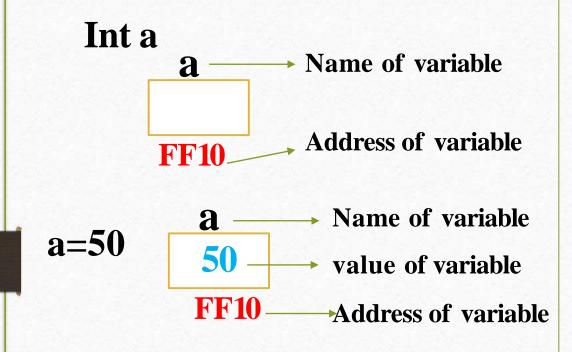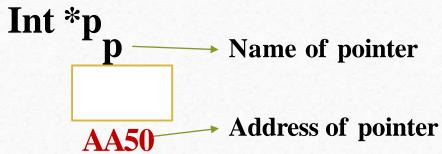# Pointer

- Pointer is a special type of variable which is used to hold the address of another variable .

- Or

- A pointer is a variable that points to another variable . This means that it holds the memory address of another variable

Some pointers must be considered while using pointers

- One pointer can hold only address at a time .

- Before use, pointer must be defined .

- At the time of definition , the symbol * must be used as prefix.

- Pointer holds only address not the value

- **Pointer when used in program after definition then**

   a) * means value of

   b) & means address

Int a

a → Name of variable

FF10 → Address of variable

a → Name of variable

50 → value of variable

FF10 → Address of variable

a=50

Pointer

Syanax Datatype * pointer name;

Int * p;

Int *p

p → Name of pointer

AA50 → Address of pointer

p=&a

p → Name of pointer

FF10 → Address of a

AA50 → Address of pointer

now

Printf(p) it print value of p means means address of a (FF10)

Printf(*p) it print value of a means (50)

Printf(&p) it print addressof p means (AA50)

Generally use two process on pointer

Referencing

Means to give address of another variable to pointer
Eg
Int *p

P=&a

Dereferencing

To access the value of that variable whose address is located on pointer
Eg
Printf("%d",*p)   // it print value of a

```c
#include<stdio.h>
#include<conio.h>
main()
{
    int a=30;
    int *p=&a;

    *p=50;

    printf("%d\n",*p);     /// 50
    printf("%d",a);        //50

    getch();
}
```

```c
#include<stdio.h>
#include<conio.h>
main()
{
    int a=10 ;
    int *p=&a;
    *p=*p+10;
    printf("%d\n",a); // 20
    printf("%d\n",*p);   //20
    getch();
}
```

```c
#include<stdio.h>
#include<conio.h>
main()
{
    int a=50 ;
    int *p=&a;
    int *ptr=p;

    printf("%d\n",*ptr); // 50

    printf("%d\n",*p);    // 50
    getch();
}
```

```c
#include<stdio.h>
#include<conio.h>
main()
{
    int a=50 ;
    int *p=&a;
    *p=*p+a+1;

    printf("%d\n",a);    // 101

    printf("%d\n",*p+1);  // 102
    printf("%d\n",*p);     // 101

}
```