**Pointers**

Pointers in C are used to store the address of variables or a memory location.

This variable can be of any data type i.e, int, char, function, array, or any other pointer.

**Syntax**

datatype \*var\_name

Example codes

// C program to illustrate Pointers

#include <stdio.h>

void pointer()

{

int var = 20;

// declare pointer variable

int\* ptr;

// note that data type of ptr and var must be same

ptr = &var;

// assign the address of a variable to a pointer

printf("Value at ptr = %p \n", ptr);

printf("Value at var = %d \n", var);

printf("Value at \*ptr = %d \n", \*ptr);

}

// Driver program

int main()

{

pointer();

return 0;

}

**Using pointers**

To access the address of a variable to a pointer, we use the unary operator **&** (ampersand) that returns the address of that variable.

#include <stdio.h>

int main()

{

int x;

// Prints address of x

printf("%p", &x);

return 0;

}

One more operator is **unary \*** (Asterisk) which is used for two things:

1. ***To declare a pointer variable***

// C program to demonstrate declaration of

// pointer variables.

#include <stdio.h>

int main()

{

int x = 10;

// 1) Since there is \* in declaration, ptr

// becomes a pointer variable (a variable

// that stores address of another variable)

// 2) Since there is int before \*, ptr is

// pointer to an integer type variable

int\* ptr;

// & operator before x is used to get address

// of x. The address of x is assigned to ptr.

ptr = &x;

return 0;

}

2.To access the value stored in the address we use the unary operator (\*) that returns the value of the variable located at the address specified by its operand. This is also called **Dereferencing**.

// C program to demonstrate use of \* for pointers

#include <stdio.h>

int main()

{

// A normal integer variable

int Var = 10;

// A pointer variable that holds address of var.

int\* ptr = &Var;

// This line prints value at address stored in ptr.

// Value stored is value of variable "var"

printf("Value of Var = %d\n", \*ptr);

// The output of this line may be different in different

// runs even on same machine.

printf("Address of Var = %p\n", ptr);

// We can also use ptr as lvalue (Left hand

// side of assignment)

\*ptr = 20; // Value at address is now 20

// This prints 20

printf("After doing \*ptr = 20, \*ptr is %d\n", \*ptr);

return 0;

}

**ARITHEMTIC OPERATIONS IN POINTERS**

They include :

* Increment/Decrement of a Pointer
* Addition of integer to a pointer
* Subtraction of integer to a pointer
* Subtracting two pointers of the same type
* Comparison of pointers of the same type.

**Increment/Decrement of a Pointer**

// C program to illustrate Pointer Arithmetic

#include <stdio.h>

int main()

{

// Declare an array

int v[3] = { 10, 100, 200 };

// Declare pointer variable

int\* ptr;

// Assign the address of v[0] to ptr

ptr = v;

for (int i = 0; i < 3; i++) {

printf("Value of \*ptr = %d\n", \*ptr);

printf("Value of ptr = %p\n\n", ptr);

// Increment pointer ptr by 1

ptr++;

}

return 0;

}

EXERCISE

**1.** Write a program in C to show the basic declaration of pointer.

2.Write a program in C to demonstrate the use of &(address of) and \*(value at address) operator.

**Further reading and exercises**

[**https://www.geeksforgeeks.org/c-pointers/**](https://www.geeksforgeeks.org/c-pointers/)

datatype \*var\_nam

datatype \*var\_name;datatype \*var\_name;datatype \*var\_name;datatype \*var\_name;

datatype \*var\_name;