SUBPROGRAME (FUNCTII)

Un subprogram reprezinta un ansamblu de instructiuni (de declarare, executabile) scrise în vederea executarii unei anumite prelucrari. Este identificat printr-un nume si implementat separat.

I.CLASIFICARE

- functii procedurale
- functii operand

Functii procedurale (void) - nu returneaza niciun rezultat sau returneaza prin parametri

Exemplu:

```
void afiseaza(int n)
{
   cout<<n;
  }
```

Functii operand - returneaza o valoare de tipul specificat

Exemplu:

```
int dubleaza(int n)
 {
    return 2*n;
  }
 int main()
 {
    cout << dubleaza(2); // va afisa 4
  }
```

II.TRANSMITEREA PARAMETRILOR

- prin valoare - pentru date de intrare

-prin referinta - pentru date de iesire sau intrare – iesire

Exemplu:

```
#include <iostream>
using namespace std;
void modifica(int a)
{
a = a + 5;
cout << "a are valoarea: " << a << '\n';
}
int main()
```

```
#include <iostream>
using namespace std;
void modifica(int &a)
{
a = a + 5;
cout << "a are valoarea: " << a << '\n';
}
int main()
```

{
int x = 1;
cout << "x inainte de apel: " << x << '\n';
modifica(x);
cout << "x dupa apel: " << x;
return 0;
}
	{
	int x = 1;
	cout << "x inainte de apel: " << x << '\n';
	modifica(x);
	cout << "x dupa apel: " << x;
	return 0;
	}

Output: x inainte de apel: 1 a are valoarea: 6 x dupa apel: 1	Output: x inainte de apel: 1 a are valoarea: 6 x dupa apel: 6

III.SUBPROGRAME SI TABLOURI

Tablourile nu se transmit utilizand operatorul &, dar orice modificare a valorilor elementelor tabloului dat ca parametru formal va afecta elementul corespunzător al tabloului dat ca parametru actual.

Exemple:

1. Subprogram citire vector

```
void citeste(int a[], int n)
{int i;
  for(i=1;i<=n;i++)
     cin>>a[i];
}
```

2. Subprogram stergere element pozitia k in vector

```
void stergere(int &n, int a[], int k)
{int i;
for(i=k;i<=n-1;i++)
   a[i]=a[i+1];
n--;
}
```

3. Subprogram inserare element pozitia k in vector

```
void inserare(int &n, int a[], int k)
{int i;
for(i=n;i>=k;i--)
   a[i+1]=a[i];
a[k]=0;
n++;
}
```

4. Subprogram ordonare:

```
void ordonare(int n, int a[])
{int i,j;
for(i=1;i<=n-1;i++)
   for(j=i+1;j<=n;j++)
     if(a[i]>a[j])
       swap(a[i],a[j]);
}
```

5.Subprogram afisare matrice

```
void afisare(int A[][101], int n)
{
   for(int i=1;i<=n;i++)
   {
      for(int j=1;j<=n;j++)
         cout<<A[i][j]<<" ";
      cout<<endl;
   }
}
```

6. Subprogram stergere linia k din matrice

```
void sterglinie(int A[][101], int &n, int m, int k)
{
   for(int i=k;i<n;i++)
      for(int j=1;j<=m;j++)
         A[i][j]=A[i+1][j];
   n--;
}
```

IV.ALTE EXEMPLE DE FUNCTII

1.Verificare numar prim (prin referinta)

```
void prim(int n, int &p)
{
   p=1;
   if(n==0 || n==1) p=0;
   else if(n%2==0 && n!=2) p=0;
     else for(int d=3;d*d<=n;d=d+2)
            if(n%d==0) p=0;
}
```

2.Cel mai mare divizor comun

```
void cmmdc(int a, int b, int &c)
{
   while(b!=0)
   {
      int r=a%b;
      a=b;
      b=r;
   }
   c=a;
}
```

3. Suma divizorilor primi ai unui numar natural

```
int sum_div_prim(int n)
{
   int s=0;
   int d=2;
   while(n>=d*d)
   {
      if(n%d==0)
      {
         s=s+d;
         while(n%d==0) n=n/d;
      }
      else if(d==2) d++;
          else d=d+2;
   }
   if(n!=1)  s=s+n;
   return s;
}
```

V. PROBLEME PROPUSE.

1.Scrieti cate un program in care sa utilizati subprogramele de mai sus.

2. Subprogramul **putere** are trei parametri:

• **n**, prin care primește un număr natural din intervalul **[2,10^9]**;

• **d** și **p**, prin care furnizează divizorul prim, **d**, care apare la cea mai mare putere, **p**, în descompunerea în factori primi a lui **n**; dacă există mai mulți astfel de divizori se afișează cel mai mare dintre ei.

Scrieți definiția completă a subprogramului.

Exemplu: dacă **n=10780**, atunci, în urma apelului, **d=7** și **p=2** (**10780=2^2** x **5** x **7^2** x **11**)

3. Un număr natural nenul se numește **sPar** dacă atât el, cât și suma divizorilor săi proprii (divizori diferiți de **1** și de el însuși), sunt numere pare.

Subprogramul **sPar** are un singur parametru, **n**, prin care primeşte un număr natural (**n** din intervalul **[1,10^9]**). Subprogramul returnează cel mai mic număr sPar, strict mai mare decât **n**. Scrieţi definiţia completă a subprogramului.

Exemplu: dacă **n=95**, atunci subprogramul returnează **98** (atât **98**, cât și **72=2+7+14+49**, sunt numere pare).

4. Subprogramul **Egal** are un parametru, **n**, prin care primeşte un număr natural cu cel puțin o cifră impară (**n** apartine intervalului **[10**,**10^9]**). Subprogramul returnează valoarea **1** dacă toate cifrele impare ale lui **n** sunt egale între ele sau valoarea **0** în caz contrar. Scrieți definiția completă a subprogramului.

Exemplu: dacă **n=7727470** sau **n=7240** atunci subprogramul returnează **1**, iar dacă **n=7921470** atunci subprogramul returnează **0**.

5. Subprogramul **inserare** are doi parametri:

- **n**, prin care primeste un număr natural (**2≤n≤20**);
- **a**, prin care primeste un tablou unidimensional care memorează un sir de **n** numere naturale, fiecare cu cel mult **4** cifre. Cel putin un element al tabloului este număr par.

Subprogramul modifică tabloul, inserând înainte de fiecare termen par al sirului numărul obtinut prin împărtirea la **2** a valorii acestuia si furnizează, tot prin parametrii **n** si **a**, valorile actualizate ale datelor primite. Scrieti în limbajul **C/C++** definitia completă a subprogramului.

Exemplu: dacă **n=7** si **a=(1,4,5,3,82,6,2)** atunci după apel **n=11** si **a=(1,2,4,5,3,41,82,3,6,1,2)**.