FREE eBook

LEARNING pointers

Free unaffiliated eBook created from
Stack Overflow contributors.

#pointers

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: pointers

It is an unofficial and free pointers ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official pointers.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with pointers

Remarks

This section provides an overview of what pointers is, and why a developer might want to use it.

It should also mention any large subjects within pointers, and link out to the related topics. Since the Documentation for pointers is new, you may need to create initial versions of those related topics.

Examples

Getting Started with Pointers

Pointers are variables that store the address of another variable.As language feature they are available in several programming languages like, but not limited to :

- Go
- C/C++
- Ada
- Pascal
- C# (available under certain constraints)
- COBOL
- FORTRAN

To get started with C/C++ pointers , follow these steps

1. Install compiler like Minimalistic GNU for Windows, http://www.mingw.org/wiki/Getting_Started

2. Go to the installation folder of g++ binary via commandline for example:

```
C:\MinGW\bin>
```

3.Create a text file and write this C++ program

```
#include <iostream>

int main () {

   int  pointed=0;
   int* ptr = & pointed;

   std::cout<<"Address of pointed variable is: "<<ptr<<std::endl;

   return 0;
}
```

4. Save as pointer.cpp 5.On the command prompt run the command

   g++ -o pointer.exe -c pointer.cpp

6. In the working directory you will get a executable as pointer.exe , this exe upon running will give some output like

   Address of pointed variable is: 0x7e892dac0a0c

If you receive the above output, you have written your first pointer program

What is a Pointer?

It is basically address of a variable in memory. It allows us to indirectly access a variable. So using pointers we can talk about a variable's address(as well as its value by dereferencing the pointer). They are useful when we want to deal with address of a memory location rather than its value.

Consider the following simple swap function in C:

```
void Swap(int firstVal, int secondVal)
{
 int tempVal = firstVal;
 firstVal = secondVal;
 secondVal = tempVal;
}
```

now in main if we have the following code:

```
.
.
int a = 9,b = 100;
swap(a,b);
//print a and b
.
.
```

The values of a and b would remain unchanged as would be clear by printing their values in the main function. To implement the swap function correctly, instead of passing the values of variables `a` and `b`, we pass the address of variables a and b as:

```
swap(&a,&b);
```

The operator `&`, returns the address of the variable. Its used like this:

```
int *address_of_a = &a;
```

the `int *address_of_a`, states that the variable `address_of_a` points to(stores the address of) an integer variable.

Now our correct swap function would be:

```
void Swap(int *firstaddress, int *secondaddress)
{
 int tempVal = *firstaddress;
 *firsaddress = *secondaddress;
 *secondaddress = tempVal;
}
```

Now the interchanged values would be reflected in main function:

```
int a = 9,b = 100;
swap(&a,&b);
//print
```

You can always dereference the pointer using `*`, if you dont have the original variable. Suppose if in a function you do not have the original variable but have its address in a pointer variable `int *x`. We can simply access the value of the memory address as `value = *x`;

If we did not have pointers we could never emulate **pass by reference** in `C`, because `C` is **pass by value**. But remember we can only **emulate**, because even when we use pointers, the `int *firstaddress, int *secondaddress` are only local pointer variables created, that have the address of variables `a` and `b`.

Read Getting started with pointers online: https://riptutorial.com/pointers/topic/9295/getting-started-with-pointers

Credits

S. No	Chapters	Contributors
1	Getting started with pointers	Community, endTunnel, Sumeet Singh