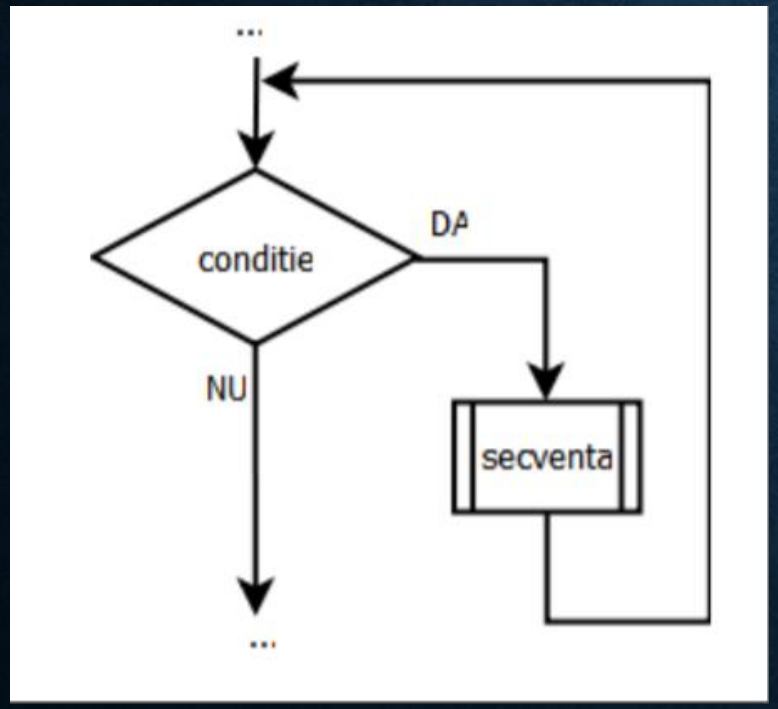
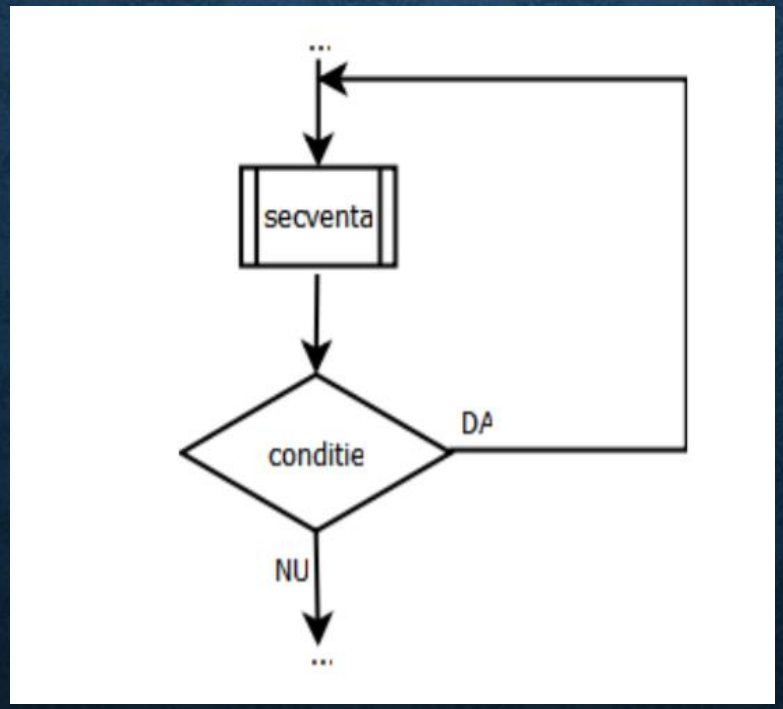
STRUCTURA REPETITIVĂ

Structură repetitivă cu test inițial

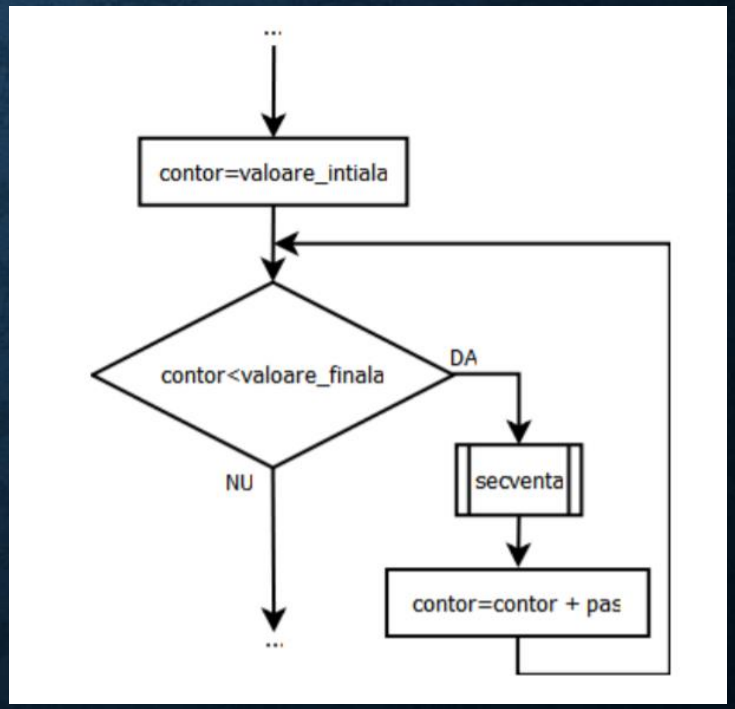
- **Structura repetitivă este un set de instrucțiuni care se repetă până când o condiție nu mai este îndeplinită.**
- Structura repetitivă poate fi:
  - Structură repetitivă cu test inițial
  - Structură repetitivă cu test final
  - Structură repetitivă cu un număr cunoscut de pași (sau structura repetitivă cu contor)



Structură repetitivă cu test inițial



Structură repetitivă cu test final



Structură repetitivă cu număr cunoscut de pași

STRUCTURA REPETITIVĂ CU TEST INIȚIAL

- Structura repetitivă cu test inițial are specificul de a verifica o condiție înainte de a executa primul ciclu de instrucțiuni. Forma generală a structurii repetitive cu test inițial este:

cât timp (condiție)
execută ...

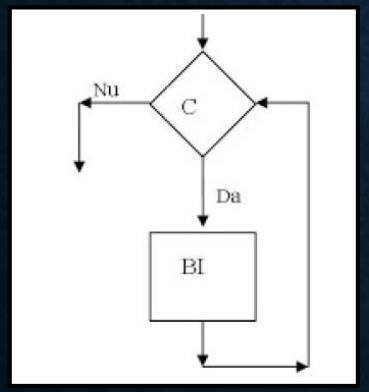
- În C++ structura repetitivă cu test inițial este structura while:

```
while (condiție)
{
...
}
```

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:

Schemă logică	Limbaj pseudocod	Instrucțiune C++

	cât timp (expresie logică) execută
	instrucțiune
	sfarsit_cat timp
		while(conditie)
		instrucțiune
		
		Sau
		
		while(condiție)
		{instructiune1;
		instructiune2;
		...
		}

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 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.

PROBLEME CU CIFRELE UNUI NUMĂR

- Una dintre cele mai frecvente aplicații de liceu se referă la prelucrarea cifrelor unui număr. De regulă această prelucrare presupune extragerea ultimei cifre din număr (cea mai din dreapta), prelucrarea ei și eliminarea acesteia din numărul dat. Acest mecanism continuă până la terminarea tuturor cifrelor din numărul dat. Dar în unele aplicații pot exista și cazuri particulare, pe care trebuie să le identificăm și să le rezolvăm ca atare. Fie un număr natural cu k cifre reprezentat matematic astfel:

$$x = \overline{c_k\, c_{k-1} \ldots c_3 c_2\, c_1}$$

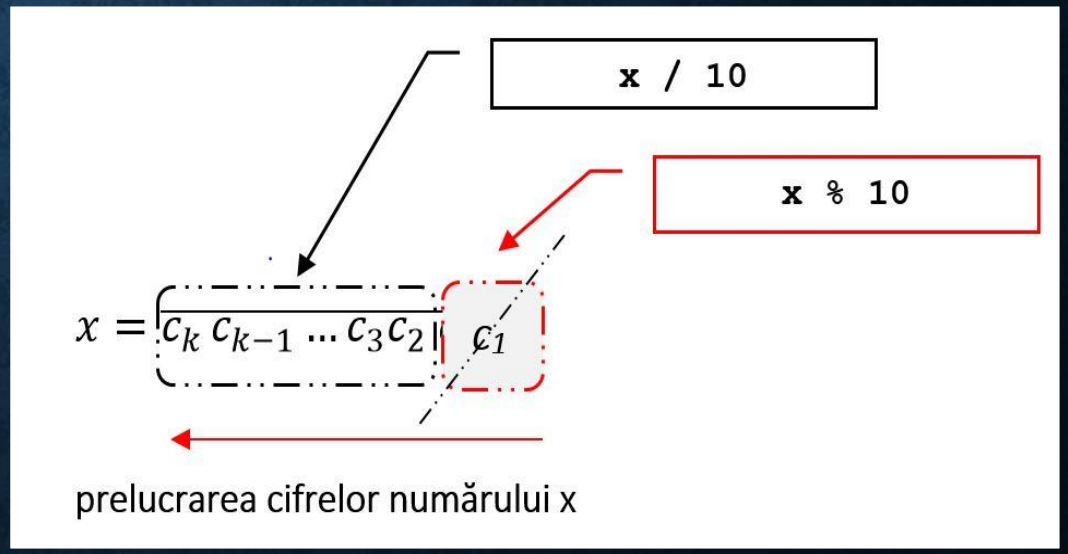
- unde:
  - c1 – este cifra unităților din numărul x
  - c2 – este cifra zecilor din numărul x
  - ...
  - ck – este cea mai semnificativă cifră din numărul x

- Numărul natural x va fi prelucrat cifră cu cifră , parcurgând cifrele numărului de la dreapta la stânga ca în figura următoare:
- Algoritmul în limbaj natural este:

1. citește x
2. cat timp (NU am prelucrat toate cifrele lui x) executa
   a. extrage din x ultima cifra: uc = x%10
   b. prelucrează uc in funcție de cerință din problema
   c. elimina uc din x: x=x/10

sfârșit_cât_timp

3. afișare rezultat



prelucrarea cifrelor numărului x

În C++:

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

Atenție!:

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

Problema 1. Se dă un număr natural n. să se determine numărul de cifre ale acestuia.
Raționament: cât timp n este mai mare decât 0 creștem numărul de cifre și îl împărțim pe n la 10.

În preudocod

```
START
  citește n;
  nrcifre = 0;
  dacă n=0
    nrcifre=nrcifre+1;
  cât timp (n > 0)
     nrcifre = nrcifre + 1;
     n = n / 10;
  sfârșit cât timp;
  scrie nrcifre;
STOP
```

În C++

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int n,nrcifre=0;
6      cin>>n;
7      if(n==0)
8          nrcifre++;
9      else
10         while(n)
11         {
12             nrcifre++;
13             n=n/10;
14         }
15     cout<<nrcifre;
16     return 0;;
17 }
```

Problema 2. se dau două numere naturale a și b. să se afișeze pe ecran a^b. restricții: 0<a<10, 0<b<10.
*Raționament: a^b = a * a * ...* a (de b ori). pentru fiecare i <= b avem p = p * a.*

Pseudocod

```
START
citește a, b;
i = 1;
P = 1;
cât timp (i <= b)
  P = P * a;
  i = i + 1;
scrie P;
STOP
```

C++

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int p=1,i=1,a,b;
6      cin>>a;
7      cin>>b;
8      while(i<=b)
9          {
10             p=p*a;
11             i++;
12         }
13     cout<<p;
14     return 0;
15 }
```