Răspunsurile corecte cu explicații

1. a

În C/C++ orice valoare nenulă este asimilată cu valoarea logică ADEVĂRAT (TRUE) care are ca valoare numerică valoarea 1 și orice valoare nulă este asimilată cu valoarea logică FALS (FALSE) care are ca valoare numerică valoarea 0.

2. c

Pentru primul pas al instrucțiunii for, variabila i va avea valoarea 10, deci cele trei instrucțiuni vor avea următorul efect:

- s va lua valoarea 10
- se va afișa valoarea 10, după care i va fi decrementat, deci va lua valoarea 9
- s va fi egal cu 10 - 9, deci va avea valoarea 1

Acești trei pași vor fi reluați pentru valorile lui i: 8 6 4 2. La fiecare trecere prin for, variabila s va fi incrementată.

3. c

Pentru a rezolva această problemă, trebuie verificat pseudocodul pentru toate variantele de răspuns.

4. c

Având în vedere că variabila k este statică, la revenirea din fiecare apelul se va afișa valoarea acesteia care este 0 (zero).

5. a

Instrucțiunea switch verifică valoarea lui x și face saltul la instrucțiunile cu eticheta respectivă. După execuția instrucțiunilor corespunzătoare etichetei la care s-a făcut saltul, execuția programului se continuă cu instrucțiunile imediat următoare (dacă nu există *break* și iese din *switch* la întâlnirea instrucțiunii *break*.

6. b

```
#include <iostream>
#include <cstring>
using namespace std;
char x[]={'A','N','A','L','I','Z','A','Q','R','E','Z','U','L','T','A','T','Q'};
int n;
void criptare (int p)
{
    for(int i=0;i<n;i++)
        x[i]=(('Z'-x[i])%p+'a');
 }

void decriptare (int p)
{
    for(int i=0;i<n;i++)
        x[i]=(('z'-x[i])+'A');
}

void afis()
{
    for(int i=0;i<17;i++)
    cout<<x[i];
    cout<<endl;
}

int main()
{
    n=strlen(x);
```

```
    afis();
    criptare(26);
    afis();
    decriptare(26);
    afis();
    return 0;
}
```

7. b

Prin parcurgerea codului se identifică următoarele aspecte: vectorul este parcurs începând cu o anumită poziție până la ultimul element (inclusiv). De la această poziție până la finalul vectorului se mută elementele de pe pozitia $i+1$ pe poziția i. La final, numărul de elemente ale vectorului se decrementează. Toate aceste operații elimină din vector elementul de pe poziția de start a instrucțiunii *for*.

8. b

Șirul de caractere este procesat de la final la început în grupuri de trei caractere. Caracterele din fiecare grup sunt puse într-un șir temporar în ordine inversă, iar șirul temporar este afișat doar dacă este înaintea șirului *inf* din punct de vedere lexicografic. Ultima grupare de trei caractere ar fi fost formată doar din caracterul A, dar aceasta nu este procesată pentru că i ajunge 0 și se iese din bucla *while*.

9. c

Raspuns c) Matricea A Orice element A[l][m] va fi interschimbat de două ori, odata pentru i = l si j = m si apoi pentru i = m si j = l. Swap de două ori înseamnă ca matricea nu se schimbă .

10. c

Algoritmul de căutare a existenței a cel putin unei perechi de numere din tabloul v a căror diferența este d, utilizeaza două variabile index inițializate cu 0 respectiv 1. Aceste variabile vor fi incrementate funcție de valoarea de adevăr a expresiei E. Dacă diferența dintre elementele indicate este strict mai mică decât d este incrementat indexul j altfel indexul i (daca nu cumva diferența este d). A nu se uita faptul elementele tabloului sunt în ordine strict crescătoare.

11. c

Algoritmul de căutare utilizează două variabile de tip index inițializate cu 0 și n-1 (elementele de capăt ale tabloului). Sunt comparate elementele de index i și j și corespunzător se incrementeaza i sau j în funcție de valoarea de adevăr a expresiei (v[i] <= v[j]. Algoritmul se opreste atunci când i == j. Elementul maxim este cel de pe poziția i (sau j).

12. b

Deoarece, o dată ce melcul a ieșit din fântână, nu mai poate să cadă înapoi, se poate considera că fântâna are $(N - Y)$ metri, iar melcul urcă zilnic $(X - Y)$ metri.

13. b

Pseudocodul calculează factorialul numărului x. Acest lucru se poate observa calculând $f(2)$ și $f(3)$. Deoarece $6! = 720$, valoarea 719 se poate obține prin instrucțiunea $f(6) - 1$.

14. b

1. Secvenţele sunt asezate în ordinea crescătoare a dimensiunilor lor.

2. Se aleg pentru interclasare primele două secvenţe. Secvenţa rezultată inlocuieşte cele două secvenţe interclasate.

3. Se repetă paşii 1 si 2 până se obţine o singura secvenţă.

15. b

Tatăl lui Cristian aşează cutiile în ordinea 3,5,4,1,2 (15/3>40/10>18/6>10/5>8/8). Alege cutiile 3,5,4 şi cere desfacerea cutiei 1.

Observație. Nu cred că trebuie să explic metoda greedy. Cei care cunosc metoda, vor rezolva problema mai rapid. Ceilalți vor încerca toate variantele posibile.

16. d

$((y_s 10^{n/2} + y_d)$ şi $((x_s 10^{n/2} + x_d)$ sunt numere formate din n cifre. În varianta (d), $(x_s - x_d)$ şi $(y_d - y_s)$ sunt numere formate din $n/2$ cifre. Înmulţirile $x_s y_s$ şi $x_d y_d$ pot fi efectuate o singură dată.

17. c

Complementul unui graf G cu n noduri și m muchii: din graful complet (care conține aceleași noduri ca G), și care are $n * (n-1)/2$ muchii, se elimină cele m muchii din G. Prim urmare, numărul de muchii din complementul lui $G = n * (n-1)/2 - m$.

18. a

(a) corect, nr max de noduri pe nivelul $level = 2^{level-1}$; (b) greșit, nr max de frunze $= 2^{10-1} = 512$; (c) greșit, nr max noduri din arbore $= 2^{level} - 1 = 1023$; (d) greșit, $n_0 = n_2 + 1$ pt un arbore strict

19. a

Nodurile nu vor avea multe muchii - acestea se definesc doar între un triunghi și triunghiurile vecine din rețea. Lista de adiacență este mai potrivită, deoarece este mai eficientă pentru stocarea grafurilor rare și se evită re-alocarea matricei în cazul adăugării/ștergerii de noduri/muchii (graful este construit 'on-the'fly', pe masură ce se explorează scena).

20. a

Raspuns a) B1 : (1 + height(n->right)), B2 : (1 + max(h1,h2))

```
typedef int Tip_cheie;
struct Nod
    {
      Tip_cheie cheie;  //cheia este chiar valoarea
      Nod *stg, *drt;
    };
int Height(Nod *n)
{
    int h1, h2;
    if (n == NULL) return -1;
    if (n->stg == NULL)
        if (n->drt == NULL) return 0;
        else return (1 + Height(n->drt));// aici este B1
    else {
        h1 = Height(n->stg);
        if (n->drt == NULL) return (1 + h1);
        else
        {
            h2 = Height(n->drt);
            if(h1>=h2)  //aici este B2 din pseudocod
                return (1 + h1);
            else return (1+ h2);
        }
    }
}
void InsertN(Nod *&r, Tip_cheie k)  //pentru creare BST
{
   if(r==0) r=MakeNod(k);
   else
   {
      Nod *p, *p1;  p=r;
      while(p!=0)
      {
        p1=p;
```

```
            if(p->cheie > k)          p=p->stg;
            else
                if(p->cheie < k)      p=p->drt;
                else                        //nodul exista
                    { printf("nodul exista\n"); return;}
        }       // se iese cu p=0 si p1 indica nodul dupa care se insereaza
        if(k < p1->cheie)       p1->stg=MakeNod(k);
        else                    p1->drt=MakeNod(k);
    }
}
```

21. a

n(n-1)/2 muchii in total in graf => 15*14/2=105 muchii arbore binar cu 3 niveluri=> 14 muchii 105-14=91

22. b

Se printează reprezentarea binară in ordine inversa 10011

23. a

Plină: VÂRF = = n-1, Goală: VÂRF = = -1

Pentru push: stiva[++varf]=val Pentru pop: return stiva[- -varf]

24. a

Funcția dată este o funcție recursivă care se va opri când se va ajunge pe un element al matricii cu valoarea 1 sau când x sau y va lua o valoare mai mică sau egală cu 0. Funcția este apelată cu valorile 2, 2. Elementul din matrice de pe această poziție are valoarea 0. Se va afișa x = 2, y = 2 și se va apela funcția recursiv cu valorile x - 1, y - 1, respectiv 1, 1. Elementul de pe această poziție are tot valoarea 0, deci se va afisa x = 1, y = 1 și se va apela funcția recursiv cu valorile x - 1, y - 1, respectiv 0, 0. Acest element are valoarea 1, deci se va afișa x = 0, y = 0, dar nu se va mai apela funcția din nou, deoarece oricare dintre cele trei condiții din if va avea valoarea fals.

25. b

Pentru afișarea în ordinea descrescatoare a vârstei, lista liniară simplu înlanțuită trebuie parcursă în ordinea de la ultimul la primul element. (a) greșit - parcurge tabloul în ordine inversă alfabetic; (b) corect - parcurge recursiv lista - afișarea se face la revenirea din recursie, deci în ordine inversă a elementelor; (c) greșit - parcurge lista de la primul la ultimul element, deci în ordinea crescatoare a vârstei; (d) greșit - parcurge lista recursiv, cu afișarea înainte de intrarea în recursie, deci de la primul la ultimul element.