PRÁCTICA II

Ejercicio 2.1

Realice una estructura que permita guardar información sobre un punto en el plano (utilizando sus coordenadas en los ejes x e y.) Luego realice un programa que solicite al usuario que se ingresen los datos de dos puntos e indique cual de esos dos puntos se encuentra más lejos del origen de coordenadas (0;0)

Ejercicio 2.2

Utilizando la estructura definida en el punto anterior para representar puntos en el plano, realice un programa que solicite al usuario que se ingresen los datos de diez puntos y almacenar los mismos en un arreglo. Una vez hecho esto calcular la distancia mínima que existe entre:

a) dos puntos consecutivos (según el orden de carga.) Para esto se debe calcular la distancia que existe entre un punto y el punto siguiente del arreglo. Esto debe hacerse para todos los puntos y se deberá encontrar y mostrar la distancia mínima encontrada.

b) dos puntos cualesquiera de los cargados. Para esto se debe calcular la distancia que existe entre un punto y cualquier otro. Esto debe hacerse para todos los puntos y se deberá encontrar y mostrar la distancia mínima encontrada.

Ejercicio 2.3

Utilizando estructuras, desarrollar un programa que permita registrar los datos de un campeonato de fútbol para N equipos (valor constante). Por cada equipo debe almacenar: El nombre, la cantidad de partidos ganados, la cantidad de partidos perdidos, la cantidad de partidos empatados, la cantidad de goles a favor y la cantidad de goles en contra. A continuación se mostrará una tabla ordenada (primero por puntos, luego por diferencia de gol y luego por goles a favor) de los equipos como se ejemplifica a continuación:

Equipo	PTS	PJ	PG	PE	PP	GF	GC	DIF
Milan	10	4	3	1	0	12	3	+9
Real Madrid	9	4	3	0	1	11	2	+9
Ajax	4	4	1	1	2	8	8	0
Arsenal	3	4	0	3	1	5	11	-6
Bayern Munich	1	4	0	1	3	2	14	-12

Ejercicio 2.4

Realizar un programa en el que se desarrolle un juego de video poker según las siguientes reglas:

a. El jugador realizará una apuesta. A los fines prácticos, la apuesta puede ser constante para todas las manos.
b. El jugador recibirá 5 cartas de una baraja francesa.
c. El jugador perderá su apuesta si no ha logrado alguno de los juegos listados o cobrará tantos créditos como indica la tabla multiplicado por la cantidad de créditos apostados:

Tabla de pagos:

Escalera real al As (10, J, Q, K, A)	250
Escalera real (escalera con cartas del mismo palo)	50
Poker	25
Full	9
Color (todas las cartas del mismo palo)	6
Escalera (de diferente palo)	4
Pierna (3 cartas iguales)	3
2 pares	2
Par de J o superior	1

Cada jugador dispondrá de una cierta cantidad de créditos para apostar. El juego finalizará cuando la cantidad de créditos sea 90% más de la cantidad inicial o cuando no disponga de más créditos para apostar.

Si lo desea, puede modificar el juego de modo que permita realizar descarte (para esto se requerirá, obviamente, de la participación del usuario).

Ejercicio 2.5

Realizar un programa en el que se ofrezca al usuario el siguiente menú de opciones:

1- Agregar personas
2- Ver listado ordenado por nombre
3- Ver listado ordenado por documento
4- Ver listado ordenado por país
5- Salir del programa

La opción 1 debe permitir ingresar por teclado hasta una cantidad máxima N (constante) de datos de personas, finalizando documento cero. Si ya existen datos cargados, deberá agregarlos al final del listado. Los datos de las personas consisten en:

- Documento (numérico)
- Nombre (texto)
- País (texto)

Las opciones 2, 3 y 4 deben imprimir por pantalla un listado ordenado con el formato del siguiente ejemplo:

```
Documento           Nombre                    Pais
============================================================
4815162             Matias Zorro              Argentina
3424815             Benjamin Lino             Brasil
1623424             Pedro Chang               China
8151623             Carlos Masanch            Uruguay
```

Ejercicio 2.6

Se dispone de las siguientes estructuras:

```
typedef struct
{
   char nombre[100];
   int legajo, materias[30];
}
t_alumno;
```

```
typedef struct
{
   char nombre[100];
   int codigo;
}
t_materia;
```

```
typedef struct
{
   int legajo_alumo;
   int cod_materia,nota;
}
t_nota;
```

De cada alumno se tiene su nombre, su número de legajo y un listado (en el arreglo) donde están guardados los códigos de las materias que ha cursado.
De cada materia se tiene su nombre y un código numérico (debe ser un número entero mayor que cero).
Las notas están en la tercera estructura, cada una de las cuales contendrá el legajo del alumno, el código de la materia y la nota obtenida en dicha materia. Se utilizará el cero para aquellas materias para las cuales no tenga calificación.

Programar lo siguiente:

1) Una función que reciba como parámetro un arreglo de alumnos y que permita ingresar, por teclado, datos de los alumnos hasta ingresar legajo cero. Se asume que hay una cantidad máxima NA (constante) de alumnos.
2) Una función que reciba como parámetro un arreglo de materias y que permita ingresar, por teclado, datos de las materias hasta ingresar código cero. Se asume que hay una cantidad máxima NM (constante) de materias.
3) Una función que reciba como parámetro un arreglo de notas, un arreglo de alumnos (ya cargado) y un arreglo de materias (ya cargado) y que permita ingresar, por teclado, las notas de cada alumno cargado para cada materia que cursa, imprimiendo, previo al ingreso de la nota, el nombre del alumno, el nombre de la materia. Por ejemplo:
   "*Ingrese la nota de Luis Huergo para la materia Informática II:*"
4) Una función que reciba como parámetro el legajo de un alumno e imprima un listado de las materias que tienen nota cargada y su nota.
5) Una función que reciba como parámetro un código de materia e imprima un listado de todos los alumnos que tienen nota cargada en dicha materia y su nota.
6) Un programa que integre el uso de todas las funciones. Se recomienda usar un juego de datos precargados (inicializados).

Ejercicios Adicionales (no se evaluarán en parcialitos. Usarlos para practicar para parciales y finales)

CODIGO MORSE

Realizar un programa que, dada una frase ingresada por teclado, la convierta a código morse según la simbología indicada a continuación:

A .-	G --.	M --	S ...	Y -.--
B -...	H	N -.	T -	Z --..
C -.-.	I ..	O ---	U ..-	
D -..	J .---	P .--.	V ...-	
E .	K -.-	Q --.-	W .--	
F ..-.	L .-..	R .-.	X -..-	

1 .----	6 -....	Punto	.-.-.-	(AAA)
2 ..---	7 --...	Coma	--..--	(MIM)
3 ...--	8 ---..	?	..--..	(IMI)
4 ....-	9 ----.	:	---...	(OS)
5	0 -----			

Para separar 2 letras entre sí, imprimir un espacio. Para separar 2 palabras entre sí, imprimir 3 espacios. Considere crear una estructura que contenga el char a convertir y su correspondiente conversión a morse. Se recomienda utilizar estructuras para almacenar la codificación.