Ejercicios del Tema 5

Punteros

1. Cambio de dos variables

Escribe un programa que, con una función que trabaja con el paso por referencia, intercambia el valor de dos variables. Emplea el siguiente prototipo:

```
void cambio (int *p, int *q);
```

2. Suma de un vector

Diseña un programa que calcule la suma de un vector usando una función basada en punteros.

```
int sumaVector(int *vector, int n);
```

3. Suma minutos

Construye un programa que añada un incremento en minutos a una hora dada en horas y minutos. Este programa utilizará una función según el siguiente prototipo.

```
int incHora(int *h, int *m, int inc);
```

Al hacer la modificación, esta función deberá respetar que las horas estén entre 0 y 23, y los minutos entre 0 y 59. Además, la función devolverá 1 si el incremento ha supuesto cambio de un día a otro, o 0 en caso contrario.

El programa recibirá por teclado los valores de hora, minutos y el incremento, y mostrará en pantalla el resultado.

4. Ecuación de segundo grado

Escribe un programa que resuelva una ecuación de segundo grado. En primer lugar, debe solicitar al usuario los coeficientes de la ecuación. A continuación, debe resolver la ecuación indicando el tipo de resultado: dos raíces reales, dos raíces complejas conjugadas, una raíz real doble, ecuación lineal, o error en los coeficientes. Debes usar el siguiente prototipo:

```
int raices(float a, float b, float c, float *r1, float *r2);
```

donde `r1` y `r2` son las raíces de la ecuación, y la salida `int` se emplea para indicar el tipo de ecuación.

5. Suma, resta, producto y división de dos números

Escribe un programa que calcule la suma, la resta, el producto, y la división (entregando 0 en caso de división por cero) de dos números reales, utilizando una función según el siguiente prototipo:

```
int foo(float x, float y, float *suma, float *resta, float *prod, float *div);
```

Además del resultado de las operaciones, esta función devolverá 1 si el número x es mayor que y, -1 si x es menor que y, 0 si son iguales. El programa recibirá por teclado el valor de x e y, y mostrará en pantalla los resultados de las operaciones y el indicador de mayor o menor.

6. Cuadrantes

Realice un programa que calcule la distancia entre dos puntos introducidos por el usuario mediante el teclado, y que decida el cuadrante en el que está localizado cada punto. El programa debe recibir por teclado las coordenadas x e y de cada punto, sin escribir ningún mensaje en pantalla. A continuación debe calcular la distancia entre los puntos, y el cuadrante de cada uno de ellos, con una función según el siguiente prototipo:

```
void detPunto(float p1[], float p2[], float *dist, int *cuad1, int *cuad2);
```

En esta función, `p1` es el primer punto, `p2` el segundo, `dist` es la distancia entre los puntos, `cuad1` el cuadrante del primer punto y `cuad2` el cuadrante del segundo punto.

Finalmente, mostrará en pantalla los resultados en ese orden separados por un espacio.

7. Media, máximo y mínimo de un vector de datos

Escribe un programa que calcule el valor medio, el valor máximo y el valor mínimo de los datos contenidos en un vector de números reales empleando punteros. La función de cálculo debe acomodarse al siguiente prototipo:

```
void stats(float *vector, int n, float *media, float *max, float *min);
```

Debes escribir dos versiones diferentes. La primera versión trabajará con un vector de dimensión fija y conocida (p.ej. 5 elementos). La segunda versión debe emplear asignación dinámica de memoria, siguiendo el mismo itinerario que el ejercicio anterior.

Nota: Este ejercicio es equivalente al ejercicio 6 del capítulo 4 (vectores). En aquel caso se usa notación de vectores, y ahora se debe emplear notación de punteros. Además, allí se usan tres funciones independientes, y ahora se debe usar una única función que entrega los tres resultados.

8. Longitud de una cadena de caracteres

Diseña un programa que calcule la longitud de una cadena de caracteres usando una función basada en punteros.

```
int stringLength(char *string);
```

9. Copia de cadenas de caracteres

Diseña un programa que copie dos cadenas de caracteres usando una función basada en punteros. Esta función imita el comportamiento de la función `strcpy` de la librería `string.h` (véase el capítulo 4, diapositiva 40).

```
void copyString(char *to, char *from);
```

10. Movimiento de un punto (estructura)

Construye un programa que sirva para mover un punto contenido en el plano (usando un tipo de datos `punto`) usando funciones basada en punteros a la estructura `punto`. El programa funciona con la interacción del usuario a partir de las teclas 'w' (arriba), 'a' (izquierda), 'x' (abajo), 'd' (derecha), y sus correspondientes mayúsculas. Por ejemplo, si el usuario aprieta la tecla 'w' el punto se desplaza hacia arriba una unidad, y si aprieta 'W' se desplaza cinco unidades hacia arriba. El programa muestra en pantalla las coordenadas del punto tras cada interacción del usuario. El programa termina cuando el usuario pulsa la tecla 'q'.

```
// Función para controlar el desplazamiento horizontal, siendo xy el
// punto, y n el número de unidades a desplazar
void horizontal(punto *xy, int n);
// Función para controlar el desplazamiento vertical
void vertical(punto *xy, int n);
```

11. Asignación dinámica de memoria

Escribe un programa que permita al usuario rellenar un vector de dimensión variable. En primer lugar, el programa pregunta al usuario el número de elementos que tendrá el vector. A continuación, empleando asignación dinámica de memoria, el programa reservará memoria para este vector, y solicitará al usuario los valores del mismo. Finalmente, el programa mostrará el contenido de este vector.

12. Estaciones agroclimáticas

Escriba un programa capaz de procesar información básica de unas estaciones agroclimáticas. Aunque lo habitual es que la información esté disponible en un fichero, en este ejercicio la información será introducida por teclado. En primer lugar, el usuario introducirá el número de estaciones disponibles. A continuación introducirá la información de cada estación separando por punto y coma los datos.

La información de una estación es: identificador numérico, longitud, latitud, y altitud. A partir de esta información el programa debe determinar:

- La estación (representada por su identificador, su longitud y su latitud) situada más al norte (máximo valor de latitud).
- El valor promedio de la altura de todas las estaciones.

El programa terminará mostrando estos resultados en pantalla.