EBook Gratis

APRENDIZAJE pointers

Free unaffiliated eBook created from
Stack Overflow contributors.

#pointers

Tabla de contenido

Acerca de

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

Capítulo 1: Empezando con punteros

Observaciones

Esta sección proporciona una descripción general de qué son los punteros y por qué un desarrollador puede querer usarlos.

También debe mencionar cualquier tema grande dentro de los indicadores, y vincular a los temas relacionados. Dado que la Documentación para punteros es nueva, es posible que deba crear versiones iniciales de los temas relacionados.

Examples

Primeros pasos con punteros

Los punteros son variables que almacenan la dirección de otra variable. Como característica del lenguaje, están disponibles en varios lenguajes de programación como, pero sin limitarse a:

- Ir
- C / C ++
- Ada
- Pascal
- C # (disponible bajo ciertas restricciones)
- COBOL
- FORTRAN

Para comenzar con los punteros C / C ++, siga estos pasos

1. Instale el compilador como Minimalistic GNU para Windows, http://www.mingw.org/wiki/Getting_Started

2. Vaya a la carpeta de instalación de g ++ binary a través de la línea de comandos, por ejemplo:

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

3.Crear un archivo de texto y escribir este programa en C ++

```
#include <iostream>

int main () {

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

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

```
   return 0;
}
```

4. Guarde como puntero.cpp 5.En el símbolo del sistema, ejecute el comando

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

6. En el directorio de trabajo obtendrá un ejecutable como puntero.exe, este archivo ejecutable le dará una salida como:

   La dirección de la variable señalada es: 0x7e892dac0a0c

Si recibe la salida anterior, ha escrito su primer programa de puntero.

¿Qué es un puntero?

Es básicamente la dirección de una variable en la memoria. Nos permite acceder indirectamente a una variable. Entonces, al usar punteros podemos hablar sobre la dirección de una variable (así como su valor al eliminar la referencia al puntero). Son útiles cuando queremos tratar con la dirección de una ubicación de memoria en lugar de su valor.

Considere la siguiente función de intercambio simple en C:

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

Ahora en main si tenemos el siguiente código:

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

Los valores de a y b permanecerían sin cambios, como quedaría claro al imprimir sus valores en la función principal. Para implementar la función de intercambio correctamente, en lugar de pasar los valores de las variables `a` y `b` , pasamos la dirección de las variables a y b como:

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

El operador `&` , devuelve la dirección de la variable. Se usa así:

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

`int *address_of_a` , indica que la variable `address_of_a` apunta a (almacena la dirección de) una variable entera.

Ahora nuestra función de intercambio correcta sería:

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

Ahora los valores intercambiados se reflejarán en la función principal:

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

Siempre puede eliminar la referencia al puntero usando `*` , si no tiene la variable original. Supongamos que en una función no tiene la variable original pero tiene su dirección en una variable de puntero `int *x` . Simplemente podemos acceder al valor de la dirección de memoria como `value = *x` ;

Si no tuviéramos punteros, nunca podríamos emular **paso por referencia** en `C` , porque `C` es **pasar por valor** . Pero recuerde que sólo puede **emular,** porque incluso cuando usamos punteros, el `int *firstaddress, int *secondaddress` son únicas variables puntero locales creados, que tienen la dirección de las variables de `a` y `b` .

Lea Empezando con punteros en línea: https://riptutorial.com/es/pointers/topic/9295/empezando-con-punteros

Creditos

S. No	Capítulos	Contributors
1	Empezando con punteros	Community, endTunnel, Sumeet Singh