

RECURSIVITATE

1. Pentru definiția de mai jos a subprogramului **f**, ce se afișează ca urmare a apelului **f(121,1);?** (6p.)
- ```
//C
void f(long n, int i)
{ if(n!=0) printf("%d",i);
 if(n%3>0) f(n/3,i+1);
}
```
2. Pentru definiția alăturată a subprogramului **f**, ce se afișează ca urmare a apelului **f(125)?** (6p.)
- ```
//C++
void f(long n, int i)
{ if(n==0) cout<<i;
  if(n%3>0) f(n/3,i+1);
}
```
3. Pentru definiția alăturată a subprogramului **f**, ce se afișează ca urmare a apelului **f(26);?** (6p.)
- ```
//C++
void f(int n)
{ cout<<n%10; | printf("%d",n%10);
 if(n!=0)
 { cout<<n%10; | printf("%d",n%10);
 f(n/100);
 }
}
```
4. Pentru definiția de mai jos a subprogramului **f**, ce se afișează ca urmare a apelului **f(10);?** (6p.)
- ```
//C
void f (int b)
{ if(3<=b)
  {f(b-2); printf("%c",'*');}
  else printf("%d",b);
}
```
5. Se consideră subprogramul cu definiția alăturată. Ce valoare se va afișa în urma executării instrucțiuni de mai jos? (4p.)
- ```
cout<<f(12); |
printf("%d",f(12));
```

a. 11002

b. 20011

```
//C++
void f (int b)
{ if(3<=b)
 { f(b-2); cout<<'*' ;}
 else cout<<b;
}

int f (int n){
 int c;
 if (n!=0)
 {if (n%2==1)
 c=1+f(n/2);
 else c=f(n/2);
 cout<<n%2; | printf("%d",n%2);
 return c;
 }
 else return 0;
}
```

c. 10102

d. 00112

6.

Se consideră subprogramul cu definiția alăturată. Ce se va afișa în urma apelului **f(12345);?** (4p.)

a. 315  
b. 24  
c. 24

```
void f(long int n){
 if (n!=0){
 if (n%2==0)
 cout<<n%10; | printf("%d",n%10);
 f(n/10);
 if (n%2!=0)
 cout<<n%10; | printf("%d",n%10);
 }
 else cout<<endl; | printf("\n");
}
```

c. 513  
d. 42  
e. 42  
f. 135

7.

Se consideră subprogramul **f** cu definiția alăturată. Ce se va afișa în urma apelului **f(12345);?** (4p.)

```
void f(long int n){
 if (n!=0){
 if (n%2!=0)
 cout<<n%10; | printf("%d",n%10);
 f(n/10);
 if (n%2==0)
 cout<<n%10; | printf("%d",n%10);
 }
 else cout<<endl; | printf("\n");
}
```

a. 531  
b. 135  
c. 24  
d. 42

e. 135  
f. 531  
g. 24  
h. 42

8.

Se consideră subprogramul cu definiția alăturată. Ce valoare se va afișa în urma executării instrucțiuni de mai jos?

**cout<<f(8);  
printf("%d",f(8));** (4p.)

a. 10003  
b. 30001

```
int f (int n){
 int c;
 if (n!=0)
 {if (n%2==0)
 c=1+f(n/2);
 else c=f(n/2);
 cout<<n%2; | printf("%d",n%2);
 return c;
 }
 else return 0;
}
```

c. 10013  
d. 00112

9.

Se consideră subprogramul cu definiția alăturată. Ce valoare are **f(3,1)?** (4p.)

a. 8  
b. 9

```
int f(int n,int y)
{ if(n!=0)
 { y=y+1;
 return y+f(n-1,y);
 }
 else return 0;
}
```

c. 7  
d. 6

10.

Ce valoare are **f(23169)**, pentru funcția **f**, definită alăturat? (6p.)

```
int f(int n){
 if (n==0) return 0;
 else
 if (n%2==0)
 return n%10+f(n/10);
 else return f(n/10);
}
```

11.

Ce valoare are **f(4063)** pentru funcția **f**, definită alăturat? (6p.)

```
long f(long n){
 if (n==0) return 1;
 else
 if (n%10==0)
 return f(n/10);
 else
 return (n%10) * f(n/10);
}
```

12.

Care este valoarea lui **f(34)** pentru funcția **f**, definită alăturat? (6p.)

```
long f(long x) {
 if (x==4) return x;
 else
 { if (x%10==4 || x%10==0) return x+f(x/10);
 else return x+f(x*2);
 }
}
```

13.

Funcția **F** are definiția alăturată. Ce valoare are **F(3)**? (4p.)

```
int F(int n){
 if(n==0 || n==1) return 1;
 else
 return 2*F(n-1)+2*F(n-2);
}
```

a. 1

b. 12

c. 6

d. 10

14.

Subprogramul **f** are definiția alăturată. Ce se va afișa în urma apelului **f(12345)**? (4p.)

```
void f(long n){
 if (n>9)
 {cout<<n/100; | printf("%d",n/100);
 f(n/10);
 }
}
```

a. 1231210

b. 123121

c. 1234123121

d. 123

15.

Funcția **f** are definiția alăturată. Ce se va afișa în urma apelului **f(12345,0)**? (4p.)

```
void f(long n, int i){
 if (i<n%10)
 {cout<<n%10; | printf("%d",n%10);
 f(n/10,i+1);
 }
}
```

a. 54321

b. 543

c. 54

d. 5432

16.

Funcția **F** are definiția alăturată. Ce valoare are **F(18)**? (4p.)

```
int F(int x){
 if (x<=1) return x;
 else
 return x+F(x-2);
}
```

a. 90

b. 171

c. 1

d. 18

17.

Funcția **F** are definiția alăturată. Ce valoare are **F(5)**?

(4p.)

```
int F(int x)
 {if(x!=0) return x+F(x-1);
 else
 return x;}
```

5                    b. 10

c. 15              d. 6

18.

Se consideră subprogramul, **f**, definit alăturat. Ce valoare are **f(20)**?

(6p.)

```
int f(int n);
{ if (n==0) return 0;
 return n%2+f(n/2);
}
```

19.

20.

Subprogramul **f** este definit alăturat.

Ce se afișează ca urmare a apelului **f(1,4)**?

(6p.)

```
void f (int x,int y)
{ if(x<=y){ f(x+1,y); cout<<i; }
```

21.

Funcția **f** are definiția alăturată.

a) Ce valoare are **f(17)**?

(3p.)

```
int f(int n);
{ if (n<=9) return 0;
 if (n%4==0) return 0;
 return 1+f(n-3);
}
```

b) Ce valoare are **f(22)**?

(3p.)

```
int f(int n);
{ if (n<=0) return -1;
 if (n % 2==0) return 0;
 if (n % 3==0) return 0;
 return 1+f(n-10);
}
```

22.

Funcția **f** are definiția alăturată:

a) Ce valoare are **f(16)**?

(3p.)

b) Scrieți o valoare de două cifre pe care o poate avea **n** astfel încât **f(n)** să fie egal cu 2.

23.

Subprogramul **afis** este definit alăturat.

Ce se afișează ca urmare a apelului **afis(4)**?

(4p.)

```
void afis (int n)
{ cout<<n; | printf("%d",n);
 if(n>0){afis(n-1);
 cout<<n; | printf("%d",n);}
}
```

24.

Subprogramul **scrive** este definit alăturat.

Ce se afișează ca urmare a apelului **scrive(1,7)**?

(6p.)

```
void scrive (int x,int y)
{ if(x<y)
 { scrive(x+1,y-1);
 cout<<(x+y)/2; | printf("%d", (x+y)/2);
 }
}
```

25.

Subprogramul **f** este definit alăturat.

Ce valoarea are **f(8,4)**?

(4p.)

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

26.

Ce se afișează ca urmare a apelului **p(123)**; dacă subprogramul **p** are definiția alăturată.

(6p.)

```
void p (int x)
{ if(x!=0){p(x/10);
 cout<<x%10; | printf("%d",x%10);
}
}
```

27.

Subprogramul **f** este definit alăturat.

Ce se afișează ca urmare a apelului **f(1, 4)**? (6p.)

```
void f (int x,int y)
```

```
{ if(x<=y) { f(x+1,y); cout<<i; }
```

28.

Se consideră subprogramul recursiv definit alăturat. Ce se va afișa în urma apelului **f1(4)**? (6p.)

```
void f1(int x)
```

```
{if (x<=9)
```

```
{ cout<<x+1; | printf("%d",x+1)
```

```
f1(x+2);
```

```
}
```

29.

Subprogramul **afis** este definit alăturat. Ce se va afișa în urma apelului **afis(17)**? (6p.)

```
void afis(int x)
```

```
{ if (x>3)
```

```
{
```

```
cout<<x-1<<" "; | printf("%d ",x-1);
```

```
afis(x/3);
```

```
}
```

```
}
```

30.

Subprogramul **f** este definit alăturat.

a) Ce valoare va avea **f(7)**?

b) Determinați două valori **x1** și **x2** ( $x_1, x_2 < 12$ )

pentru care **f(x1)=f(x2)**. (6p.)

```
int f(int i)
```

```
{
```

```
if (i>12) return 1;
```

```
else return 1+f(i+2);
```

```
}
```

32.

a) Scrieți definiția completă a unui subprogram recursiv **sum** care primește prin parametrul **x** un număr natural de cel mult 4 cifre și returnează suma divizorilor numărului **x**, diferență de 1 și de el însuși.

**Exemplu:** dacă **x=10** se va returna valoarea 7 ( $7=2+5$ ). (4p.)

b) Scrieți programul c/c++ care citește de la tastatură un număr natural **n** ( $0 < n < 100$ ), apoi **n** numere naturale (cu cel mult 4 cifre fiecare). Programul determină, folosind apeluri utile ale subprogramului **sum**, pentru fiecare număr natural, suma divizorilor săi proprii și afișează pe ecran sumele determinate, în ordinea crescătoare a valorilor lor, separate prin câte un spatiu. (6p.)

**Exemplu:** dacă **n=5** și numerele citite sunt **10 2 33 6 11**

valorile afișate pe ecran vor fi: **0 0 5 7 14**

deoarece suma divizorilor lui **10** este **7**, suma divizorilor lui **2** este **0**, suma divizorilor lui **33** este **14**, suma divizorilor lui **6** este **5**, suma divizorilor lui **11** este **0**.

33.

Subprogramul **scif** returnează suma cifrelor unui număr natural transmis ca parametru.

Care este valoarea expresiei **scif(scif(518)+scif(518))**? (4p.)

- a. 10                    b. 14                    c. 28                    d. 1

34.

Considerăm subprogramul **f** definit alăturat.

Ce valoare are **f(11, 7)**? (6p.)

```
int f(int x,int y)
```

```
{if(x<=y) return x-y;
```

```
return f(y-x,x-1)+3;}
```

35.

Pentru definiția alăturată a subprogramului **sc**, stabiliți ce valoare are **sc(901324)**? (6p.)

```
int sc(long x)
```

```
{if(x<10) return x;
```

```
return sc(x/10)+x%10;}
```

36.

Pentru definiția alăturată a subprogramului **f**, ce valoare are **f(8)**? (4p.)

```
int f(int x)
```

```
{if(x<=4) return x*x-3;
```

```
return f(x-3)+4;}
```

37.

Pentru definiția alăturată a subprogramului **f**, stabiliți ce valoare are **f(23461)**? (4p.)

```
int f(long x)
{ if(x<10) return 1;
 return f(x/10)+1; }
```

38.

Subprogramul **f** are definiția alăturată. Ce valoare are **f(7,2)**? Dar **f(35,2)**? (6p.)

```
int f(int x, int y)
{ if(x%y==0) return y;
 else return f(x,y+1);
}
```

39.

Subprogramul **f** are definiția alăturată. Ce valoare are **f(7)**? Dar **f(100)**? (6p.)

```
int f(int x)
{ if(x%6==0) return x;
 else return f(x-1);
}
```

40.

Subprogramul **f** are definiția alăturată. Ce valoare are **f(3)**? Dar **f(10)**? (6p.)

```
int f(int x)
{ if(x==0) return 0;
 else return f(x-1)+2;
}
```

41.

Subprogramul **f** are definiția alăturată. Ce valoare are **f(5,10)**? (6p.)

```
int f(int x,int y)
{ if(x==y) return x;
 else if(x<y) return f(x+1,y-1);
 else return f(x-1,y);
}
```

42.

Subprogramul **f** are definiția alăturată. Ce valoare are **f(4)**? Dar **f(11)**? (6p.)

```
int f(int x)
{ if(x<1) return 1;
 else return f(x-3)+1;
}
```

43.

Se consideră subprogramul **f** definit alăturat: Ce valori are **f(7)**? Dar **f(100)**? (6p.)

```
long f(int nr)
{
 if(!nr) return 0;
 else return f(nr-1)+2*nr;
}
```

44.

45.

46.

47.

48.

49.

Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(4)**? Dar **f(100)**? (6p.)

```
long f(unsigned int n)
{ if (n==0) return 0;
 else return n+f(n-1);
}
```

50.

Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(2138)**? Dar **f(513)**? (6p.)

```
int f(unsigned int n)
{ if (n==0) return 0;
 else if (n%2==0)
 return n%10+f(n/10);
 else
 return f(n/10);
}
```

51. Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(97,2)**? Dar **f(175,2)**? (6p.)

```
int f(int n, int x)
{
 if (n<=1) return 0;
 else if (x<=n/2)
 if n%x==0 return 0;
 else return f(n,x+1);
 else return 1;
}
```

52. Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(5)**? Dar **f(40)**? (6p.)

```
int f(unsigned int n)
{
 if (n>20) return 0;
 else return 5+f(n+5);
}
```

53. Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(20)**? (6p.)

```
int f(int n)
{
 if (n==0) return 0;
 return 1+f(n/2);
}
```

54. Se consideră subprogramul **f**, definit alăturat. Ce se va afișa la apelul **f(38)**? (6p.)

```
void f(int x)
{
 if(x)
 {
 f(x/3);
 printf("%d",x%3+1); | cout<<x%3+1;
 }
}
```

55. Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(3713)**? (6p.)

```
int f(int n){
 if(n==0)
 return 0;
 return f(n/10)*10+1;
}
```

56. Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(8261)**? (6p.)

```
int f(int a)
{
 if(a<10)
 return 7;
 return f(a/100)*10+8;
}
```

57. Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(23)**? (6p.)

```
int f(int x)
{
 if (x%2==0)
 return 0;
 return 1+f(x/2);
}
```

58. Se consideră subprogramul **f** definit alăturat. Ce valoare are **f(456)**? (6p.)

```
int f(int x)
{
 if(x>=1)
 return f(x-1)+1
 else
 return 0;
}
```

59.

Considerăm subprogramul **f** definit alăturat. Ce se afișează pe ecran la apelul **f(4962);?** (6p.)

```
void f1(int n)
{
 int c;
 if(n!=0)
 {c=n%10;
 printf("%d",c); | cout<<c;
 f1(n/10);
 printf("%d",c); | cout<<c;
 }
}
```

60.

Se consideră subprogramul **f** definit alăturat. Ce valoare are **f(3)?**

```
int f(int i)
{
 if(i>=1)
 return f(i-1)+i;
 else return 0;
}
```

61.

Se consideră subprogramul **f** definit alăturat. Ce se afișează pe ecran la apelul **f(9,9)?**

(6p.)

```
void f(int i,int j)
{
 if(j>=0) f(i,j-1);
 printf("%d*%d=%d\n",i,j,i*j);
 | cout<<i<<'*'<<j<<'='<<i*j<<endl;
}
```

62.

Subprogramul recursiv alăturat este definit incomplet. Care este expresia cu care se pot înlocui punctele de suspensie astfel încât **f(40,16)** să fie egal cu **8**.

(6p.)

```
int f(unsigned int a, unsigned int b)
{
 if (...)
 return a;
 else
 if (a>b) return f(a-b,b);
 else return f(a,b-a);
}
```

63.

Se consideră subprogramul recursiv definit alăturat. Ce se va afișa în urma apelului **bac(5)?** (4p.)

```
void bac(int x)
{
 if (x)
 { bac(x-1);
 cout<<x; | printf("%d",x);
 }
}
```

55555

b. 54321

c. 12345

d. 11111

64.

Se consideră subprogramul recursiv definit alăturat. Ce se va afișa în urma apelului **bac(5);?** (4p.)

```
void bac(int x)
{
 if (x)
 { cout<<x;
 bac(x-1); | printf("%d",x);
 }
}
```

a. 54321

b. 12345

c. 11111

d. 55555

65.

Se consideră subprogramul recursiv definit alăturat. Câte apeluri ale funcției **bac** au loc pentru **n=5**? (Se va număra inclusiv apelul din funcția principală.)

(4p.)

```
void bac(int x)
{ if (x)
 { bac(x-1);
 cout<<x; | printf("%d",x);
 }
}
```

5

b. 6

c. 4

d. 3

66.

Se consideră subprogramul recursiv definit alăturat. Câte apeluri ale funcției **bac** au loc pentru **n=5**? (Se va număra inclusiv apelul din funcția principală.)

(4p.)

```
void bac(int x)
{ if (x>0)
 { bac(x-2);
 cout<<x; | printf("%d",x);
 }
}
```

4

b. 3

c. 6

d. 5

67.

Se consideră subprogramul recursiv definit alăturat. Câte apeluri ale funcției **bac** au loc pentru **n=4**? (Se va număra inclusiv apelul din funcția principală.)

(4p.)

```
void bac(int x)
{ if (x>0)
 { cout<<x;
 bac(x-2);
 }
}
```

6

b. 4

c. 5

d. 3

68.

Ce valoare va avea variabila întreagă **x**, în urma apelului **F(1, x)**, știind că, înainte de apel, variabila **x** are valoarea 0, iar subprogramul **F** este definit alăturat?

(6p.)

```
void F(int i, int &x)
{ if (i <= 10)
 { if(i % 2) x = x + 2;
 else x = x - 1;
 F(i + 1, x);
 }
}
```

69.

Ce se va afișa în urma executării subprogramului alăturat, la apelul **F(57)**? (6p.)

```
void F(int x)
{ if(x)
 { F(x/2);
 cout << x%2; | printf("%d",x%2);
 }
}
```

70.

Ce se va afișa în urma executării subprogramului alăturat, la apelul **F(56)**? (6p.)

```
void F(int x)
{ if(x)
 { F(x/2);
 cout << x%10; | printf("%d",x%10);
 }
}
```

71.

Se consideră funcția **Suma**, definită alăturat. Ce valoare are **Suma(8)**? Dar **Suma(11)**? (6p.)

(6p.)

```
int Suma(int x)
{
 if(x == 1) return 0;
 if(x%2==0) return Suma(x-1)+(x-1)*x;
 return Suma(x-1)-(x-1)*x;
}
```

72.

Ce valoare are  $F(2758)$ , pentru funcția  $F$ , definită alăturat?

(4p.)

- a. 0      b. 20

```
int F(int x)
```

{

```
 if(x == 0) return 0;
```

```
 if(x%10 == 0) return 2 + F(x/10);
```

```
 return 10 - F(x/10);
```

c. 12

d. 4

73.

Pentru definiția alăturată a subprogramului  $sub$ , stabiliți ce valoare are  $sub(132764)$ ?

(6p.)

```
long sub(long n)
```

{if (n!=0)

```
 if(n%2!=0) return n%10*sub(n/10);
```

```
 else return sub(n/10);
```

```
 else return 1;
```

}

74.

Pentru definiția alăturată a subprogramului  $sub$ , stabiliți ce valoare are  $sub(132764)$ .

(6p.)

```
int sub(long n)
```

{if (n!=0)

```
 if(n%2!=0) return n%10+sub(n/10);
```

```
 else return sub(n/10);
```

```
 else return 0;
```

}

75.

Pentru definiția alăturată a subprogramului  $sub$ , stabiliți care este valoarea expresiei  $sub(123986)$ ?

(6p.)

```
int sub(long n)
```

{if (n!=0)

```
 if(n%2!=0) return 1+sub(n/10);
```

```
 else return sub(n/10);
```

}

76.

Pentru definiția alăturată a subprogramului  $f$ , stabiliți ce valoare are expresia  $f(1209986)$ .

(6p.)

```
int f(long x)
```

{ int y,z;

```
 if (x==0) return x;
```

```
 else {y=x%10;
```

```
 z=f(x/10);
```

```
 if(y>z) return y ;
```

```
 else return z;}
```

}

77.

Pentru definiția alăturată a subprogramului  $sub$ , stabiliți ce valoare are  $sub(132764)$ .

(6p.)

```
int sub(long n)
```

{if (n!=0)

```
 if(n%2!=0) return n%10+sub(n/10);
```

```
 else return sub(n/10);
```

```
 else return 0;}
```

78.

Pentru definiția alăturată a subprogramului  $f$ , scrieți ce valoare are  $f(5552)$ .

(6p.)

```
int f(int x)
```

{ if(x==0)

```
 return 0;
```

```
 else
```

```
 return f(x/10)+1;
```

}

79.

Pentru definiția alăturată a subprogramului  $f$ , scrieți ce valoare are  $f(123)$ .

(6p.)

```
int f(int x)
```

{ if(x==0) return 0;

```
 else if(x%2==0) return 1+f(x/10);
```

```
 else return 2+f(x/10);
```

}

80.

Pentru definiția alăturată a subprogramului **f**, scrieți ce valoare are **f(123)**. (6p.)

```
int f(int x)
{
 if(x==0) return 0;
 else
 if(x%2==0) return 3+f(x/10);
 else return 4+f(x/10);
}
```

81.

Pentru definiția alăturată a subprogramului **f**, scrieți ce valoare are **f(0)**. (6p.)

```
int f(int x)
{
 if(x==100) return 1;
 else return 1+f(x+1);
}
```

82.

Pentru definiția alăturată a subprogramului **f**, scrieți ce valoare are **f(30)**. (6p.)

```
int f(int x)
{
 if(x==20) return 20;
 else if(x%2==1)
 return 1+f(x-1);
 else
 return 2+f(x-1);
}
```

83.

Se consideră subprogramul **f**, definit alăturat. Ce se afișează la apelul **f(4)**? (6p.)

```
void f(int n)
{
 cout<<"*"; | printf("*");
 if(n>2)
 {
 f(n-1);
 cout<<"#"; | printf("#");
 }
}
```

84.

Se consideră subprogramul **f**, definit alăturat. Ce se afișează la apelul **f(4)**? (6p.)

```
void f(int n)
{
 if(n>0)
 {
 cout<<n; | printf("%d",n);
 f(n-1);
 cout<<n; | printf("%d",n);
 }
}
```

85.

Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(4)**? (6p.)

```
int f(int n)
{
 if (n==0) return 1;
 else if (n==1) return 2;
 else return f(n-1)-f(n-2);
}
```

86.

Se consideră subprogramul **f**, definit alăturat. Ce valoare are **f(4)**? (6p.)

```
long f(int n)
{
 if (n==0) return 0;
 else return n*n+f(n-1);
}
```

87.

Se consideră subprogramul **f**, definit alăturat. Ce se afișează la apelul **f('a')**?

(6p.)

```
void f(char c)
{
 if (c != 'e')
 {
 f(c+1);
 cout<<c; | printf("%c", c);
 }
}
```

88.

Se consideră subprogramul **f**, definit alăturat. Ce se afișează la apelul **f(20)**?

(6p.)

```
void f (int i)
{if(i!=0)
{ printf("%d",i); | cout<<i;
f(i/2);
printf("%d",i); | cout<<i;
}
}
```

89.

Se consideră definit subprogramul **f**.

a) Ce se va afișa în urma apelului **f(14)**?

b) Scrieți valorile pe care le poate avea **x**, astfel încât în urma apelului **f(x)**; să se afișeze pe ecran exact 10 numere.

(6p.)

```
void f(int x)
{
 if (x<=10)
 cout<<0<<" "; | printf("%d ",0);
 else
 { f(x-2);
 cout<<x<<" "; | printf("%d ",x);
 }
}
```

90.

Subprogramul **f** este definit alăturat.

Ce se afișează ca urmare a apelului **f(1,4)**?

(4p.)

```
void f (int x,int y)
{ if (x<y)
 {y=y-1; f(x,y);}
 else
 cout<<x<<y; | printf("%d%d",x,y);
}
```

91.

Funcția **f** este astfel definită încât **f(1)=8**, iar **f(n+1)=2\*f(n)-4** (**n** natural, **n>1**).

a) Ce valoare are **f(5)**?

(3p.)

b) Care este cea mai mare valoare pe care o poate lua **x** astfel încât **f(x) < 1000**?

(3p.)

92.

Funcția **f** are definiția alăturată.

a) Ce valoare are **f(10)**?

(3p.)

b) Ce valoare are **f(20)**?

(3p.)

93.

Funcția **f** are definiția alăturată. Scrieți 4 valori de apel pe care le poate avea **n** astfel încât, pentru cele 4 apeluri, corespunzătoare acestor valori, să se obțină 4 valori, disticte două căte două. (6p.)

```
int f(int n)
{
 if (n<=9) return 0;
 if (n%5==0) return 0;
 return 1+f(n-3);
}

int r(int n)
{if (n<=9) return 0;
if (n%4==0) return 0;
return 1+f(n-3);
}
```

94.

Subprogramul recursiv alăturat este definit incomplet. Care dintre următoarele expresii poate înlocui punctele de suspensie astfel încât, în urma apelului, subprogramul **f** să returneze suma primelor două cifre ale numărului primit prin intermediul parametrului **x**.

**Exemplu:** în urma apelului **f(2318)** valoarea returnată este 5.

(4p.)

a. **x<=100**b. **x<=99**c. **x==99**d. **x!=0**

```
int f(int x){
 if (...)
 return x%10 + x/10;
 else
 return f(x/10);
}
```

95.

Se consideră subprogramul recursiv alăturat, definit incomplet.

Cu ce valoare trebuie înlocuite punctele de suspensie, pentru ca funcția să returneze cifra minimă a numărului natural nenul transmis prin intermediul parametrului **x**?

(4p.)

a. -1

b. 1

c. 9

d. 0

```
int Min(int x){
 int c;
 if (x==0) return ... ;
 else {
 c=Min(x/10);
 if (c < x%10) return c;
 else return x%10;
 }
}
```

96.

Se consideră subprogramul recursiv alăturat, **s**, definit incomplet.

Cu ce expresie pot fi înlocuite punctele de suspensie astfel încât, în urma apelului **s(2)**, să se afișeze 3 caractere \* ?

(4p.)

a. **x>1**b. **x>2**c. **x>=3**d. **x>0**

```
void S(int x)
{ cout<<'*';
 if (...) {
 cout<<'*';
 S(x-1);
 }
}
```

97.

Ce afișează subprogramul **F**, descris alăturat, la apelul **F(5)**?

(6p.)

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