Structuri repetitive

- Atunci când se dorește repetarea unei acțiuni de un anumit număr de ori este necesară utilizarea unei structuri repetitive. În funcție de numărul repetărilor acestea se clasifică în:
- Structuri repetitive cu număr nedeterminat de pași
  - Cu test inițial
  - Cu test final
- Structuri repetitive cu număr fix de pași

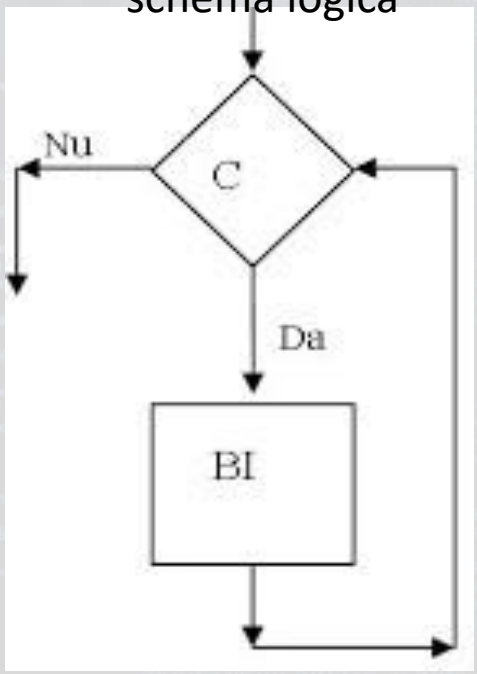
Structura repetitivă cu test inițial

- Numele acestei structuri repetitive provine de faptul că înaintea executării unei instrucțiuni se verifică îndeplinirea unei expresii logice. În limbajul C++ este cunoscută sub denumirea de structura **while**.
- Reprezentarea acestei structuri este:

pseudocod

```
cât timp (expresie logică) execută
    instrucțiune
```

schema logică

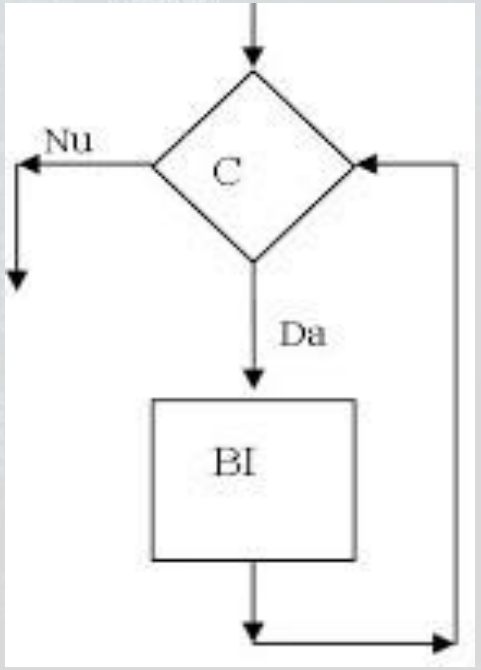


Mod de execuție:

- Pas 1 :Se determină valoarea de adevăr a expresiei logice(C);
- Pas 2: Dacă este adevărată atunci se execută blocul de instrucțiuni(BI) și se revine la pas1
- Pas 3: Dacă expresia logică este falsă atunci se părăsește structura repetitivă.

Observații

- Dacă la prima evaluare a expresia logică(C) este falsă, atunci instrucțiunea(BI) nu va fi executată nici măcar o singură dată și se va părăsi structura repetitivă;
- Evaluarea expresiei logice(C)se face la fiecare reluare a structurii si este recomandat ca forma ei să fie cât mai simplă;
- Utilizarea acestei structuri este recomandată atunci când nu se cunoaște numărul de repetări ale buclei și atunci când valoarea expresiei logice(C) poate fi falsă încă de la intrarea în structură;
- Instrucțiunea (BI) poate conține orice fel de structură de control, deci și o altă structură repetitivă cu test inițial (while-uri imbricate);
- Buclele infinite nu vor fi sesizate ca erori de compilare;
- Utilizatorul este obligat să fie atent în evitarea formării de bucle infinite, și să verifice că după un anumit număr de repetări valoarea de adevăr a expresie logice (C) să devină falsă.



- Instrucțiunea while este instrucțiunea C++ folosită pentru a implementa în limbaj de programare o structură repetitivă cu număr necunoscut de pași și test inițial.
- Sintaxa instrucțiunii este:

```
while (Expresie)
Instrucțiune;
```

```
while (Expresie)
{
Instrucțiune1;
Instrucțiune2;
..............
}
```

Mod de execuție:

- Se evaluează `Expresie`
- Dacă `Expresie` este nenulă
  - Se execută `Instrucțiune`;
  - Se reia pasul 1.
- Dacă `Expresie` este nulă, se trece la instrucțiunea de după **while**.

Observații

- `Instrucțiune;` se execută cât timp `Expresie` este nenulă – condiție adevărată.
- Dacă Expresie este de început vidă, `Instrucțiune;` nu se execută deloc.
- `Instrucțiune;` poate fi orice fel de instrucțiune, dar una singură. Dacă sunt necesare mai multe instrucțiuni, se va folosi instrucțiunea compusă.
- Este necesar ca cel puțin o variabilă care apare în `Expresie` să-și modifice valoarea în `Instrucțiune;`. Altfel se obține o buclă infinită.

Probleme cu cifrele unui număr

- Frecvent se pune problema prelucrării cifrelor unui număr și pentru realizarea acestuia se pune problema **separării** și **analiza individuală** a **fiecărei cifre** dintr-un număr dat. Algoritmul propus va prelucra **cifrele numărului** dat **de la dreapta la stânga** (adică de la cifra unităților către prima cifră).
- Acestă separare se realizează **prin împărțiri succesive la 10**, unde ultima cifră este restul împărțirii numărului la 10, iar eliminarea ultimei cifre se face determinând **câtul împărțirii numărului la 10**.

```
cin>>n;
inițializare variabile de manevră (sume, contoare cu valoarea 0, produse cu valoarea 1
prelucrare caz n=0
while(n)
{ ……;//prelucrează ultima cifră (n%10)
n=n/10;//  ”taie” ultima cifră din numărul n
}
```

Atenție!

- La sfârșitul acestui algoritm valoarea lui **n** este **0**
- Un caz particular îl reprezintă situația în care inițial valoarea lui **n** este **0**, el trebuie prelucrat separat înainte de instrucțiunea **while**
- Numărul de operații efectuate de algoritm este egal cu numărul cifrelor lui **n**

1. Numărul de cifre a numarului x

```
#include <iostream>
using namespace std;
int main()
{
int x,nr=0; //Se folosesc două variabile întregi: x este numărul care se prelucrează și nr este variabila ce afișează câte cifre are numărul x
cin>>x; //Se citește variabila de intrare, x
if(x==0) //Se verifică dacă numărul citit x este 0
  nr=1; //caz în care variabila nr va primi valoarea 1 (nu există numere fără cifre…)
else //în cazul în care numărul citit x este diferit de 0, se vor prelucra cifrele numărului
while(x)//cât timp numărul x este mai mare decât 0
{
  nr++; //variabila nr va crește cu 1
  x=x/10;//se elimină ultima cifra a numărului
}
cout << nr; //se afișează variabila nr
return 0;
}
```

2. Suma cifrelor unui număr x

```
#include <iostream>
using namespace std;
int main()
{
int x,s=0; //variabilele folosite: numărul care se prelucrează x și suma cifrelor s, variabilă inițializată cu 0
cin>>x; //se citește x
if(x==0)//dacă numărul citit este 0
   s=0; //suma cifrelor numărului este 0
else //dacă numărul citit este diferit de 0
   while(x)//cât timp x este mai mare decât 0
   {
      s=s+x%10; //s primește valoarea vechiului s la care se adaugă ultima cifră a numărului x
      x=x/10;//se ”taie” ultima cifră a numărului x
   }
cout << s; //se afișează suma cifrelor numărului x
return 0;
}
```