FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE

DEPARTAMENTO DE INGENIERIA Y ARQUITECTURA

Guía 3 Investigación

Docente:

Ing.Carlos Stanley Linares Paula

Instructor:

Br.Juan Carlos Zepeda Abrego

Alumno/a:

Tepas Mazariego, Kenia Stephanie TM17013

Operador sizeof

El operador sizeof es el operador más común en C. Es un operador unario en tiempo de compilación y se usa para calcular el tamaño de su operando. Devuelve el tamaño de una variable. Se puede aplicar a cualquier tipo de datos, tipo flotante, variables de tipo puntero.

Cuando sizeof () se usa con los tipos de datos, simplemente devuelve la cantidad de memoria asignada a ese tipo de datos. La salida puede ser diferente en diferentes máquinas, como un sistema de 32 bits puede mostrar una salida diferente, mientras que un sistema de 64 bits puede mostrar diferentes tipos de datos.

El operador sizeof informa del tamaño de almacenamiento utilizado por cualquier objeto, sea un tipo básico o derivado. El operador unitario sizeof tiene dos formas posibles de sintaxis:

sizeof (nombre-de-tipo)

Un puntero es una variable que contiene la dirección de memoria de un dato o de otra variable que contiene al dato en un arreglo. Esto quiere decir, que el puntero apunta al espacio físico donde está el dato o la variable. Un puntero puede apuntar a un objeto de cualquier tipo, como por ejemplo, a una estructura o una función. Los punteros se pueden utilizar para referencia y manipular estructuras de datos, para referencia bloques de memoria asignados dinámicamente y para proveer el paso de argumentos por referencias en las llamadas a funciones.

Para la utilización de size en punteros se utiliza en la inicialización de el.

Hay varias maneras de inicializar un puntero. Una ya ha sido vista en los ejemplos del punto anterior (pchar = &a;). Ejemplo:

```
#include <stdio.h>

#include <malloc.h>

void *malloc( size_t size );
```

donde 'size' es el numero de bytes que queremos reservar de tipo 'void', es decir, de cualquier tipo.

```
char *pchar;

int  *pint;

pchar = malloc (6); /* pchar apunta al primer byte de los que se han  reservado */
```

pint = malloc (sizeof(int)*2);

/* pint apunta al primero de los dos enteros que se han reservado

Malloc

La asignación dinámica de memoria en el Lenguaje de programación C, se realiza a través de un grupo de funciones en la biblioteca estándar de C , es decir, malloc , realloc , calloc y free . En C++, se incluyen estas funciones por retrocompatibilidad, pero han sido sustituidas en gran parte por los operadores new y new[].

La función malloc sirve para solicitar un bloque de memoria del tamaño suministrado como parámetro. Devuelve un puntero a la zona de memoria concedida:

void* malloc (unsigned numero_de_bytes);

El tamaño se especifica en bytes. Se garantiza que la zona de memoria concedida no está ocupada por ninguna otra variable ni otra zona devuelta por malloc.

Si malloc es incapaz de conceder el bloque (p.ej. no hay memoria suficiente), devuelve un puntero nulo. El problema de malloc es conocer cuántos bytes se quieren reservar. Si se quiere reservar una zona para diez enteros, habrá que multiplicar diez por el tamaño de un entero.

Free

Cuando una zona de memoria reservada con malloc ya no se necesita, puede ser liberada mediante la función free.

void free (void* ptr);

ptr es un puntero de cualquier tipo que apunta a un área de memoria reservada previamente con malloc.

Si ptr apunta a una zona de memoria indebida, los efectos pueden ser desastrosos, igual que si se libera dos veces la misma zona.