# Pointers

# What is Address in C?

- If you have a variable var in your program, &var will give you it's address in the memory.

- We have used address numerous times while using the scanf() function.

# What is Pointer in C?

- A pointer in C language is a variable that holds a memory address.
- This memory address is the address of another variable(mostly) of same data type.
- In simple words, if one variable stores the address of second variable then the first variable can be said to point towards the second variable.
- Whenever a variable is declared in a program, system allocates a location i.e an address to that variable in the memory, to hold the assigned value.
- This location has it’s own address number.

# Benefits of using pointers in C?

- Pointers are more efficient in handling Arrays in C and Structures in C.
- Pointers allow references to function and thereby helps in passing of function as arguments to other functions.
- Pointers also provide means by which a function in C can change it’s calling arguments.
- It reduces length of the program and it’s execution time as well.
- It allows C language to support Dynamic Memory management.

# Program to see memory address of any variable

Example: -

```
#include<stdio.h>
void main()
{
    int var = 7;
    printf("Value of the variable var is: %d\n", var);
    printf("Memory address of the variable var is: %p\n", &var);
}
```

# Pointer Variables

- The variables which are used to hold memory addresses are called Pointer variables.
- A pointer variable is therefore nothing but a variable which holds an address of some other variable. And the value of a pointer variable gets stored in another memory location.

# Declaration of pointer

- **Syntax-** type *name;

- type is the data type of the pointer, and the name is the name of the pointer variable.

- And the * operator with the name, informs the compiler that the variable is a pointer.

**Examples**

```
int* p;
int *p1;
int * p2;
```

- The data type of the pointer variable should be the same as of the variable to which the pointer is pointing.
- The declaration int *p doesn't mean that a is going to contain an integer value. It means that a is going to contain the address of a variable of int type.

# Pointer Operators

- There are two pointer operators in C, they are:

1) & operator
The & operator returns the memory address of its operand. In the variable ‘a’, the memory address of the variable b will get stored.
e.g. a = &b;

2) * operator
The * operators is the complement of &. This operator returns the value located at the given address. e.g. if ‘a’ contains the memory address of the variable b, then the following line will store the value of the variable b into c.
e.g. c = *a;

# Assigning value to Pointers

- While declaring a pointer variable, if it is not assigned to anything then it contains garbage value. which means it could be pointing anywhere in the memory.
- And that can lead to unexpected errors in your program. Hence, it is recommended to assign some value to it.

**Example –**

```
int *pc, c;
c = 5;
pc = &c;
```

- Here, 5 is assigned to the c variable. And, the address of c is assigned to the pc pointer.

# Get value of variable

- To get the value of the thing pointed by the pointers, we use the * operator.
- **Example –**

```
int *pc, c;
c = 5;
pc = &c;
printf("%d", *pc);          // Output: 5
```

Here, the address of c is assigned to pc pointer. To get the value stored in that address, we used *pc

# Changing Value Pointed by Pointers

## Example1 –

```
int* pc, c;
c = 5;
pc = &c;
c = 1;
printf("%d", c);
printf("%d", *pc);
```

Output : 1

Output : 1

We have assigned the address of c to pc pointer.

Then, we changed the value of c to 1. Since pc and the address of c is the same, *pc gives us 1.

# Changing Value Pointed by Pointers

## Example2 –

```
int *pc, c, d;
c = 5;
d = -15;

pc = &c;
printf("%d", *pc);
pc = &d;
 printf("%d", *pc);
```

Output : 5

Output : -15

We have assigned the address of c to pc pointer.

Since pc and the address of c is the same, *pc gives us 5.

# Null pointer in C

- While declaring a pointer variable, if it is not assigned to anything then it contains garbage value. And that can lead to unexpected errors in your program. Hence, it is recommended to assign a NULL value to it.

- When we assign NULL to a pointer, it means that it does not point to any valid address. NULL denotes the value 'zero'.

- A pointer that is assigned a NULL value is called a NULL pointer in C.

# Null pointer in C

- We can give a pointer a null value by assigning it zero to it.

  E.g. int *ptr = 0;

- We can also use the NULL keyword, which is nothing but a predefined constant for null pointer.

  E.g. int *ptr = NULL;

# Common mistakes when working with pointers

```
int c, *pc;


pc = c;        // pc is address but c is not so it's an Error

*pc = &c;      // &c is address but *pc is not so it's an Error

pc = &c;       // both &c and pc are addresses. // Not an error

*pc = c;       // both c and *pc are values. // Not an error
```

# Working of Pointers

```
#include <stdio.h>
void main()
{
   int* pc, c;

   c = 22;
   printf("Address of c: %p\n", &c);
   printf("Value of c: %d\n\n", c);  // 22

   pc = &c;
   printf("Address of pointer pc: %p\n", pc);
   printf("Content of pointer pc: %d\n\n", *pc); // 22

   c = 11;
   printf("Address of pointer pc: %p\n", pc);
   printf("Content of pointer pc: %d\n\n", *pc); // 11
```

# Explanation of the program

1. int* pc, c; Here, a pointer pc and a normal variable c  both of type int are created.

2. c = 22; This assigns 22 to the variable c. That is, 22 is stored in the memory location of variable c.

3. pc = &c; This assigns the address of variable c to the pointer pc.

4. c = 11; This assigns 11 to variable c.

5. *pc = 2; This change the value at the memory location pointed by the pointer pc to 2.