java?

**A. Va produce eroare la compilare;**

B. Va produce eroare la execuție;

C. Programul se compilează și la execuție afișează -3;

D. Programul se compilează și la execuție afișează -4;

5. Ce puteți afirma despre următorul program Java?

```
public class Static1{
public static void main(String []args){
Static2 a=new Static2();
Static2 b=new Static2();
System.out.print("a.x= "+ a.x);
a.x=100; b.x=200;
System.out.print("a.x= "+a.x);
}
}
class Static2{
static int x=0;
Static2()
{
x++;
};
}
```

**A. Afășează:**

**a.x=2 a.x=200;**

B. Afășează:

a.x=0 a.x=100;

C. Afășează:

a.x=1 a.x=100;

D. Programul nu este corect deoarece asignarea lui x, conform obiectului b, este ilegală în Java;

6. Ce se va afișa la execuția următorului program Java?

```
interface I1{
float x=2.3f;
}
public class Test implements I1{
public static void main(String [] args){
System.out.print(x+" ");
x=6.7f;
System.out.print(x);
}
}
```

**A. Va apărea eroare la compilare deoarece valoarea variabilei x nu se mai poate modifica;**

B. La execuție se va afișa:

2.3f 6.7f;

C. La execuție se va afișa:

2.3f 2.3f;

D. La execuție se va afișa:

2.3 6.7;

7. Ce puteți spune despre următorul program Java?

```
class C1{
int x=1;
void f(int x){
this.x=x;
}
```

```

int getX_C1(){
return x;
}
}
class C2 extends C1{
float x=5.0f;
int f(int x){
super.f((int)x);
}
float getX_C2(){
return x;
}
}
public class Subiect9{
public static void main(String []args){
C2 obiect = new C2();
obiect.f(4);
System.out.print(obiect.getX_C2() + " ");
System.out.println(obiect.getX_C1());
}
}

```

- A. Programul este corect și va afișa la execuție 5 4;
- B. Programul este corect și va afișa la execuție 4.0 4;
- C. Va apărea eroare la compilare deoarece în clasa C2 s-a suprascris greșit atributul x din clasa C1;
- D. Va apărea eroare la compilare deoarece metoda suprascrisă f() din clasa C2 întoarce un tip diferit de void;**

8. Ce puteți spune despre următorul program Java?

```

public class Test{
public static void main(String []args){
C1 obiect =new C1();
obiect.f(4,3);
}
}
class C1{
public void f(int xx, final int yy){
int a=xx+yy;
final int b=xx-yy;
class C2{
public void g(){
System.out.print("a= "+a);
System.out.print(", b= "+b);
}
}
C2 obiect2 = new C2();
obiect2.g();
}
}

```

- A. Programul este corect și va afișa la execuție a=4, b=3;
- B. Va apărea eroare la compilare, deoarece clasa C2 nu poate fi definită în metoda f() din clasa C1;
- C. Va apărea eroare la compilare deoarece în metoda g() nu putem accesa variabila locală a din metoda f());**
- D. Va apărea eroare la compilare deoarece nu se creează în clasa Test un obiect de tip C1.C2 ;

9. Un fir de execuție poate intra în starea "blocat" (blocked) astfel?
- A. Prin apelul metodei `sleep()`;**
  - B. Automat de către sistemul de operare;
  - C. Prin apelul metodei `block()`;
  - D. Prin apelul metodei `wait()`;**
10. Care dintre următoarele propoziții referitoare la metodele unui applet sunt adevărate?
- A. Trebuie să redefinim măcar o metodă altfel obținem eroare la compilare;
  - B. Sunt apelate automat de navigatorul Web;**
  - C. Pot fi apelate direct de către utilizator;**
  - D. Nu se pot declara noi metode;
11. Care dintre gestionarii de poziționare (Layout Managers) de mai jos pot fi utilizați pentru applet-uri?
- A. `GridBagLayout`;**
  - B. `TableLayout`;
  - C. `DefaultLayout`;
  - D. `FlowLayout`;**
12. Ce metode trebuie definite pentru a putea desena pe suprafața applet-ului?
- A. Nu trebuie definită nici o metodă;
  - B. `update()`;
  - C. `paint()`;**
  - D. `repaint()`;
13. Care sunt deosebirile dintre Swing și AWT?
- A. Componentele Swing sunt scrise în totalitate în Java, pe când în AWT, componentele sunt scrise folosind cod nativ;**
  - B. Componentele AWT au vizualizarea dependentă de sistemul de operare, iar în Swing componentele pot avea o aceeași vizualizare, indiferent de sistemul de operare;**
  - C. În sistemele de operare Unix componentele AWT nu sunt vizibile, pe când cele din Swing da;
  - D. Fiecare componentă AWT are o componentă corespondentă în Swing;
14. Care din fragmentele de cod Java de mai jos dau un buton unui container de baza JFrame?
- A. `JFrame f= new JFrame();`  
`JPanel p = (JPanel)f.getContentPane();`  
`p.add(new JButton("Buton"));` ;**
  - B. `JFrame f= new JFrame();`  
`f.getContentPane().add(new JButton("Buton"));`**
  - C. `JFrame f= new JFrame();`  
`f.add(new JButton("Buton"));`
  - D. `JFrame f=new JFrame();`  
`JButton b=new JButton("Buton");`  
`f.add(b);`
15. Fie următoarea secvență de cod:
- ```
JPanel p =new JPanel();
JLabel e = new JLabel("Eticheta:");
e.setDisplayedMnemonic('E');
p.add(e, BorderLayout.EAST);
JTextField t = new JTextField(7);
e.setLabelFor(t);
p.add(t, BorderLayout.WEST);
```

Ce se va întâmpla la apăsarea combinației de taste: Alt + E?

- A. Eticheta e va deține focusul;
  - B. Câmpul t va deține focusul;**
  - C. Nici o componentă nu va deține focusul;
  - D. Panoul p va deține focusul;
16. Care dintre următoarele afirmații referitoare la componenta grafică *JTable* sunt adevărate?
- A. Întotdeauna o componentă *JTable* are asociată un model de date, chiar dacă nu specificăm explicit acest lucru;**
  - B. Prin modificările efectuate asupra datelor unui model, se va actualiza automat și vizualizarea tabelului;**
  - C. Pentru a crea o tabelă folosind modelul *AbstractTableModel*, este îndeajuns să suprascriem metodele *getRowCount()* și *getColumnCount()*;
  - D. Modelul de date *AbstractTableModel* ține datele întotdeauna într-un Vector având elemente Vector;
17. *JDesktopPane* este un exemplu de:
- A. Container pentru *JInternalFrame*;**
  - B. Subclasă pentru *JLayeredPane*;**
  - C. Subclasă pentru *JInternalFrame*;
  - D. Componentă atomică simplă;
18. Fie următorul program Java:

```
public class Afisare {
    public static void main (String[] args) {
        for (int i = 0; i < args.length; i++)
            System.out.println(args[i]);
    }
}
```

Un apel de genul `java Afisare "Hello Java"`; va produce următorul rezultat:

- A. Hello Java;**
  - B. Hello  
Java;
  - C. Programul este incorect, deoarece nu sunt prezente argumentele;
  - D. NaN;
19. Fie următorul program Java:

```
public class Program{
    static void f(int k){
        switch(k){
            default: System.out.print("i "); break;
            case 1: System.out.print("1 "); break;
            case 2: case 3: System.out.print("21 "); break;
            case 4: case 5: System.out.print("26 ");
        }
    }
    public static void main(String []args){
        for(int i=0;i<6;i++)
            f(i);
    }
}
```

Care afirmații sunt false?

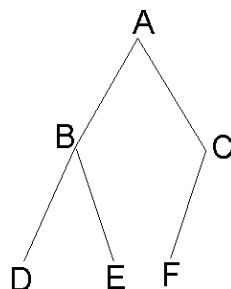
- A. Eroare la compilare;**
- B. Programul se compilează și la execuție afișează  
i 1 21 21 26 26 ;
- C. Programul se compilează și la execuție afișează  
i 1 21 26 ;**
- D. Programul se compilează și la execuție afișează  
i 1 21 21 26 26 i;**
20. Se utilizează *protected* în Java pentru date și metode la care nu este necesar:
- A. Să facem o inițializare;
- B. Să le accesăm în subclase;
- C. Ca utilizatorul să aibă acces;
- D. Accesul direct atunci când clasa este utilizată dar care prezintă interes atunci când cineva creează o subclasă a acesteia ca parte a unui pachet diferit, în vederea extinderii ei;**
21. O subclasă a unei clase abstracte poate fi instanțiată numai dacă:
- A. Se folosește cuvântul cheie *abstract*;
- B. Suprascrie fiecare metodă declarată abstractă în superclasa sa, și furnizează implementări pentru toate acestea;**
- C. Se folosește moștenirea multiplă;
- D. O subclasă abstractă nu poate fi instanțiată;
22. Prin modalitatea sa de tratare a excepțiilor, Java are următoarele avantaje față de mecanismul tradițional de tratare a erorilor:
- A. Există o metodă care se ocupă cu acest lucru;
- B. Separarea codului pentru tratarea unei erori de codul în care ea poate să apară;**
- C. Propagarea unei erori până la un analizor de excepții corespunzător;**
- D. Gruparea erorilor după tipul lor;**
23. Metodele care sunt apelate uzual pentru un obiect de tip *excepție* sunt definite în clasa *Throwable* și sunt:
- A. Declarate cu modificatorul de acces *private*;
- B. dinamice;
- C. publice;**
- D. excepții;
24. Un fir de execuție poate intra în starea de *ready* astfel:
- A. Prin apelul metodei *sleep()*;
- B. Automat de către sistemul de operare;
- C. Prin apelul metodei *join()*;
- D. Niciodată;
25. Când browser-ul întâlnește tag-ul `<APPLET >`, rezervă o zonă pentru afișare cu dimensiunile specificate de parametrii *WIDTH*, *HEIGHT* și:
- A. Se instalează un manager de securitate, adică un obiect de tip *SecurityManager* care va monitoriza activitatea metodelor appletului, aruncând excepții de tip *SecurityException*;
- B. Încarcă codul compilat al applet-ului cu numele specificat de parametrul *CODE*;
- C. Creează o instanță a clasei *Applet* după care apelează metodele *init()* și *start()*;**
- D. Se deschid mai multe procese pe mașina client;



26. Care dintre următoarele coduri nu reprezintă arhivarea fișierelor unui applet?
- A. **jar cvf arhiva.jar ClasaPrincipala.java imagine.jpg;**
  - B. jar cvf arhiva.jar \*.class \*.jpg \*.au;
  - C. **jar cvf arhiva.jar \*.class \*.jpg \*.au;**
  - D. jar cvf arhiva.jar ClasaPrincipala.class AltaClasa.class imagine.jpg sunet.au
27. În care din exemplele de mai jos se folosește corect variabila *iLocation*?
- A. **tabbedPanel.insertTab( "Inserted Page", new ImageIcon( "image.gif" ), pagePanel,"My tooltip text",iLocation );**
  - B. JFrame f= new JFrame();  
f.getContentPane().add(new JButton("Buton", iLocation));
  - C. **tabbedPanel.removeTabAt( iLocation );**
  - D. JFrame f=new JFrame();  
JButton b=new JButton("Buton");  
f.add(b, iLocation);
28. Ce rezultă din următorul fragment de cod Java?
- ```
int x=1;
String []names={"Fred","Jim","Sheila"};
names[--x]+=".";
for(int i=0;i<names.length;i++)
    System.out.println(names[i]);
```
- A. Output-ul include Sheila.;
  - B. **Output-ul include Fred.;**
  - C. Output-ul include Jim.;
  - D. Nimic din cele de mai sus;
29. Applet-urile se diferențiază de aplicațiile Java standard prin:
- A. **Restricțiile impuse de necesitatea asigurării unui anumit nivel de securitate și faptul că nu au o metodă *main()*;**
  - B. Faptul că trebuie să suprascrise toate metodele: *init()*, *start()*, *stop()*, *pause()* și *destroy()*;
  - C. Faptul că dețin un instrument cu ajutorul căruia se crează relații de moștenire;
  - D. Faptul că dețin un instrument care desemnează meniurile din cadrul unei forme;
30. Declararea constructorilor trebuie să țină cont de:
- A. relația de moștenire dintre clase;
  - B. **numele constructorului, care trebuie să fie identic cu numele clasei;**
  - C. comportamentul obiectelor pe care le instanțiază;
  - D. o metoda prin care poate fi accesat de toate tipurile din Java sau de tipuri mostenite din tipul care conține membrul in discutie;

## 5 Algoritmica grafurilor

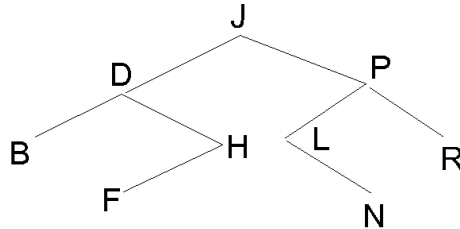
1. O procedură ce parcurge arborele binar



în *postordine* va afișa:

- A. A B C D E F
- B. A B D E C F
- C. D B E A F C
- D. D E B F C A**
- E. D E F B C A

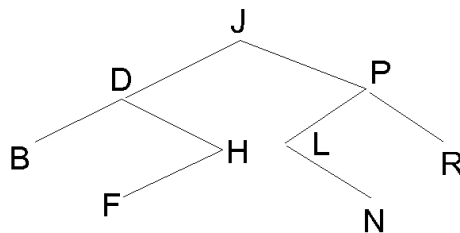
2. O procedură ce parcurge arborele binar



în *preordine* va afișa:

- A. J D P B H L R F N
- B. J D B H F P L N R**
- C. N F R L H B P D J
- D. B D H F N L P R J
- E. B D F H L J N P R

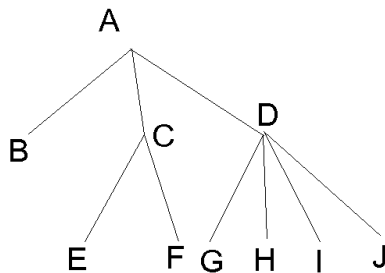
3. O procedură ce parcurge arborele binar



în *inordine* va afișa:

- A. B H F D L N P R J
- B. B D H F N L P R J
- C. N F R L H B P D J
- D. J D P B H L R F N
- E. B D F H J L N P R**

4. O procedură ce parcurge arborele binar



în *A-preordine* va afișa:

- A. A B C E F D G H I J**

- B. A B C D E F G H I J
- C. B E F C G H I J D A
- D. B A E C F G D H I J
- E. B E F G H I J C D A

5. Care este numărul maxim de componente conexe pe care le poate avea un graf neorientat cu 20 de noduri și 12 muchii?
- A. 6
  - B. 12
  - C. 10
  - D. 15**
  - E. 18
6. Câte grafuri neorientate, distincte, cu 4 vârfuri se pot construi? Două grafuri se consideră distincte dacă matricele lor de adiacență sunt diferite.
- A.  $4^6$
  - B.  $2^6$**
  - C.  $6^4$
  - D. 4
  - E.  $2^{10}$
7. Într-un graf neorientat cu 10 muchii, fiecare nod are gradul un număr nenul. Doar trei dintre noduri au gradul un număr par, restul nodurilor având gradele numere impare. Care este numărul maxim de noduri pe care poate să le aibă graful?
- A. 14
  - B. 10
  - C. 17**
  - D. 16
  - E. 19
8. Determinați numărul de frunze ale arborelui cu rădăcină descris prin următorul vector **tata**:  $(6, 5, 5, 2, 0, 3, 3, 3, 8, 7, 7)$ ?
- A. 1
  - B. 2
  - C. 4
  - D. 5**
  - E. 7
9. Care dintre următoarele valori pot reprezenta gradele nodurilor unui graf neorientat cu 6 noduri?
- A.  $(3, 2, 2, 2, 3, 3)$
  - B.  $(4, 2, 2, 2, 3, 2)$
  - C.  $(5, 2, 2, 2, 0, 3)$
  - D.  $(5, 2, 2, 2, 1, 2)$**
  - E.  $(5, 2, 3, 2, 1, 2)$
10. Se consideră graful neorientat cu mulțimea vârfurilor  $\{1, 2, 3, 4, 5, 6\}$  și mulțimea muchiilor  $\{[1, 2], [2, 3], [3, 4], [3, 5], [4, 5], [1, 3], [2, 6], [2, 4], [4, 6]\}$ . Care este numărul minim de muchii ce trebuie eliminate (care sunt aceste muchii) astfel încât graful parțial obținut să nu mai fie conex?
- A. 1
  - B. 3
  - C. 2**

- D. 5
- E. nici una

11. Se consideră graful neorientat  $G$  cu 8 noduri, care are următoarele proprietăți:

- I. suma gradelor tuturor nodurilor este 12;
- II. graful are exact 3 noduri cu gradul 1.

Care este numărul maxim de noduri de grad 0 ale grafului  $G$ ?

- A. 1
- B. 4
- C. 2**
- D. 3
- E. 0

12. Se consideră graful neorientat cu 80 de noduri și 3160 de muchii. Care este numărul de muchii ce pot fi eliminate astfel încât graful parțial obținut să devină arbore?

- A. 80
- B. 3160
- C. 3081**
- D. 6320
- E. 2450

13. Se consideră graful orientat  $G$  reprezentat prin listele de adiacență alăturate. Care este lungimea maximă a unui drum elementar din acest graf? Să se specifice arcele ce compun un drum cu aceste proprietăți.

```
1 2 6 5
2 3
3 1
4 6
5 6
6 2
```

- A. 1
- B. 7
- C. 3
- D. 4
- E. 5**

14. Pentru care dintre următorii arbori cu rădăcină, fiecare având 9 noduri, numerotate de la 1 la 9, memorai cu ajutorul vectorilor **tata**, nodul 3 are cei mai mulți descendenți?

- A. (2, 0, 2, 3, 2, 3, 4, 4, 3)
- B. (3, 3, 4, 0, 2, 3, 4, 4, 4)
- C. (4, 2, 4, 0, 3, 3, 3, 3, 3)**
- D. (0, 1, 1, 3, 4, 3, 4, 4, 3)
- E. (0, 1, 2, 3, 4, 5, 6, 7, 8)

15. Care dintre următoarele proprietăți este adevărată pentru un graf orientat cu  $n$  vârfuri și  $n$  arce ( $n > 3$ ) și care prezintă un circuit de lungime  $n$ :

- A. există un vârf cu gradul intern  $n - 1$ .
- B. pentru orice vârf, gradul intern și gradul extern sunt egale.**
- C. graful nu are drumuri de lungime strict mai mare decât 2.
- D. gradul intern al oricărui vârf este egal cu 2.

E. dacă notăm graful cu  $G = (V, E)$ ,  $|V| = n$ , atunci  $|E| = n - 1$ .

16. Care este ordinea de parcurgere a nodurilor grafului din figura 1, dacă se folosește metoda de vizitare în lățime (BFS) pornind din nodul 5?

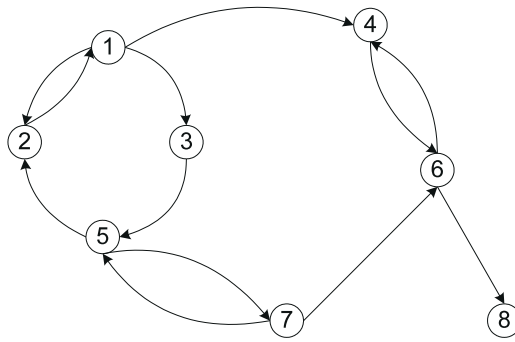


Figure 1: Un exemplu de graf orientat

- A. 5, 2, 1, 3, 4, 7, 6, 8
  - B. 5, 2, 1, 3, 5, 7, 6, 8
  - C. 5, 7, 6, 4, 8, 2, 1, 3
  - D. 5, 2, 7, 1, 6, 3, 4, 8**
  - E. 5, 7, 2, 6, 1, 8, 4, 3
17. Care este ordinea de parcurgere a nodurilor grafului din figura 2, dacă se folosește metoda de vizitare în adâncime (DFS) pornind din nodul 3?

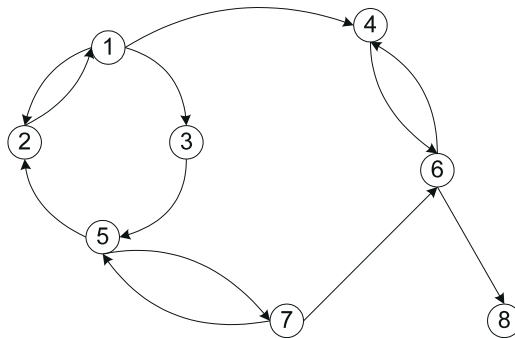


Figure 2: Un exemplu de graf orientat

- A. 3, 5, 7, 6, 4, 8, 2, 1
  - B. 3, 5, 2, 7, 1, 6, 4, 8
  - C. 3, 5, 2, 1, 4, 7, 6, 8
  - D. 3, 5, 2, 1, 4, 6, 8, 7**
  - E. 3, 5, 2, 7, 1, 4, 6, 8
18. Determinați ordinul și dimensiunea grafului din figura 3?

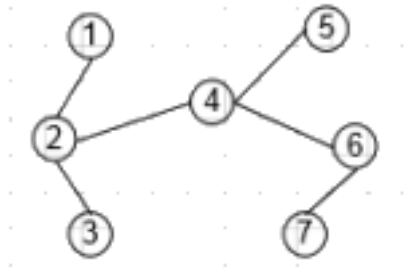


Figure 3: Un exemplu de graf neorientat

- A. 5,6
  - B. 8,5
  - C. 7,6**
  - D. 7,7
  - E. 6,7
19. Dacă pentru un arbore binar avem un număr de  $N$  muchii, să se determine numărul de muchii critice ale arborelui?
- A.  $N - 1$
  - B.  $N + 1$
  - C.  $N/2$
  - D.  $(N + 1)/2$
  - E.  $N$**
20. Dacă avem un arbore binar plin cu 9 nivele, să se determine numărul de vârfuri ce se află pe nivelul cu numărul 5?
- A. 15
  - B. 31
  - C.  $2^5$
  - D.  $2^4$**
  - E. 14
21. Având un arbore binar reprezentat prin următoarea expresie cu paranteze:  $1(2(3(0, 4(0, 0)), 5(6(0, 0), 7(0, 0))), 8(0, 9(0, 0)))$ . Să se determine numărul de frunze ale grafului.
- A. 4**
  - B. 3
  - C. 5
  - D. 9
  - E. 6
22. Având graful ponderat din figura 4, să se determine un arbore de acoperire de cost minim. Vom considera nodul  $A$  ca nod de plecare. Care este ordinea de parcurgere a nodurilor?

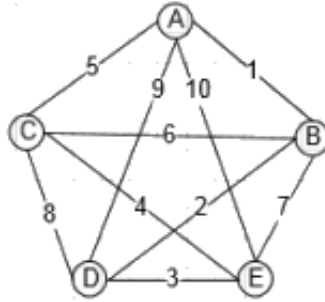


Figure 4: Un exemplu de graf ponderat

- A.  $A, C, B, D, E$ ;
- B.  $A, B, D, E, C$ ;**
- C.  $A, D, E, C, B$ ;
- D.  $A, B, D, C, E$ ;
- E.  $A, B, E, B, D$ ;

## 6 Metode de analiză a algoritmilor

1. Care este expresia timpului de execuție  $T(n)$  al următorului algoritm în cazul cel mai favorabil, în cazul mediu și în cazul cel mai defavorabil, unde  $n$  notează dimensiunea setului de intrare, iar  $k, k_1, \dots, k_8$  - orice constantă pozitivă:

```

procedure ALGORITM( $v, n$ )
  for  $i \leftarrow 2, n$  do
     $key \leftarrow v[i]$ 
     $j \leftarrow i - 1$ 
    while ( $j > 0$ ) and ( $v[j] > key$ ) do
       $v[j+1] \leftarrow v[j]$ 
       $j \leftarrow j - 1$ 
    end while
     $v[j] \leftarrow key$ 
  end for
end procedure

```

- A.  $k, k_1n, k_2n + k_3$
  - B.  $k_1n + k_2, k_3n + k_4, k_5n + k_6$
  - C.  $k, k_1n + k_2, k_2n^2 + k_3n + k_4$
  - D.  $k_1n + k_2, k_3n + k_4, k_5n^2 + k_6n + k_7$
  - E.  $k, k_1n^2 + k_2n + k_3, k_4n^2 + k_5n + k_6$
  - F.  $k_1n + k_2, k_3n^2 + k_4n + k_5, k_6n^2 + k_7n + k_8$**
2. Determinați în care caz timpii de execuție  $T_1(n)$  și  $T_2(n)$  au același ordin de creștere, unde prin  $n$  notăm dimensiunea datelor de intrare iar  $k$  notează o constantă oarecare pozitivă:
    - A.  $\lim_{n \rightarrow \infty} \frac{T_1(n)}{T_2(n)} = 0$
    - B.  $\lim_{n \rightarrow \infty} \frac{T_1(n)}{T_2(n)} = \infty$
    - C.  $\lim_{n \rightarrow \infty} \frac{T_1(n)}{T_2(n)} = k$**
  3. Pentru  $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \infty$ , care din următoarele afirmații sunt adevărate?
    - A.  $f(n) \in \omega(g(n))$**
    - B.  $f(n) \in \Theta(g(n))$

C.  $f(n) \in \Omega(g(n))$

4. Pentru  $f(n) \in \Theta(n)$  și  $g(n) \in \Omega(n)$  atunci

A.  $f(n)g(n) \in \Omega(n^2)$

B.  $f(n)g(n) \in \Theta(n^2)$

C.  $f(n)g(n) \in O(n^2)$

5. Pentru  $f(n) = 6n^3 + 14n^2 - 8n + 4$  și  $g(n) = 2n^3$  care din următoarele afirmații sunt adevărate?

A.  $f(n), g(n) \in O(2n^3)$

B.  $f(n), g(n) \in O(n^3)$

C.  $f(n) \in O(6n^3)$  și  $g(n) \in O(2n^3)$

6. Pentru  $f(n) \in \Omega(n)$  și  $g(n) \in O(n^2)$  atunci

A.  $\frac{f(n)}{g(n)} \in \Omega(n)$

B.  $\frac{f(n)}{g(n)} \in O(n)$

C.  $\frac{f(n)}{g(n)} \in \Theta(n)$

7. Pentru  $f(n) \in O(h(n))$  și  $h(n) \in O(g(n))$  atunci

A. funcția  $f(n)$  este echivalentă cu  $g(n)$

B.  $f(n) \in O(g(n))$

C.  $f(n) \in \Theta(g(n))$

8. Pentru  $n$  - număr natural nenul și  $f(n) = 1 + 2 + \dots + n$  atunci

A.  $f(n) \in O(n)$

B.  $f(n) \in O(n^2)$

C.  $f(n) \in \Theta(n^2)$

9. Care este ordinul de complexitate al algoritmului de căutare binară?

```
procedure BINSEARCH(start, finish)
```

```
  if start=finish then
```

```
    return start
```

```
  end if
```

```
  mij ← (start + finish + 1)/2
```

```
  if x ≤ v[mij] then
```

```
    binsearch(start, mij)
```

```
  else
```

```
    binsearch(mij+1, finish)
```

```
  end if
```

```
end procedure
```

A.  $\Omega(\log n)$

B.  $O(\log n)$

C.  $O(n)$

10. Care este ordinul de complexitate al algoritmului de ridicare la putere al unui număr  $x^n$ ?

A.  $\Theta(n)$

B.  $\Theta(\log n)$

C.  $\Theta(x^n)$

11. Care este ordinul de complexitate al funcției  $f(n)$  cu relația de recurență  $t(n) = 2t(\sqrt{n}) + \log n$ ?

A.  $f(n) \in O(\log n \log(\log n))$

B.  $f(n) \in O(\log n)$

C.  $f(n) \in O(n^{\frac{1}{2}})$



12. Care este ordinul de creștere al unui algoritm având timpul de execuție de forma următoare:

$$t(n) = \begin{cases} t(n-1) + n, & n > 0 \\ 0, & n = 0 \end{cases}$$

- A.  $f(n) \in O(n^2)$
- B.  $f(n) \in \Theta(n^2)$
- C.  $f(n) \in O(n)$

13. Care este ordinul de creștere al unui algoritm având timpul de execuție de forma următoare:

$$t(n) = \begin{cases} t(n/2), & n > 1 \\ 1, & n = 1 \end{cases}$$

- A.  $f(n) \in O(\log_2 n)$
- B.  $f(n) \in O(\log n)$
- C.  $f(n) \in O(\lg n)$

14. Care este ordinul de creștere al algoritmului pentru problema Turnurilor din Hanoi având timpul de execuție de forma

$$t(n) = \begin{cases} 2t(n-1) + 1, & n > 1 \\ 1, & n = 1 \end{cases}$$

- A.  $f(n) \in O(n)$
- B.  $f(n) \in O(2^n)$
- C.  $f(n) \in O(\log n)$

15. Care este ordinul de creștere al unui algoritm având timpul de execuție de forma următoare:

$$t(n) = \begin{cases} 2t(n/2) + 1, & n > 2 \\ 1, & n = 2 \end{cases}$$

- A.  $f(n) \in O(n)$
- B.  $f(n) \in O(2^{\log n})$
- C.  $f(n) \in O(\log n)$

16. Care este ordinul de complexitate al algoritmului mergesort având relația de recurență  $t(n) = 2t(n/2) + \Theta(n)$ ?

- A.  $f(n) \in \Theta(n \log n)$
- B.  $f(n) \in \Theta(n)$
- C.  $f(n) \in \Theta(n + n \log n)$

17. Să se indice ordinul de creștere al funcției  $f(n)$  cu următoarea relație de recurență:

$$t(n) = \begin{cases} t(n-1) + 1, & n > 0 \\ 0, & n = 0 \end{cases}$$

- A.  $f(n) \in O(n^2)$
- B.  $f(n) \in \Theta(n)$
- C.  $f(n) \in O(\log n)$

18. Să se indice ordinul de creștere al funcției  $f(n)$  cu următoarea relație de recurență:

$$t(n) = \begin{cases} n(t(n-1) + 2), & n > 1 \\ 0, & n = 1 \end{cases}$$

- A.  $f(n) \in O(n!)$
- B.  $f(n) \in O(n^2)$
- C.  $f(n) \in O(n)$

19. Să se indice ordinul de creștere al funcției  $f(n)$  cu următoarea relație de recurență:

$$t(n) = \begin{cases} 2t(n/2) + n, & n > 1 \\ 1, & n = 1 \end{cases}$$

- A.  $f(n) \in \Theta(n)$
- B.  $f(n) \in \Theta(n + n \log n)$
- C.  $f(n) \in \Theta(n \log n)$

20. Care din următoarele interpretări fac adevărată formula  $F$  ilustrată în graficul din Figura 5?

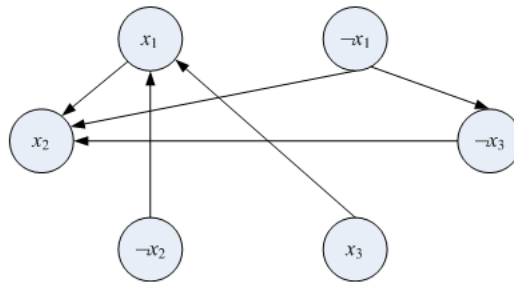


Figure 5: Graficul  $G(F)$

- A.  $x_1 = true, x_2 = true, x_3 = true$
- B.  $x_1 = false, x_2 = true, x_3 = false$
- C.  $x_1 = true, x_2 = false, x_3 = true$