

Universitatea din București
Facultatea de Fizică – Admitere 2026
Test de informatică C/C++

Subiectele 1-10 au un singur răspuns corect
 N_1 = punctajul total de la subiectele 1-10 + 1p din oficiu
 N_2 = punctajul total de la subiectele 11-12 + 1p din oficiu
Nota finală: $N = 0.6 \times N_1 + 0.4 \times N_2$

1. (0.9p) Variabilele x , y și z pot memora valori întregi pozitive. Indicați $x < y$ && $3 * x == 5 * z$ și r crescător format cu valorile acestor variabile, dacă expresia C/C++ alăturată are valoarea 1.

A) x, y, z B) z, x, y C) z, y, x D) y, x, z

2. (0.9p) Utilizând metoda backtracking, se generează toate posibilitățile de a forma seturi de 5 particule distincte din mulțimea {electron, pozitron, neutron, proton, muon}, astfel încât în fiecare set protonul precede electronul și pozitronul. Două seturi sunt distincte dacă particulele sunt așezate în altă ordine. Primele cinci soluții generate sunt, în ordine, (neutron, proton, electron, pozitron, muon), (neutron, proton, electron, muon, pozitron), (neutron, proton, pozitron, electron, muon), (neutron, proton, pozitron, muon, electron), (neutron, proton, muon, electron, pozitron). Cea de-a șasea soluție este:

A) (pozitron, neutron, proton, muon, electron) B) (neutron, electron, proton, muon, pozitron)
C) (neutron, muon, proton, electron, pozitron) D) (neutron, proton, muon, pozitron, electron)

3. (0.9p) Se consideră fragmentul de cod alăturat, în care A este o matrice 5x5 având ca elemente numere întregi, iar $x \% y$ este restul împărțirii numărului natural x la numărul natural nenul y. Valoarea afișată este:

A) 24 B) 52 C) 4 D) 15

```
int A[5][5], tr=0, k, l;  
for(k=0;k<5;k++)  
{  
    for(l=0;l<5;l++)  
        A[k][l]=(2*k+k*l+1)%3;  
    tr += A[k][k];  
}  
printf("tr = %d\n\n",tr);
```

4. (0.9p) Se consideră subprogramul fun alăturat ($x \% y$ este restul împărțirii numărului natural x la numărul natural nenul y). Rezultatul executării instrucțiunii fun(11) este:

A) 1101 B) 00100 C) 1100 D) 110000

```
void fun(int x)  
{ if(x>=4)  
    { if(x%2==1) cout<<1;  
      fun((x-1)/2);  
    }  
    else cout<<0;  
    cout<<0;  
}
```

5. (0.9p) Indicați care dintre expresiile C/C++ de mai jos are valoarea 1 în cazul în care numărul natural memorat în variabila întregă x este divizibil cu 6 și cu 9:

A) $!(x \% 6 == 1) \ || \ (x \% 9 != 0)$ B) $(x \% 6 == 1) \ || \ (x \% 9 != 0)$
C) $(x \% 6 == 0) \ \&\& \ (x \% 9 != 0)$ D) $(x \% 6 == 1) \ \&\& \ (x \% 9 != 0)$

6. (0.9p) Se consideră subprogramul f alăturat ($x \% y$ este restul împărțirii numărului natural x la numărul natural nenul y). Rezultatul executării instrucțiunii:

f(7);
este:
A) 10 B) 010 C) 101 D) 1010

```
void f (int n)  
{ cout << n%2 | printf("%d",n%2;  
    if (n>=3) f(n-3);  
    return ;  
}
```

7. (0.9p) Se consideră graful orientat cu 6 vârfuri, numerotate de la 1 la 6, reprezentat prin listele de adiacență indicate alăturat. Două drumuri sunt distincte dacă diferă prin cel puțin un arc. Numărul de drumuri distincte de la vârful 2 la vârful 4 este:

A) 0 B) 2 C) 1 D) 3

1: 3, 5
2: 1, 6
3: 4
4: 5
5: listă vidă

8. (0.9p) Se consideră funcția `foo()` definită alăturat. Secvența de cod:

```
char txt[32];
strcpy(txt, "polariton");
foo(txt);
```

are ca efect afișarea textului:

- A) plrtn B) notiralop
C) POLARITON D) nu este nimic afișat

```
void foo(char str[])
{
    int i, j, len = strlen(str);
    for (i = 0; i < len; i++)
    {
        if (
            str[i] == 'a' || str[i] == 'e' ||
            str[i] == 'i' || str[i] == 'o' ||
            str[i] == 'u' || str[i] == 'A' ||
            str[i] == 'E' || str[i] == 'I' ||
            str[i] == 'O' || str[i] == 'U'
        )
        {
            for (j = i; j < len; j++) {
                str[j] = str[j + 1];
            }
            i--;
            len--;
        }
        str[len + 1] = '\0';
    }
    printf("%s\n", str);
    return ;
}
```

9. (0.9p) Frunzele arborelui cu rădăcină, având 8 noduri, numerotate de la 1 la 8, reprezentat prin vectorul "de tați" (5,0,5,2,2,4,1,2) sunt:

- A) 3,6,7,8 B) 1,8,4 C) 2,7 D) 1,6,7,8

10. (0.9p) Se consideră funcția `subst()` definită alăturat. Secvența de cod:

```
long nr = 287452;
printf("subst(%ld) = %ld\n", nr, subst(nr));
```

are ca efect afișarea textului:

- A) `subst(nr) = -1L`
B) `subst(287452) = 28745`
C) `subst(287452) = 254782`
D) `subst(287452) = 288462`

```
long subst(long n)
{
    if(n<0 || n>1000000)
        return -1L;
    unsigned int c[6], i;
    long val = 0L;
    for(i=0; i<6; i++)
    {
        c[5-i] = n%10;
        if(c[5-i]%2)
            c[5-i] = c[5-i]+1;
        val += c[5-i]* (long)pow(10,i);
        n = n/10;
    }
    return val;
}
```

11. (4p) Se consideră șirul definit alăturat, în care n este un număr natural nenul. ai cărui primi termeni sunt:
0, 2, 8, 22, 52 ...

$$a_n = \begin{cases} 0 & \text{dacă } n=1 \\ 1 & \text{dacă } n=2 \\ 3a_{n-1} - 2a_{n-2} + 2 & \text{dacă } n>2 \end{cases}$$

Se citesc de la tastatură două numere naturale din intervalul $[0, 10^6]$, $x < y$, reprezentând doi termeni consecutivi ai șirului. Scieți un program C/C++ care să determine și să scrie în fișierul text output.dat toți termenii șirului mai mici sau egali cu y , în ordine descrescătoare, câte 10 pe o linie, separați de un spațiu.

Exemplu: dacă se citesc numerele 494 240

fișierul va conține numerele

494 240 114 52 22 8 2 0

12. (5p) În algoritmul alăturat, prezentat în pseudocod, $[a]$ este partea întreagă a numărului real a , iar $x\%y$ este restul împărțirii numărului natural x la numărul natural nenul y .

i) Indicați numărul afișat dacă se citește valoarea 10523.

ii) Indicați patru numere întregi care pot fi citite astfel încât după executarea algoritmului pentru fiecare dintre ele să fie afișat numărul 722.

iii). Scrieți programul C/C++ corespunzător algoritmului indicat.

```

citește n (număr întreg)
m ← 0
p ← 1
x ← 0
└dacă n < 0 atunci
| n ← -n
└─
└repetă
|   c ← n%10
|   n ← [n/10]
|   └dacă c > m atunci
|       | m ← c
|       └─
|   x ← m*p+x
|   p ← p*10
└până când n=0
scrie x

```

University of Bucharest
Faculty of Physics – Admission 2026
C/C++ Programming Quiz

Items 1-10 have a single correct answer
 Items 11 and 12 will be solved in detail
 N_1 = total points for items 1-10 + 1p ex officio
 N_2 = total points for items 11 and 12 + 1p ex officio
 Final grade: $N = 0.6 \times N_1 + 0.4 \times N_2$

1. **(0.9p)** The variables x, y, and z store positive integer values. Indicate the $x < y$ && $3 * x == 5 * z$ ascending array formed by the values of these variables, if the adjacent C/C++ expression has the value 1.

A) x,y,z B) z,x,y C) z,y,x D) y,x,z

2. **(0.9p)** Using the backtracking method, all possibilities of forming sets of 5 distinct particles from the set {electron, positron, neutron, proton, muon} are generated, such that in each set the proton precedes the electron and positron. Two sets are distinct if the particles are placed in a different order. The first five generated solutions are, in order, (neutron, proton, electron, pozitron, muon), (neutron, proton, electron, muon, pozitron), (neutron, proton, pozitron, electron, muon), (neutron, proton, pozitron, muon, electron), (neutron, proton, muon, electron, pozitron). The sixth solution is:

A) (pozitron, neutron, proton, muon, electron) B) (neutron, electron, proton, muon, pozitron)
 C) (neutron, muon, proton, electron, pozitron) D) (neutron, proton, muon, pozitron, electron)

3. **(0.9p)** Consider the following code fragment, where A is a 5x5 matrix with integer elements, and $x \% y$ is the remainder of dividing the natural number x by the nonzero natural number y. The displayed value is:

A) 24 B) 52 C) 4 D) 15

```
int A[5][5], tr=0, k, l;
for(k=0;k<5;k++)
{
    for(l=0;l<5;l++)
        A[k][l]=(2*k+k*l+1)%3;
    tr += A[k][k];
}
printf("tr = %d\n\n",tr);
```

4. **(0.9p)** Consider the adjacent subroutine fun ($x \% y$ is the remainder of dividing the natural number x by the nonzero natural number y). The result of executing the instruction `fun(11);` is:

A) 1101 B) 00100 C) 1100 D) 110000

```
void fun(int x)
{ if(x>=4)
  { if(x%2==1) cout<<1;
    fun((x-1)/2);
  }
  else cout<<0;
  cout<<0;
}
```

5. **(0.9p)** Indicate which of the C/C++ expressions below has the value 1 if the natural number stored in the integer variable x is divisible by 6 and 9:

A) `!( (x%6==1) || (x%9!=0) )` B) `(x%6==1) || (x%9!=0)`
 C) `( (x%6==0) && (x%9!=0) )` D) `(x%6==1) && (x%9!=0)`

6. **(0.9p)** Consider the adjacent subprogram f ($x \% y$ is the remainder of dividing the natural number x by the nonzero natural number y). The result of executing the instruction: `f(7);` is:

A) 10 B) 010 C) 101 D) 1010

```
void f (int n)
{ cout << n%2 | printf("%d",n%2;
  if (n>=3) f(n-3);
  return ;
}
```

7. **(0.9p)** Consider the directed graph with 6 vertices, numbered from 1 to 6, represented by the adjacency lists indicated next to it. Two paths are distinct if they differ by at least one edge. The number of distinct paths from vertex 2 to vertex 4 is:

A) 0 B) 2 C) 1 D) 3

1: 3, 5
 2: 1, 6
 3: 4
 4: 5
 5: empty list

8. (0.9p) Consider the function `foo()` defined next to this cell. The code sequence:

```
char txt[32];
strcpy(txt, "polariton");
foo(txt);
```

results in displaying:

- A) plrtn B) notiralop
C) POLARITON D) nothing is displayed

```
void foo(char str[])
{
    int i, j, len = strlen(str);
    for (i = 0; i < len; i++)
    {
        if (
            str[i] == 'a' || str[i] == 'e' ||
            str[i] == 'i' || str[i] == 'o' ||
            str[i] == 'u' || str[i] == 'A' ||
            str[i] == 'E' || str[i] == 'I' ||
            str[i] == 'O' || str[i] == 'U'
        )
        {
            for (j = i; j < len; j++) {
                str[j] = str[j + 1];
            }
            i--;
            len--;
        }
        str[len + 1] = '\0';
    }
    printf("%s\n",str);
    return ;
}
```

9. (0.9p) The leaves of the rooted tree, having 8 nodes, numbered from 1 to 8, represented by the "fathers" vector (5,0,5,2,2,4,1,2) are:

- a. 3,6,7,8 b. 1,8,4 c. 2,7 d. 1,6,7,8

10. (0.9p) Consider the function `subst()` defined in the next cell. Code sequence:

```
long nr = 287452;
printf("subst(%ld) = %ld\n",nr,subst(nr));
```

results in displaying:

- a. `subst(nr) = -1L`
b. `subst(287452) = 28745`
c. `subst(287452) = 254782`
d. `subst(287452) = 288462`

```
long subst(long n)
{
    if(n<0 || n>1000000)
        return -1L;
    unsigned int c[6],i;
    long val = 0L;
    for(i=0;i<6;i++)
    {
        c[5-i] = n%10;
        if(c[5-i]%2)
            c[5-i] = c[5-i]+1;
        val += c[5-i]* (long)pow(10,i);
        n = n/10;
    }
    return val;
}
```

11. (4p) Consider the adjacent array, where n is a nonzero natural number, whose first terms are:
0, 2, 8, 22, 52 ...

$$a_n = \begin{cases} 0 & \text{if } n=1 \\ 1 & \text{if } n=2 \\ 3a_{n-1} - 2a_{n-2} + 2 & \text{if } n>2 \end{cases}$$

Two natural numbers from the interval $[0, 10^6]$, $x < y$, are read from the keyboard, representing two consecutive terms of the array. Write a C/C++ program that determines and writes to the text file *output.dat* all terms of the array less than or equal to y , in descending order, 10 per line, separated by a space.

Example: if the numbers 494 240 are read, the file will contain the numbers
494 240 114 52 22 8 2 0

12. (5p) In the algorithm shown in pseudocode, $[a]$ is the integer part of the real number a , and $x\%y$ is the remainder of dividing the natural number x by the nonzero natural number y .

i) Indicate the number displayed if the value 10523 is read.

ii) Indicate four integers that can be read so that after executing the algorithm, the number 722 is displayed for each of them.

iii). Write the C/C++ program corresponding to the indicated algorithm.

```
read n (integer number)
m ← 0
p ← 1
x ← 0
┌if n < 0 then
| n ← -n
└─
┌repeat
|   c ← n%10
|   n ← [n/10]
|   ┌if c > m then
|   | m ← c
|   └─
|   x ← m*p+x
|   p ← p*10
└until n=0
write x
```