**Pointer**

The **Pointer**  is a variable that stores address of another variable

**Declaring a Pointer**

data\_type \* pointer\_variable\_name;

eg:

int \*ptr\_thing; /\* pointer to an integer \*/

int \*ptr1,thing; /\* ptr1 is a pointer to type integer and thing is an integer variable \*/

double \*ptr2; /\* pointer to a double \*/

float \*ptr3; /\* pointer to a float \*/

char \*ch1 ; /\* pointer to a character \*/

float \*ptr, variable; /\*ptr is a pointer to type float and variable is an ordinary float variable \*/

#include <stdio.h>

int main()

{

int a=10;

printf("%d",&a);

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

return 0;

}

#include <stdio.h>

int main()

{

int a=10; //variable declaration

int \*p; //pointer variable declaration

p=&a; //store address of variable a in pointer p

printf("Address stored in a variable p is:%x\n",p); //accessing the address

printf("Value stored in a variable p is:%d\n",\*p); //accessing the value

return 0;

}

# Structures

A structure creates a data type that can be used to group items of possibly different types into a single type.

‘struct’ keyword is used to create a structure.

### Syntax of struct

struct structureName

{

dataType member1;

dataType member2;

...

};

Following is an example.

struct Person

{

char name[50];

int empNo;

float salary;

};

## Create struct variables

When a struct type is declared, no storage or memory is allocated.

To allocate memory of a given structure type and work with it, we need to create variables.

Here's how we create structure variables:

struct Person

{

char name[50];

int empNo;

float salary;

};

int main()

{

struct Person person1, person2, p[20];

return 0;

}

Another way of creating a struct variable is:

struct Person

{

char name[50];

int empNo;

float salary;

} person1, person2, p[20];

In both cases, two variables person1, person2, and an array variable p having 20 elements of type struct Person are created.

## Access members of a structure

There are two types of operators used for accessing members of a structure.

1. . - Member operator

eg :person1.salary

person2.salary

1. -> - Structure pointer operator

eg :person1->salary= 10000;

person2->salary

