Práctica No. 4

Estructuras de control.

Objetivo: Desarrollar aplicaciones de software utilizando estructuras de datos para modelar y manipular la información requerida para la solución de problemas, de forma analítica, propositiva y organizada.

- Arreglos
  - Unidimensionales (vectores)
  - Bidimensionales (matrices)
  - N-dimensionales

Material:
- Computadora Personal (PC)
- Programa Editor de texto (ASCII), compilador GCC

Equipo:
- Computadora Personal

Introducción

Un arreglo es un grupo de posiciones en memoria relacionadas entre sí, por el hecho de que todas tienen el mismo nombre y son del mismo tipo. para referirse a una posición en particular o elemento dentro del arreglo, especificamos el nombre del arreglo y el número de posición del elemento particular dentro del mismo.

En la **figura 1** se muestra un arreglo de enteros llamado ***c***. Este arreglo contiene 6 elementos. Cualquiera de estos elementos puede ser referenciado dándole el nombre del arreglo seguido por el número de posición de dicho elemento en particular entre corchetes([]). El primer elemento de cualquier arreglo es el *elemento cero*. entonces el primer elemento de un arreglo ***c*** se conoce como ***c***[0], el segundo como ***c***[1], el sexto como ***c***[5] y en general, el elemento de orden ***i*** del arreglo ***c*** se conoce como ***c***[***i-1***]. Los nombres de los arreglos siguen las mismas reglas convencionales que los demás nombres de variables.

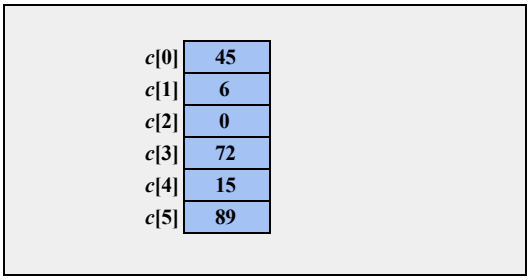

Figura 1. Arreglo Unidimensional

Teoría:

- Arreglos Bidimensionales (matrices)
- Arreglos N-dimensionales

Desarrollo:

El alumno deberá desarrollar un programa para resolver la siguiente problemática. Cada una de las funciones deberá estar en una **biblioteca** (ej. `Calificaciones.h`) implementada por el alumno (basarse en la práctica 2). El archivo que contenga el método main (ej. prac4.c) solo deberá desplegar el menú y ejecutar la selección hecha por el usuario.

1) Implemente un programa para administrar las calificaciones de las prácticas de laboratorio del grupo 531, el cual puede visualizarse como se muestra en la **figura 2**, donde las columnas representan las prácticas y las filas corresponden a los alumnos. Las calificaciones y promedios deberán visualizarse en forma de tabla. Donde **promAx** corresponde al promedio obtenido por el alumno y **promPx** corresponde al promedio por práctica.

2) La tabla deberá ser de 20 alumnos por 10 prácticas, pero flexible a modificación.

3) Las calificaciones deberán ser generadas de forma aleatoria en un rango de 1 a 10.

Núm Práctica / Alumno	0	1	2	3	4	5	6	7	8	9	
0	10	8	6	2	...						promA0
1											promA1
...											promA2
...											...
19								...	7	8	promA19
	promP0	promP1	promP2	….	...	...				promP9	

Figura 2. Calificaciones del grupo 531

4) Realizar función para calcular el promedio por **alumno**

5) Realizar función para calcular el promedio por **práctica**

6) Realizar funciones para calcular i) la media, ii) moda y iii) mediana. Use el algoritmo de ordenamiento de la **figura 3** y despliegue los resultados después de la impresión de la tabla.

```
//Algoritmo de ordenamiento por Burbuja
void bubbleSort(int M[][PRAC], int row, int col)
{
   int i, n, tmp, j=0, swapped = 1;

   n = row*col;

   while (swapped) {
         swapped = 0;
         j++;
         for (i=0; i<n-j; i++) {
               if (M[0][i] > M[0][i + 1]){
                     tmp = M[0][i];
                     M[0][i] = M[0][i + 1];
                     M[0][i + 1] = tmp;
                     swapped = 1;
               }
         }
   }
}
```

Figura 3. Metodo de ordenamiento

7) El programa debe mostrar un menú al usuario de las acciones que puede realizar.
   a) Mostrar tabla de calificaciones
   b) Mostrar calificaciones de un alumno indicado por el usuario
   c) Mostrar calificaciones de una práctica indicada por el usuario
   d) Mostrar moda, mediana y media de calificaciones
   e) Salir

Restricciones

1. El código fuente deberá contener comentarios que indique el objetivo del programa y descripción de instrucciones más relevantes.
2. Evitar nombres de variables y funciones que no reflejen su propósito claramente.
3. Se debe evitar repetir código en medida de lo posible, se deben realizar las funciones de tal forma que puedan ser reutilizables por otras funciones.

Consideraciones

1. La función de ordenamiento modifica el arreglo original debido a que opera en la misma dirección de memoria. Implementar una solución que después de calcular la media, moda o mediana, pueda volver a imprimir las calificaciones originalmente creadas.

Conclusiones y Comentarios.