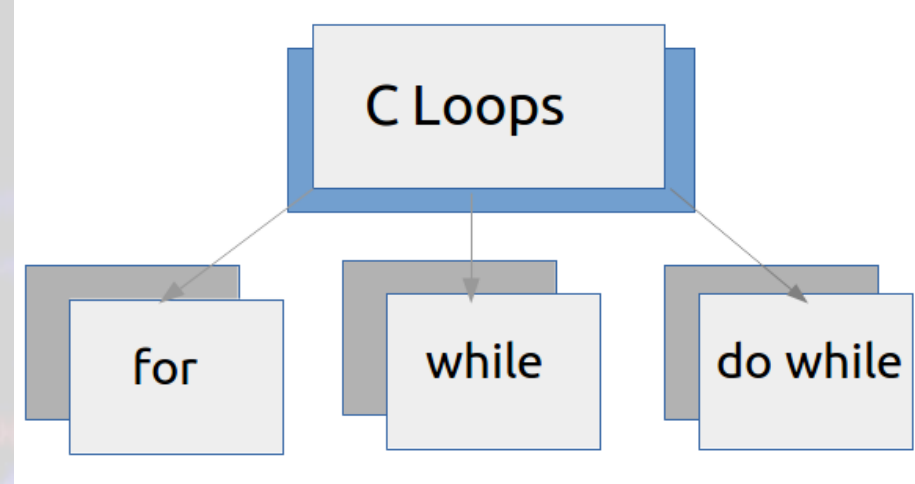


# Structura repetitivă

Structura repetitivă condiționată anterior



# Structura repetitivă condiționată anterior

- Atunci când se descrie o secvență repetitivă, este posibil să nu se știe câți pași trebuie făcuți.
- Se folosește o structură repetitivă care se oprește când o condiție nu mai este îndeplinită

*Exemplu* – cât timp mai este prăjitură în farfurie, mănânc din ea.

**Cât timp condiție execută**

*instrucțiuni*

**Funcționare:** *cât timp condiția este adevărată, se execută corpul instrucțiunii repetitive*

# 1. Scrieți un program care afișează primele n numere naturale.

```
#include <iostream>
using namespace std;
int main()
{
    int n,i=0;
    cin>>n;
    while(i<=n)
    {
        cout<<i<<" ";
        i++;
    }
    return 0;
}
```

C

++

## 2. Scrieți în ordine descrescătoare numerele mai mici ca **n**.

```
#include <iostream>  
using namespace std;  
int main()  
{  
    int n;  
    cin>>n;  
    while(n>=0)  
    {  
        cout<<n<<" ";  
        n--;  
    }  
    return 0;  
}
```



documentation

update

algorithm

deployment  
maintenance

mysql  
system  
interpreter  
programmer  
computing

programming  
Software  
computer  
application  
source  
optimization  
program  
client  
bug  
compiler  
approach  
coding  
requirements  
step  
Perl  
classical  
structure  
engineering  
bug  
upgrade  
feature  
embedded  
software  
procedure  
platform  
update

3. Scrieți un program care afișează numerele naturale impare până la  $n$ .

```
#include <iostream>
using namespace std;
int main()
{
    int n,i=0;
    cin>>n;
    while(i<=n)
    {
        if(i%2==1)
            cout<<i<<" ";
        i++;
    }
    return 0;
}
```



4. Scrieți un program care testează dacă un număr  $n$  este prim sau nu, folosind structura **while**.

```
database
#include <iostream> data
using namespace std;
int main()
{
long int n,prim=1,d=2;
cin>>n;
while (d<=n/2)
{
if (n%d==0)
prim=0;
d++;
}
if (prim==1)
cout<<"n este numar prim";
else
cout<<"n nu este numar prim";
return 0;
}
```

5. Scrieți un program care afișează suma primelor  $n$  numere naturale.

```
#include <iostream>
using namespace std;
int main()
{
    int n,i=0,s=0;
    cin>>n;
    while(i<=n)
    {
        s=s+i;
        i++;
    }
    cout<<s;
    return 0;
}
```

C

++

## 6. Scrieți un program care afișează suma cifrelor unui număr

```
database
#include <iostream>
using namespace std;
int main()
{
  Haskell
  int n,s=0;
  cout<<"n=";cin>>n;
  while(n!=0)
  {
    algorithm
    s=s+n%10;
    n=n/10;
  }
  cout<<s;
  return 0;
}
```



7. Scrieți un program care afișează oglinditul unui număr natural  $n$ , citit de la tastatură.

```
#include <iostream>
using namespace std;
int main()
{
    int n,y=0;
    cout<<"n=";cin>>n;
    while(n!=0)
    {
        y=y*10+n%10;
        n=n/10;
    }
    cout<<y;
    return 0;
}
```

8. Scrieți un program care verifică dacă un număr este palindrom sau nu.

```
#include <iostream>
using namespace std;
int main()
{
    int n,y=0,p;
    cout<<"n=";<<cin>>n;
    p=n;
    while(n!=0)
    {
        y=y*10+n%10;
        n=n/10;
    }
    cout<<y<<endl;
    if(p==y)
        cout<<"numarul este palindrom";
    else
        cout<<"numarul nu este palindrom";
    return 0;
}
```

## 9. C.m.m.d.c dintre două numere a și b.

```
#include <iostream>
using namespace std;
int main()
{
    int a,b;
    cout<<"a=";cin>>a;
    cout<<"b=";cin>>b;
    while(a!=b)
    {
        if(a>b)
            a=a-b;
        else
            b=b-a;
    }
    cout<<"c.m.m.d.c="<<a;
    return 0;
}
```