TIPUL POINTER
(TIPURI DE DATE SPECIFICE PENTRU ADRESAREA MEMORIEI)

Se știe că o *variabilă* este stocată într-o anumită zonă de memorie caracterizată de adresa acestei zone. Accesul la această variabilă se poate face *prin numele* ei sau *prin adresa* la care este stocată.

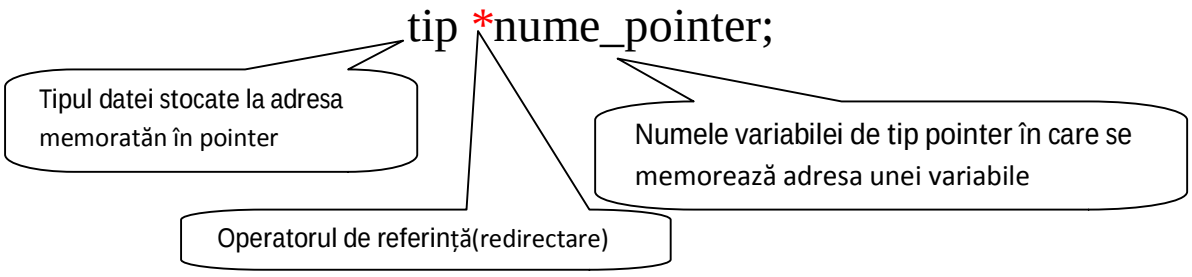
Fiecare variabilă ocupă un număr de octeți determinat de tipul său. Adresa primului octet al variabilei se numeste adresa variabilei.

Pentru a putea manipula datele cu ajutorul adreselor de memorie sunt implementate două tipuri de date:

- Tipul *pointer*
- Tipul *referință*

Pointerul este o variabilă de memorie în care se memorează o adresă de memorie.

Declararea unei variabile de tip pointer se face astfel:



Exemplu:
*int *p1, *p2; //s-au declarat doi pointeri catre tipul int*
*float *adr1, *adr2; //poineri catre tipul float*

Pe varibilele de tip pointer se pot aplica următoarele tipuri de operatori:

a. Operatori specifici
b. Operatori de atribuire
c. Operatori aritmetici
d. Operatori relaționali

a. Operatori specifici

Operatorul **&** - se numește operator de adresare. El se poate aplica pe o variabilă de memorie sau un element de tablou de memorie și ne va da adresa variabilei de memorie(sau a elementului de tablou). Operatia se numește referențiere.

Operatorul * - se numește operator de redirectare. El se poate aplica pe o variabilă de tip pointer și ne va da valoarea variabilei de memorie care se găsește la adresa memorată în pointer. Operația se numește dereferențiere.

b. Operatori de atribuire

Unei variabile de tip pointer i se poate atribui numai o valoare care reprezintă o adresă de memorie:

- o adresă de memorie obținută cu ajutorul operatorului de adresare;
- o altă variabilă de tip pointer de același tip;

- o constantă de tip adresă – adică elementul cu indice 0 al unui tablou sau constanta NULL care corespunde pointerului 0)

c. Operatori aritmetici
   - Adunare, scăderea unei constant (+, -)
   - Incrementare, decrementare (++,--)
   - Scăderea a doi pointeri (-)

Referința

Fie următorul exemplu:

```
int a=1;
int &b=a;
cout<<a;        //se va afisa 1
cout<<b;        //se va afisa 1
```

Acest lucru se traduce prin faptul că variabila **b** este o referință către variabila **a**. Aceasta înseamnă că variabilei **a** i s-a dat prin referință un nume **b**.

Obs: Chiar dacă s-au declarat două variabile a și b se va lucra cu o singură zonă de memorie care va avea același identificator

Exemple:

```
#include <iostream>
using namespace std;
int main()
{
   int *p, a=3, b=9;
   p=&a;
   cout<<p<<endl;        //afiseaza adresa lui a
   cout<<*p<<endl;       //afiseaza 3

   int &ra=a;
   cout<<"ra="<<ra<<endl;    //afiseaza 3
   cout<<&ra;                //afiseaza adresa lui a

   ra++;                  //are ca efect si incrementarea lui a
   cout<<endl<<a;           //afiseaza 4

    ra=b;                  // atentie! va avea ca rezultat modificarea valorii lui a si nu a referintei
                              care ramane in continuare la a
   cout<<endl<<ra<<" "<<a;   //afiseaza 9 9

   p=&b;                  //valoarea pointerului se modifica, va lua adresa lui b
   cout<<endl<<p<<endl;          // va afisa adresa lui b

   int **p1=&p;
      cout<<**p1;              //va afisa 9 (*p1 ne arata ca p1 este un pointer iar **p1 arata ca
                            p1 primeste ca valori adrese de pointeri)


return 0;
}
```

OBSERVAȚII:

1. Este foarte important să facem distincție între un pointer, adresa memorată într-un pointer și valoarea ce se găsește în memorie la acea adresă.
   Fie următoarea declarație:
   int x=1ș
   int *px=&x;

   px – este o variabilă de tip pointer
   px - conține adresa variabilei x
   *px – conține valoarea aflată la adresa memorată în pointer

2. În mod real pointerii se utilizează pentru:
   a. Lucrul cu memoria heap;
   b. Transmitera parametrilor prin referință în cadrul funcțiilor

3. Pointerii se pot aplica si asupra tablourilor sau matricelor. Numele tabloului reprezinta tocmai adresa de început a acestuia in memorie.

   Exemplu de afișare a elementelor unui vector :

```
.....
char v[]="abcd";
.....
cout<<*v; //se va afisa primul element (adica a)
.....
for(int i=0;i<=4;i++)
      cout<<*(v+i); // se va afisa abcd
```

   Exemplu de afisare a primului element a unei matrici;

```
.....
cout<<*(a[i]+j);
.....
```