Universidad Autónoma del Estado de México
Centro Universitario Valle de Chalco

Apuntadores

Dra. María de Lourdes López García

Unidad de competencia I
Estructuras de datos
Ingeniería en Computación

Contenido

Introducción

- Las estructuras de datos son altamente utilizadas en aplicaciones donde es necesario optimizar el uso de la memoria.

- **El principal componente es el apuntador que permite acceder a un espacio en memoria que puede ser asignado o eliminado en cualquier instante mientras se ejecuta el programa.**

¿Qué es un apuntador?

Un apuntador es una variable que contiene como valor una dirección en memoria.

La dirección en memoria corresponde a una variable previamente definida y del mismo tipo declarado del apuntador.

¿Qué es un apuntador?

Un ejemplo de la vida cotidiana que corresponde al uso de un apuntador, es un número telefónico. Cuando se llama a una persona por teléfono, se utiliza como referencia el número telefónico que permite acceder un canal directo con dicha persona.

Un apuntador permite acceder al espacio de memoria de una variable, lo que implica tener un canal directo para modificar su valor.

Características de un apuntador

1. Es una variable como cualquier otra que se declara antes de ser usada.

2. Contiene una dirección de memoria de una variable de su mismo tipo previamente declarada o la dirección de un espacio en memoria asignado.

3. Modifica el valor de la variable o del espacio en memoria a la que apunta.

Notación

```
<tipo_de_dato> *<identificador>;
```

```
 int *x;
float *y;
char *z;
```

Operadores

1. El operador (*) es indireccional y accede al espacio en memoria de la variable apuntada.

   *p accede al contenido de la variable apuntada por *p*.

2. El operador (&) obtiene la dirección de la variable indicada.

   &q es la dirección de la variable *q*

Uso de los operadores

int $c = 8$;

int $*d$;

$d = \&c$;

$*d = 10$;

¿cuánto valen c y d?

Uso de los operadores

int $c = 8$; //declaración de la variable entera c con valor 8.

int $*d$; //declaración de una variable d apuntador a un entero.

$d = \&c$; //asignación de la dirección de c a d.

$*d = 10$; //acceso al valor de c a través de d.

- c vale 10 y d contiene la dirección de c.

Consideraciones

- Los apuntadores se enlazan a tipos de datos específicos, de modo que el compilador del lenguaje verificará si se asigna la dirección de un tipo de dato al tipo correcto del apuntador.

- Ejemplo:
  - int x, $*y$;
  - float a, $*b$;
  - $y = \&a$; //es incorrecto ya que a es un real y y es un apuntador a un entero.

  - $b = \&x$; //es incorrecto ya que x es un entero y b es un apuntador a un real.

Asignación de memoria estática

Cuando ya se ha declarado un apuntador y se le ha asignado una dirección, el asterisco delante de la variable indica el contenido en memoria de la variable apuntada.

Este tipo de inicialización es estática, ya que la asignación de memoria utilizada para almacenar el valor es fija y sólo se libera al finalizar la ejecución del programa.

Asignación de memoria estática

- Una variable puede ser apuntada por uno o varios apuntadores de su mismo tipo.
- La memoria asignada a un apuntador de forma estática es cuando se asigna la dirección de una variable previamente definida de su mismo tipo.
- El apuntador a una variable puede cambiar, pero el espacio asignado a la variable apuntada permanece reservado hasta el término de la ejecución del programa.

Asignación de memoria dinámica

- Los apuntadores son declarados con un tipo de dato que es el mismo al que van a apuntar durante la ejecución del programa.
- La asignación de memoria dinámica indica asignar la memoria al apuntador sin vincularlo con una variable en específico.
- La función para asignar memoria a un apuntador es *malloc*.

Asignación de memoria dinámica

- malloc() es una función definida en la biblioteca del Lenguaje C denominada stdlib.h.

- La notación es la siguiente:
  - apuntador = (tipo de dato *) malloc (tamaño en bytes del tipo de dato);

  - int *x;
  - x = (int *) malloc (sizeof(int));

  - *sizeof()* es una función que devuelve la cantidad en bytes de su argumento.

Asignación de memoria dinámica

- Ejemplos:
  - float *ptr;
  - ptr = (float *) malloc (sizeof(float));
  - *ptr = 10;

Liberación de memoria

- La liberación de memoria se realiza cuando al apuntador se le ha asignado memoria de manera dinámica.

- La función que permite la liberación se llama ***free()***.
  - float *ptr;
  - ptr = (float *) malloc (sizeof(float));
  - *ptr = 10;
  - free(ptr); //libera el espacio en memoria asignado al apuntador *ptr*.

Apuntador a NULL

- La inicialización de un apuntador requiere un valor neutro que no afecte las operaciones de asignación y liberación de memoria.
- El valor neutro de un apuntador es **NULL**
- El valor *NULL* o *nulo*, no tiene asignado un espacio en memoria, es decir, no direcciona a algún lado en específico que contenga un dato en memoria.

Apuntador a NULL

- Al inicializar un apuntador con NULL se evita extraer información basura de la memoria.
- int *ptr = NULL;
- float *y = NULL;
- char *z = NULL;

Apuntador a NULL

- Null también es utilizado para validar la asignación correcta de la memoria a un apuntador.
- char *p;
- p = (char *) malloc (sizeof(char));
- if (p==NULL) printf("Error");

Dirección en memoria

- Cuando una variable se declara, se asocia a ella lo siguiente:
  - Un tipo de dato,
  - Un identificador,
  - Una dirección en memoria,
  - Un valor.

Dirección en memoria

- int x;
- x=10;
  - El tipo de dato es entero (int)
  - El identificador es x,
  - La dirección en memoria se obtiene con $\&x$,
  - El valor es 10.

Apuntador de apuntador

- Un apuntador puede contener la dirección de otro apuntador del mismo tipo de dato.

- La declaración de un doble apuntador se realiza con dos asteriscos previo al identificador.
  - int a = 10;
  - int *b;
  - int **c; //declaración del doble apuntador.

  - b = &a; //asignación de la dirección de la variable *a* al primer apuntador.

  - c = &b; //asignación de la dirección de la variable *b* a *c*

Apuntador de apuntador

- int *b es la declaración del apuntador *b* a un entero.
- int **c es la declaración del apuntador *c* que contiene la dirección del apuntador *b* que tiene como valor la dirección del entero *a*.

Apuntador de apuntador

- Los valores del apuntador *c* son los siguientes:
  - El tipo de dato es entero (int)
  - El identificador es *c*,
  - La dirección en memoria se obtiene con &*c*,
  - El valor es &b, es decir, la dirección de *b*.

Apuntador de apuntador

- Es importante notar que:
  - c contiene la dirección de b.
  - $*c$ contiene la direcicón a.
  - $**c$ contiene el valor de a.

Otros ejemplos mas...

01. float a = 10;
02. float *b;
03. float **c;
04. float ***d;

05. b = &a;
06. c = &b;
07. d = &c;

08. ***d = 10;
09. **c = a + 10;
10. *b = *b -20;
11. a = a + 1;

- ¿cuánto vale *a*?

Otros ejemplos mas...

01. float a = 10;
02. float *b;
03. float **c;
04. float ***d;

05. b = &a;
06. c = &b;
07. d = &c;

08. ***d = 10; // a=10;
09. **c = a + 10; // a = a +10;
10. *b = *b -20; //a = a -20;
11. a = a + 1; //a = a + 1;

- ¿cuánto vale *a*? // a = 1

Conclusiones

1. Los apuntadores son elementos fundamentales en el manejo de la memoria.

2. La asignación de la memoria puede ser de forma estática o de forma dinámica.

3. La asignación dinámica permite la liberación de la memoria en cualquier momento de la ejecución del programa.

Conclusiones

1. La asignación estática permite la liberación del apuntador pero el espacio de la variable apuntada permanece reservado.

2. Los apuntadores pueden contener la dirección de otro apuntador, en ese caso de llama doble apuntador.

3. A partir de un apuntador es posible modificar el valor contenido en el espacio de memoria que apunta.

Referencias

- Joyanes, L. y Zahonero I. (1999) Programación en C, metodología, algoritmos y estructura de datos. Mc Graw Hill.

- Marquez, G., Osorio S. y Olvera N. (2011) Introducciíon a la programación estructurada en C. Pearson Education.