**Pointers**

As an individual who is learning programming language think that pointers are the most difficult concept in this language and it’s quite obvious because for me also it was a huge complication. But it will not be as prodigious as you think when you understand it. To understand it, there is only one solution which is practice. Here I am going to explain a little bit about pointers.

Any language you consider, it is clever to use pointers makes it an excellent language, even pointers make CPP language a popular language. For getting the address of any memory location, we use & sign. And **POINTERS ARE BORN TO STORE ADDRESSES OF VARIABLES.** This means instead of storing the actual variable, it stores the address of the variable. There are two operators used in the case of a pointer, one is & and the other is \*. & represents the address-of operator and \* represents the value at that address. All game is of addresses.

Call-by-value & call-by-reference concept comes from here. When you pass variables to a function, this kind of function call is called call-by-value, and when you pass the address of a variable to a function then this kind of function is called call-by-reference, this is the main difference. And the passed address is accepted by a pointer that’s it. The Call-by-reference concept is used in the case of pointers. We know that variables are stored somewhere in the memory, so instead of passing the value of a variable, we pass the address of a variable.

To understand the concept of pointers, we have to understand how the variables are stored in the memory. There is a space reserved for a variable according to its data type in bytes. When you declare a variable, a name is assigned to that memory location and whatever value you assign will be stored in that memory location. The difference is you are not going to store the value of that location, instead, you will store that memory location address itself. Here the selected memory location is decided by the computer and not the user and it’s not necessary to be the same.

To use a pointer, use the simple concept as follows :

To declare the pointer we use int \*p → Here it tells the compiler that pointer p is used only for storing the address of the integer value and there is another location created for storing the pointer p. And also tells that pointer are capable of storing addresses.

int \*p = &Val; → Here p is a pointer storing the address of the integer value and Val is the variable whose address we want to store in pointer p.

Simple concept to learn :

**&Val → Address of Val**

**p → Address of Val (value at p)**

**&p → Address of p**

**Val → Value at Val**

**\*(&Val) → Value at Val**

**\*p → Value at Val**

One most important thing is that addresses are always going to be whole numbers, therefore, pointers always contain whole numbers. Address, Reference, Memory Location, and cell number are the same.

The main confusion user faces is that the declaration of int \*p does not mean that p is going to contain the integer value. What it means is that p is going to contain the address of the integer value. Simply we can say that pointer is the variable that contains the address of another variable. This pointer may contain the address of another pointer, it is called a double pointer and is initialized as int \*\*k = &p

While using call-by-reference we copy the actual parameters to the formal parameters (Change in formal arguments doesn’t affect actual arguments) but in call-by-reference, we do changes on actual memory locations (Using formal arguments actual arguments can be changed) which is a great advantage in C language and it is done by using pointers. Pointers come with an advanced concept and there are several applications of pointers.

swap(int val1, int val2) → Call-by-value

swap(int &val1, int &val2) → Call-by-reference

when you are using a function, you can return only one value from that function, but in the case of pointers, you don’t even have to return a value from the function because you are doing changes at the actual location. In short, using a call by reference intelligently, we can make a function return more than one value at a time in an indirect manner.

To learn about pointers and start from basics, you can solve the Hacker Rank question Introduction to pointers.

============================================================================  
Assignment   
  
1) write a program for Compare Two Strings using Overloading   
 using classes and object   
Write a program to Swap variables using function overloading  
  
2) write a program for show Counter using Overloading unary operator ++  
using overloading concept.   
3) write a Program to Enter or Insert Data into File  
  
4) write a Program to Read Write Student Details using File Handling  
using file handling   
  
5) write a Program to access protected data member using Inheritance  
6) write a Program to find Area of Rectangle using inheritance  
7 write a Program to show access to private,public and protected using Inheritance  
8) write a Program to illustrates the use of Constructors in single inheritance  
9) Program to Multiply every member by k using class  
10) Swap two numbers and characters using call by address  
  
  
**pointer**  
Write a C++ program to dynamically allocate an integer, a character and a string and assign a value to them.  
  
Write a C++ program to dynamically allocate two two-dimensional arrays of floating values and strings. Initialize its elements.  
  
Write a C++ program to dynamically create an object of a class using the new operator.