Práctica 6. Punteros

Objetivos de la práctica:

- Adquirir la capacidad de programar en lenguaje C.
- Comprender el uso y funcionamiento de los punteros.

1. Los punteros en C

Los punteros nos permiten almacenar datos reservando memoria de forma dinámica. Los vectores y matrices vistos anteriormente nos permitían reservar memoria de forma estática. Cuando al programar se conoce el tamaño de los vectores y matrices se usan vectores y matrices. Cuando no se conoce su tamaño, se usan punteros. En C los punteros se declaran de la siguiente forma:

`tipo_de_datos *nombre_puntero;`

Una vez declarado un puntero, antes de usarlo debemos reservar memoria. Para ello, se utiliza la función `malloc` que está en la librería `stdlib.h` de la siguiente forma:

`nombre_puntero=(tipo_de_datos *)malloc(numero_de_datos_que_queremos_guardar * tamaño_del_tipo_de_datos_que_guardamos);`

Para saber el tamaño de un tipo de datos se puede utilizar la función `sizeof()`

Para acceder al contenido de la dirección de memoria donde apunta un puntero usaremos el carácter `*`

Cuando dejamos de usar el puntero se libera mediante `free(nombre_puntero);`

Por ejemplo:

```
...
int tam,*p;
printf("Introduce cuántos datos vas a almacenar: ");
scanf("%d", &tam);
p=(int *)malloc(tam*sizeof(int));
...
*p=5;
...
free(p);
...
```

El equivalente a acceder a la posición i-ésima de un vector `v[i]` usando punteros sería `*(p+i)`

El equivalente a acceder al elemento i,j de una matriz `m[i][j]` usando punteros sería `*(p+i*numero_de_columnas_de_la_matriz+j)`

2. Ejercicios propuestos

2.1. Ejercicio 1

Realice un programa `maximo.c` en lenguaje C que tenga 1 función `calculamaximo` a la cual se le pasan n números enteros y devuelve el valor máximo. La función `main` leerá el número n, reservará memoria para el puntero y leerá los n enteros por teclado, almacenándolos en el puntero. A continuación, llamará a dicha función e imprimirá el valor que le devuelva la función (el máximo) y los números leídos.

2.2. Ejercicio 2

Realice un programa `minimo.c` en lenguaje C que tenga 1 función `calculaminimo` a la cual se le pasa una matriz de n filas y m columnas de números enteros y devuelve el valor mínimo de dicha matriz. Además tiene una función `inicializamatriz` que generará nxm números aleatorios entre 1 y 10 y los almacenará en una matriz de n filas y m columnas que se le pasa como argumento (también se le tendrá que pasar el número de filas y el número de columnas). La función `main` leerá los números n y m por teclado, reservará memoria para el puntero y llamará a la función `inicializamatriz` que rellenará la matriz con nxm enteros aleatorios entre 1 y 10. A continuación, llamará a la función `calculaminimo` e imprimirá el valor que le devuelva la función (el mínimo) y la matriz de números.

2.3. Ejercicio 3

Realice un programa `minimoFilas.c` en lenguaje C que tenga 1 función `minimoFilas` a la cual se le pasan 1 matriz de tamaño nxm y un vector y calcula el valor mínimo de cada fila de la matriz, almacenando los resultados en el vector que se le pasa como argumento. La función principal leerá el tamaño de la matriz y la inicializará con números aleatorios entre -10 y 10 (se puede usar una función para inicializar la matriz), llamará a la función `minimoFilas` e imprimirá el resultado.

2.4. Ejercicio 4

Realice un programa `multiplicacion.c` en lenguaje C que tenga 1 función `multiplica` a la cual se le pasan 3 matrices de tamaño nxn (A, B y C) y calcula el producto de la matriz A por la matriz B almacenando el resultado en la matriz C. La función principal inicializará la matriz con números aleatorios entre 1 y 10 (se puede usar una función para inicializar la matriz). La función principal imprimirá la matriz resultado.