Programación Sobre Redes

L2: Referencias, arreglos, vectors, y streams (parte I)

Nicolás Mastropasqua

April 15, 2020

Instituto Industrial Luis A. Huergo

Contenidos

1. Pasaje de parámetros
2. Arreglos
3. Vectores

Pasaje de parámetros

Desafío!

Ejercicio

Hacer una función **swap**, que tome dos variables de tipo int e intercambie sus valores. Por ejemplo si a = 2 y b = 3, luego de swap(a,b), se tendría a = 3 y b = 2

Pasaje por copia

Pasaje por copia

En C, los parámetros de las funciones son pasados por copia. Esto significa que la función llamada recibe una copia de los valores originales en variables temporales.

¿Cómo hago para que mi función modifique un parámetro?

En C, punteros. En C++, también tenemos referencias.
Las diferencias son sutiles, las veremos más adelante.

Achievement unlocked!

Referencias

Utilizando referencias, ahora podemos resolver el problema de swapear dos variables en C++.

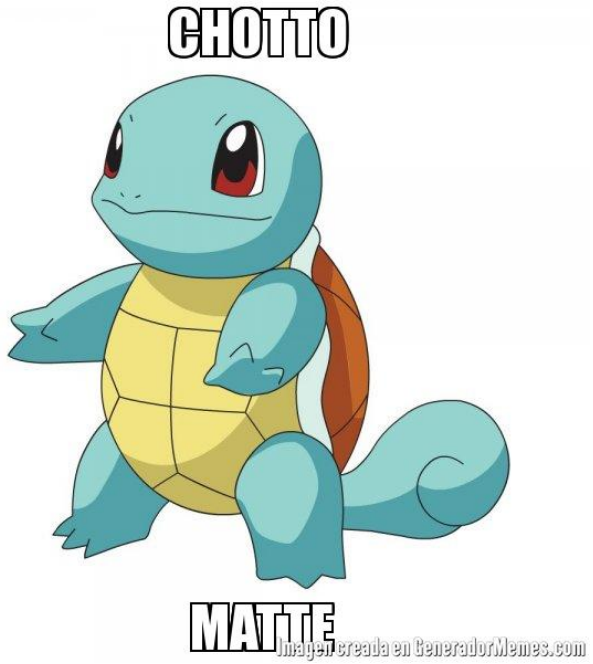
Más aún, podemos hacer que los cambios que realiza una función sobre algún parámetro ”vivan” por fuera de su scope.

Pasaje por referencia

```
int foo(int& a) {
    a += 1;
    return a;
}

int main() {
    int x = 3;
    int y = foo(x);
    y++;
    cout << y << endl; // 5
    cout << x << endl; // 4
}
```

Devolver unan referencia...



Ojo con el scope

Hay que tener cuidado al devolver una referencia a un objeto. Si el objeto muere al finalizar el scope de la función, la referencia no sería válida.

Arreglos

Arreglo convencional

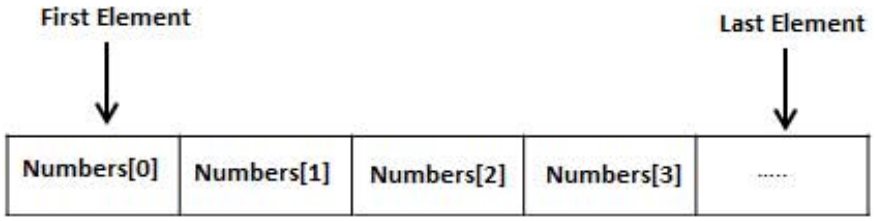


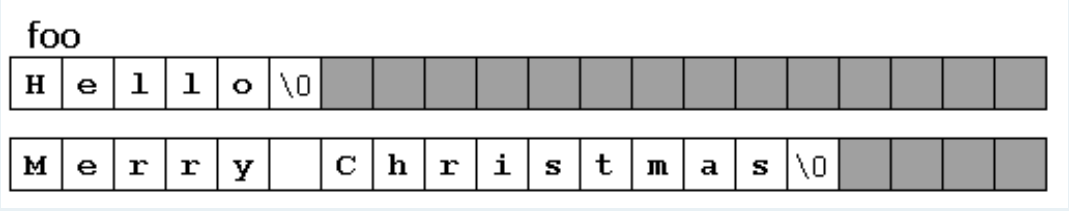
Figure 1: Ejemplo de un arrelgo de enteros llamado Numbers

- Un arreglo ”es como” una lista
- Podemos almacenar elementos, todos del mismo tipo, de forma ordenada
- Podemos acceder o modificarlos utilizando un índice
- Importante: Su tamaño debe especificarse en el momento de su creación. Son ”estáticos”.

Arreglo de carácteres

Podemos pensar que un string no es más que un arreglo de carácteres. En C, está es la forma de interpretar un **string**.

char foo[20]



Notar la importancia del carácter de terminación '\0'

Vectores

Vectores y arreglos

Arreglos

- Los arreglos son soportados nativamente por C
- Tienen tamaño fijo. No se pueden redimensionar.

Vectores

- Sólamente disponibles en C++, utilizando la biblioteca adecuada
- Implementan un arreglo dinámico
- Ojo! Por defecto, los vectors se pasan **por copia**

Existen otras diferencias más sutiles, que iremos viendo con el tiempo. Por ahora, nos interesan estas

Operaciones sobre vector

Creación:

vector<tipoDeDato> nombre;

vector<int> v(4,0);

vector<int> v2(v);

Funciones:

Obtener el tamaño: miVector.size()

Indexar: miVector[posicion]

Agregar al final: miVector.push_back(14)

Sacar del final: miVector.pop_back()

Mas información:

http://www.cplusplus.com/reference/vector/vector/

Ejercicio

Enunciado (ejercicio 1 del taller)

Implementar una función que dado un vector v, devuelve el reverso.

vector<int> reverso(vector<int> v);

¡A trabajar!

Palabras finales

- Con lo que vimos hasta ahora, sumado a sus conocimientos previos, deberían estar en condiciones de empezar con el segundo taller de la materia!
- Los ejercicios están indicados dentro de los archivos correspondientes del taller
- No duden en consultar :)