Apuntes Métodos Computacionales 2

Semana 1 Viernes (31/03/2023)

**Data types:**

Int -> 4 bytes \* 8 bits

32 bits

Espacio de representación: 2^31-1, -2^31

Char

**Prefix and postfix:**

Hay dos tipos

x++

++x

**If statement:**

**While loop:**

También hay do-while

**For loop:**

DUDA: los arreglos se crean vacíos o con valor predet (i.e. 0)? RTA: “vacíos”.

También hay vectores (esos sí cambian len)

**Switch statement:**

*if* secuenciales más sencillos:

El *else* sería *default*

**And/Or**

&& y ||

**Arrays**

En los arrays no se pueden mezclar tipos de datos como en las listas de Python.

DUDA: Insert. RTA: se usa con vectores, para añadir elementos (Python style)

**Multidimensional Arrays:**

**Functions:**

Se debe indicar el tipo de retorno (si no retorna, se coloca *void*).

DUDA: Si se quiere pasar un parametron desconocido, se coloca *auto.* RTA: sirve también para inicializar variables con el tipo de datos adecuado dependiendo de lo que se le asigne (sin tener que especificarlo explícitamente).

**Default parameters:**

Igual que en python.

**OOP:**

Constructor: crea una instancia (objeto) de la clase. Al invocar una clase, se ejecuta automáticamente el constructor.

**Classes:**

Atributos suelen ser privados

Constructor es una función que tiene el mismo nombre que la clase (en Python el constructor es con init)y es privado.

Al crear un objeto se llama al constructor automáticamente.

Además de *public* y *private*, hay *protected* (igual que en Java).

OTRAS DUDAS:

Buscar printf. RTA: printf("format string", arg1, arg2, ...); %d, %f y %s para int, float y string.

Cambiar tamaño de array. RTA: como tal no se puede por medio de una built-in, pero hay dos opciones: crear un nuevo array y trasladar los valores antiguos o usar un arreglo dinámico (ver Chat).

Texto

Descripción generada automáticamente

Semana 2 Miércoles

DUDA: this en c++?

**Punteros:**

&score indica la dirección en memoria donde se almacena el int score. Al imprimierlo, se muestra un número hexadecimal.

Un pointer se define con \*. Una variable puntero guarda la dirección en memoria.

El puntero guarda la dirección.

**Pass by value vs pass by reference:**

Dentro de las funciones no cambia el valor de las variables afuera (en Python sí, pero en C++ no).

Lo que se hace es pasarle la dirección a la función. Dentro de la función se cambia el valor a la variable.

a -> ptrA => &a

a <- ptrA => \*prtA (de referencia)

Una cosa es inicializar el pointer con \* y otra cosa es usar \* para acceder al valor.

**Función swap:**

a=1, b=2 -> a=2, b=1

en Python sería

*def swap (a, b):*

*temp = a*

*a = b*

*b = temp*

en cpp se debe hacer by reference y no by value: