Práctica No. 6

Estructuras de datos.

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.

Material:

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

Equipo:

- Computadora Personal

Introducción

ESTRUCTURAS (REGISTROS)

Una estructura es un tipo de dato compuesto que permite almacenar un conjunto de datos de diferente tipo. Los datos que contiene una estructura pueden ser de tipo simple (caracteres, números enteros o de punto flotante etc.) o a su vez de tipo compuesto (vectores, estructuras, listas, etc.).

A cada uno de los datos o elementos almacenados dentro de una estructura se les denomina miembros de esa estructura y éstos pertenecerán a un tipo de dato determinado. La definición de una estructura en C corresponde con la sintaxis siguiente:

```
struct miEstructura {
        tipo1 miembro1;
        tipo2 miembro2;
        ...
        tipoN miembroN;
};
```

Los miembros de una misma estructura deben tener nombres únicos, pero dos estructuras diferentes pueden contener miembros con el mismo nombre, sin problema. Toda definición de una estructura debe terminar con un punto y coma.

También es una práctica muy común asignarle un alias o sinónimo al nombre de la estructura, para evitar el tener que escribir "struct miEstructura" cada vez. C nos permite la posibilidad de hacer esto usando la palabra reservada typedef, lo que crea un alias a un tipo de dato ya definido:

typedef struct { ... } MiEstructura;
La estructura misma no tiene nombre (por la ausencia de nombre en la primera línea), pero tiene de alias "MiEstructura". Entonces se puede usar así:

```
MiEstructura variable;
```

Si la estructura es declarada antes de la declaración del alias de tipo entonces se puede usar:

```
typedef struct miEstructura MiEstructura;
```

y se puede usar de esta manera:

```
MiEstructura variable;
```

Note que es una convención, y una buena costumbre usar mayúscula en la primera letra de un sinónimo de tipo.

ESTRUCTURAS ANIDADAS

Una estructura puede estar dentro de otra estructura a esto se le conoce como anidamiento o estructuras anidadas.

Teoría:

Funciones de manipulación de cadenas de la biblioteca de manejo de cadenas.

Desarrollo:
El alumno deberá desarrollar un programa para resolver la siguiente problemática.
Para cada ejercicio propuesto el alumno deberá crear una biblioteca que contenga las funciones que se requieren en cada uno de ellos. Hacer la abstracción de las entidades y representarlos a través de estructuras de datos, baśandose en los ejemplos vistos en clase.

Ejercicio 1. Crear una biblioteca para manipulación de números complejos. En la biblioteca se deberá definir la estructura con la abstracción de un número complejo y contener las siguientes funciones:
-Captura de número complejo
-Impresión del número complejo en su forma rectangular
-Impresión del número complejo en su forma polar
-Suma de números complejos
-Resta de números complejos
-Multiplicación de números complejos
-División de números complejos

Ejercicio 2. Crear una biblioteca para manipulación de números racionales. Definir una estructura con los miembros numerador y denominador. Crear las siguientes funciones para operar números racionales:
-Captura de número racional
-Impresión de número racional
-Suma de número racional
-Resta de número racional
-Multiplicación de número racional
-División de número racionales
Para cada una de las operaciones se deberá expresar el resultado en su mínimo expresión

Ejercicio 3. Crear una biblioteca para manipulación de vectores R3. Definir una estructura con los miembros correspondientes a las componentes i, j y k. Crear las siguientes funciones para operar vectores R3:
-Captura de vector. Formato (ui + vj + wk)
-Impresión de vector. Formato (ui + vj + wk)
-Módulo (operación unaria)
-Suma de vectores
-Resta de vectores
-Producto escalar
-Producto vectorial

Crear un programa externo para cada ejercicio que contenga la función main y probar cada uno de las funciones programadas imprimiendo de forma ordenada los resultados.

Subir el código fuente así como una captura de pantalla de la ejecución de cada ejercicio.

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.

Conclusiones y Comentarios.