TP N°6: Arreglos

A partir de esta práctica modularice al máximo los programas que se solicitan, separando especialmente las funciones de entrada y salida de datos de aquellas que sólo realizan procesamiento.

Cuando se soliciten solamente funciones, se deberá escribir además un programa simple que verifique el correcto funcionamiento de las mismas.

Ejercicio 1

Detectar errores de compilación en el siguiente programa :

```
#define N 10
#define M 20

int
main (void)
{
      int vectorA[ M * N];
      int vectorB[ -10 ];
      int vectorC[ 10.0 ];

      vectorC[2.5] = 'a';
      vectorB[-1] = 5;
      vectorA['0'] = 10;
      vectorC[vectorA[48]] = 5.5;
      vectorA[1000] = 0;
      vectorA[M * N] = 10;
      return 0;
}
```

Ejercicio 2

Reescribir el ejercicio 10 de la práctica 5 de forma tal que quede separado correctamente el front-end del back-end.

Ejercicio 3

Dado un arreglo lineal de números reales, cuyo indicador de fin de elementos es cero, escribir una función para obtener la mayor diferencia entre dos elementos consecutivos. (En no más de 5 líneas). Tener en cuenta que los números pueden ser negativos. En caso de que el vector tenga un solo elemento deberá retornar como diferencia el valor cero.

Ejercicio 4

Escribir una función que reciba un vector desordenado de números enteros y su dimensión, y construya otro vector eliminando los valores repetidos. La función deberá retornar en su nombre la dimensión del nuevo vector (La función solicitada no debe superar las 10 líneas).

Ejercicio 5

Reimplementar la función anterior para vectores ordenados.

Ejercicio 6

Dado un arreglo ordenado ascendentemente se pide escribir una función que reciba como parámetro de entrada/salida el arreglo y como parámetro de entrada su dimensión y que lo devuelva desordenado, simulando la mezcla de un mazo de cartas o de un bolillero (en no más de 10 líneas).

Ejercicio 7

Hacer una función que reciba dos parámetros de entrada representando arreglos de números enteros positivos, ordenados en forma ascendente y sin elementos repetidos. El último elemento de cada arreglo es -1. La función debe devolver en un tercer parámetro de salida un arreglo ordenado con la unión de los dos primeros, también terminado en -1.

Ejercicio 8

Repetir el ejercicio anterior, teniendo en cuenta que los arreglos de entrada pueden tener elementos repetidos, pero el de salida NO debe tener repeticiones.

Ejercicio 9

Se desea calcular la ***desviación estándar*** de un arreglo de números enteros. Los números del arreglo toman valores entre 0 y 15 inclusive, por lo que para almacenar cada número se utilizarán solo 4 bits, pudiendo almacenar dos números en un solo byte.

Para representar dicho arreglo se utilizará un vector de caracteres, donde cada elemento del vector contendrá dos números (uno en los cuatro bits superiores y el otro en los cuatro bits inferiores). Escribir una función que reciba un arreglo como el mencionado anteriormente y la cantidad de números que contiene y retorne en su nombre la ***desviación estándar*** de los números recibidos.

Ejemplo: Si se define el siguiente arreglo:

```
char arreglo[] = { 0x37, 0xF2, 0x03, 0x4A, 0xFF };
```

Representa al arreglo de los elementos: 3, 7, 15, 2, 0, 3, 4, 10, 15, 15.

Ejercicio 10

Los laboratorios de Propulsión por Reacción tienen la representación del cielo y sus estrellas, digitalizada en una matriz bidimensional de hasta 80 columnas por 20 filas. Cada elemento de la misma representa la cantidad de luz que hay en una zona del cielo con un rango de intensidad entre 0 y 20. En el lugar de coordenadas **(i,j)** del cielo se considera que hay una estrella si el elemento $\mathbf{A_{ij}}$ correspondiente cumple con la siguiente relación:

(A[i,j] + suma de las ocho intensidades circundantes) / 9 > 10

Escribir una función (en no más de 15 líneas) que reciba tres parámetros de entrada representando a una matriz de dichas características y sus dimensiones. Dicha función debe localizar gráficamente las estrellas en la pantalla representando las mismas con el caracter '*'. La función debe ignorar las aristas de la matriz.

Nota: para completar la matriz no hace falta interactuar con el usuario, utilizar números aleatorios.

Ejercicio 11

Escribir ***una función*** que ordene las filas de una matriz de cualquier tamaño, según el valor de una determinada columna. La función recibirá como parámetros la matriz, la cantidad de filas, la cantidad de columnas y el número de columna a tomar como clave de ordenación, teniendo en cuenta que la primer columna es la columna 1 (uno).

Ejemplo:

Si la matriz original fuese

1	2	3	4	5
6	7	8	9	10
3	5	8	2	1
8	1	3	6	7

Si se pide ordenar por la columna 1 :

1	2	3	4	5
3	5	8	2	1
6	7	8	9	10
8	1	3	6	7

Si se pide ordenar por la columna 3 :

1	2	**3**	4	5
8	1	**3**	6	7
3	5	**8**	2	1
6	7	**8**	9	10

A igualdad de valores, el orden de las filas coincidentes es indistinto

Ejercicio 12

Escribir una función que cambie una matriz cuadrada por su traspuesta, recibiendo sólo los siguientes 2 parámetros:

- la matriz cuadrada
- un número entero positivo que indique la dimensión de la matriz

Dicha función debe hacer la conversión sobre la misma matriz recibida, sin usar vectores auxiliares.

Nota: La traspuesta de una matriz se obtiene cambiando cada elemento A_{ij} por el elemento A_{ji}.

Ejercicio 13

Escribir una función que realice el producto de dos matrices cuadradas y lo devuelva en una tercera. El algoritmo de la función que realiza el producto no debe tener más de dos ciclos ***for*** anidados explícitamente, pero sí puede utilizar funciones auxiliares que contengan ciclos (Ninguna de las funciones debe superar las 5 líneas).

Ejercicio 14

Escribir una función que reciba un string con el formato 'dd/mm/yyyy' que representa una fecha y devuelva en tres parámetros de salida el número de día, el número del mes y el año. En caso de que la fecha sea incorrecta retorna el valor cero y no altera los parámetros recibidos, caso contrario retorna 1. (Ninguna función debe superar las 8 líneas).

Ejercicio 15

Hacer una función que reciba como único parámetro de entrada/salida un string y elimine los espacios de más. Por ejemplo, si recibe **"Hola mundo "**, deberá transformarlo en **"Hola mundo"**

Ejercicio 16

Escribir una función que devuelva un subvector de un arreglo de caracteres, recibiendo sólo los siguientes parámetros:

- **arregloIn**: vector de entrada, *null terminated*.
- **arregloOut**: vector de salida *null terminated*.
- **desde, hasta**: valores enteros, representan las posiciones de valores a copiar.
- **dimMax**: dimensión máxima del vector de salida (incluyendo el cero).

Dicha función debe almacenar en arregloOut los elementos de arregloIn cuyos índices estén comprendidos entre **desde** y **hasta** inclusive y colocar el valor cero al final. La cantidad máxima de elementos a copiar está dada por el parámetro **dimMax**. En caso de que la cantidad de valores a copiar supere dimMax, se copiarán solamente dimMax - 1 valores.

Si alguno de los parámetros es erróneo o la posición **desde** esté fuera de los límites del arreglo de entrada, la función debe retornar el valor cero y no alterar el arreglo de salida, caso contrario debe retornar la cantidad de elementos copiados.