UNIDAD 3.

Ficheros y aplicaciones MDI

1.- Hacer un programa que permita realizar altas, bajas, modificaciones y consultas de los libros de una biblioteca utilizando un fichero de acceso directo que, como mínimo, tenga los campos *num_referencia y titulo*.

2.- Una empresa tiene contratado el servicio Ibercom de Telefónica. Este servicio almacena en un fichero los siguientes datos de cada llamada:

nº tlf. origen, nº tlf. destino, tiempo de la llamada en segundos y hora y minuto del inicio de la llamada.

a) Se desea hacer un programa que permita: Saber cuáles son los 10 números de teléfono origen que más dinero han gastado.
b) Saber cuáles son los 10 números de teléfono origen que gastan más que la media.
c) Saber cuáles son los 5 números de teléfono destino a los que más llama cada número de teléfono origen.

Los 3 primeros minutos de una llamada local (nº tlf. destino de 7 dígitos) cuestan 15 ptas. y cada minuto adicional 10 ptas. Los 3 primeros minutos de una llamada provincial (nº tlf. destino de 9 dígitos) cuestan 50 ptas. y cada minuto adicional 20 ptas.

3.- Realizar una función que calcule el tamaño de un fichero.

4.- Realizar un programa que utilice un fichero directo con encadenamiento de sinónimos. El programa almacenará los 1.000 empleados de una empresa, tomando como clave el DNI, y utilizando un fichero directo de 1.000 registros y otro fichero de excedentes con 1.000 registros en el que se almacenarán los sinónimos. Se permitirán realizar altas, bajas y consultas.

5.- Se tiene el fichero *f_provincias*, cada uno de cuyos registros consta de los campos:

cod_prov y nombre_prov

El número de registros puede ser cualquiera, pero siempre, al mismo código de provincia le corresponde el mismo nombre de provincia. El *cod_prov* sólo toma valores del 1 al 54.

Hacer un programa que dé como salida el fichero *f_ord_prov* que se obtendrá ordenando el fichero anterior por el campo *cod_prov*.

Ejemplo:

f_provincias	*f_ord_prov*
3 Almería	*3 Almería*
23 Sevilla	*3 Almería*
3 Almería	*23 Sevilla*
54 Zaragoza	*54 Zaragoza*

Se recomienda utilizar un vector de 54 componentes

6.- Se tienen los siguientes ficheros:

f_ventas_mes, ordenado por número de vendedor con los campos:

num_vendedor / cantidad_vendida / mes

f_sueldo, que permita acceso directo por número de vendedor y tiene los campos:

num_vendedor / cantidad_vendida / mes

f_mes, ordenado por mes con los campos:

mes / media_ventas_al_mes

Hacer un programa que lea del fichero *f_ventas_mes* y si la cantidad vendida por ese vendedor ese mes, es mayor que la media de ventas al mes en ese mes (dato obtenido del fichero *f_mes*), aumente el salario anual de ese vendedor en el fichero de sueldo en 50.000 ptas.

El sueldo de cada vendedor puede aumentar varias veces hasta 12 ó 600.000 pesetas, pero en el fichero *f_sueldo* sólo se escribirá una vez por cada vendedor.

7.- Una empresa tiene almacenada las ventas que realiza a sus clientes en un fichero con los campos:

cod_producto, cod_cliente y cantidad_vendida

Puede haber varios registros de un mismo producto (aunque ninguno de ellos tendrá el mismo *cod_cliente*) y puede haber productos sin ventas (que no aparecerán en el fichero). El *cod_producto* varía de 1 a 100. También existe un fichero similar en el que se registran los productos devueltos por sus clientes.

Realizar un programa con los siguientes subprogramas:

a) Subprograma que indique si existe algún error en el fichero de devoluciones. Es decir, si para algún producto existe algún cliente que ha devuelto más unidades de las que ha comprado.
b) Subprograma que calcule cuál es la media de compras por producto, teniendo en cuenta las devoluciones. Utilizará un vector como parámetro.
c) Subprograma que reciba un *cod_producto* y un número N (entre 1 y 10) y devuelva, si existen, N clientes que hayan comprado ese producto.
d) Subprograma que cree un fichero con los códigos de los clientes que tienen ventas y no tienen devoluciones de cada producto. El registro de este fichero tendrá como campos el *cod_producto* y el *cod_cliente*. (Sólo para este subprograma, se supondrá que no hay más de 10 clientes que hayan comprado un determinado producto y se intentará minimizar el número de accesos a fichero).

Sólo se escribirá en pantalla y se aceptarán datos de teclado en el programa principal.

8.- Se tienen los ficheros siguientes:

Fichero *alumnos*, que almacena las notas de cada alumno en cada asignatura. Está ordenado por *codigo_alumno* y sus campos son:

codigo_alumno / nombre / asignatura / nota

Fichero *media* ordenado por *curso*, con 8 registros, uno por curso y los campos:

curso / nota_media_curso

Realizar un programa que:

a) Cree un fichero con los nombres de los alumnos cuya nota media sea mayor que la nota media del curso en el que están (en un curso hay varias asignaturas).
b) Cree un fichero ordenado con las asignaturas en las que estén matriculados menos de 10 alumnos.

El código del alumno es un número entero, el primer dígito de la izquierda indica el curso y los 3 ó 4 siguientes indican el número de matrícula.

Ejemplo:

cod_alumno	curso
345	3
5678	5

Se realizará una función que, dado el código del alumno, calcule el curso en el que se encuentra.

9.- Sean los ficheros:

f_nombre, ordenado por DNI con los campos:

dni / nombre

f_sueldo, ordenado por DNI con los campos:

dni / sueldo

Realizar un programa que cree un fichero con el *dni*, el *nombre* y el *sueldo* de cada persona. Todos los DNI que existen en el fichero de sueldos están en el fichero de nombres, pero existen DNI del fichero de nombres que no están en el fichero de sueldos (personas de nueva incorporación); a estas personas se les asignará un sueldo de 100.000 ptas.

10.- Se quiere gestionar una competición ciclista y se poseen los siguientes ficheros:

f_tiempo, con el tiempo y puntos que tiene cada corredor y los campos: *dorsal*, *horas*, *minutos*, *ptos.*

f_ciclista, con los campos: *dorsal*, *nombre_ciclista*, *pais*, *cod_equipo*. El dorsal *m* está en la posición *m* del fichero. Hay 180 corredores.

f_equipos con los campos: cod_equipo, *nombre_equipo*. El *cod_equipo* de un registro coincide con su posición en el vector. Sólo hay 15 equipos.

Realizar un programa que permita:

a) Mostrar en pantalla el nombre del ciclista mejor clasificado por tiempo, indicando el nombre del equipo al que pertenece.
b) Pedir un cod_equipo al usuario y mostrar los nombres de cinco ciclistas de ese equipo que tengan menos de 30 h y 10 m de tiempo.
c) Los códigos de los tres equipos que más ciclistas tienen.
d) Escribir en pantalla la clasificación por equipos, teniendo en cuenta que para obtenerla hay que sumar los tiempos de los cinco mejores corredores de cada equipo.

e) Escribir en pantalla los nombres de los tres primeros ciclistas de la clasificación de la regularidad (los tres que más puntos tienen)

El número de accesos a los ficheros debe ser el menor posible.

11.- Realizar un editor de texto de tres líneas. Se leerán las teclas pulsadas con getch y se aceptarán los movimientos de los cursores, la tecla de borrado, etc.

Con <F1> se salvará el texto en un fichero y con <F2> se recuperará.

Para detectar cuándo se pulsa una tecla de función (<F1>, <F2> , etc.) o una de movimientos de cursos se deben realizar dos llamadas a la función **getch**, la primera devolverá 0 y la segunda un valor entero que permitirá identificar la tecla pulsada.

Se deberá utilizar también una función para poder escribir en cualquier parte de la pantalla (**gotoxy** o similar)

12.- Realizar un programa que calcule la media, la varianza y cualquier otra medida estadística que se estime oportuna de unos datos numéricos dados por pantalla. Los datos leídos serán almacenados en un fichero para posteriormente poder ser utilizados de nuevo como entrada al programa.

Se permitirá una opción que genere datos aleatorios y trate con ellos como si se hubieran introducido por pantalla; en esta última opción el usuario podrá elegir la posibilidad de que los datos estén o no repetidos.

13.- Realizar un programa que reciba como parámetros una cadena de caracteres y un nombre de fichero y muestre las líneas del fichero en las que aparece dicha cadena de caracteres.

Por ejemplo, si el programa se llama **buscar** la llamada podría ser

buscar palabra fich.dat.

14.- Realizar un programa que permita jugar al ahorcado (adivinar palabras dando letras) contra el ordenador. Se deberá crear, utilizando un editor de textos convencional, un fichero con al menos 50 palabras.

El ordenador elegirá la palabra que hay que adivinar tomándola aleatoriamente del fichero anterior y después mostrará en pantalla un guión por cada letra de la palabra elegida. Después se pedirán al usuario letras hasta que adivine la palabra o pierda. Si la letra dada por el usuario forma parte de la palabra, se cambiará el o los guiones correspondientes por dicha letra; si no está, se añadirá una letra a la palabra **AHORCADO**.

El jugador perderá si da 8 letras (que son las que tiene la palabra AHORCADO) que no pertenezcan a la palabra que hay que adivinar.

15.- Se tienen los siguientes ficheros:

f_hijos, para acceso secuencial con los campos:

dni_hijo, *dni_padre*

Un hijo tiene un solo padre. Un hijo puede ser a su vez padre de otra persona.

f_pers, para acceso secuencial con los campos:

dni, *edad*, *nombre*, *cod_provincia*

Todas las personas que están en el *f_hijos* están en el f_personas.

f_prov, para acceso directo con los campos:

cod_provincia, *nombre_prov*

El *cod_provincia* de un registro del fichero *f_prov* coincide con su posición en el fichero. El *cod_provincia* varía de 1 a 52.

Hacer un programa que muestre en pantalla:

a) El nombre de la persona más vieja
b) Los nombres de los hijos que tienen menos de 20 años que el padre, indicando el nombre de la provincia en la que viven.
c) El número de personas que viven en cada una de las 52 provincias.
d) El nombre de un nieto de una persona. Se pedirá por pantalla el DNI del abuelo. Ninguna persona tiene más de tres hijos.

16.- Se tienen los siguientes ficheros:

f_eleccion, secuencial con los campos:

cod_actividad, cod_socio

f_socios, directo con los campos:

cod_socio, *cant_debida*

El *cod_socio* varía de 1 a 100 y coincide con la posición del registro en el fichero.

f_activ, directo con los campos:

cod_actividad, *precio*, *maximo_num_pers*

El *cod_actividad* varía de 1 a 20 y coincide con la posición del registro en el fichero.

Hacer un programa que:

a) Aumente el campo *cant_debida* a todas las personas que están en el fichero *f_eleccion* con el precio de la actividad.
b) Escriba en pantalla cuántas personas están apuntadas a cada actividad
c) Cree un fichero *f_seleccinados* con los campos: *cod_actividad*, *cod_socio*,

que indique qué socios han sido admitidos en cada actividad, teniendo en cuenta que en una actividad no puede haber más socios que los que hay en el campo *maximo_num_pers*, y que se deberá escoger siempre a los primeros apuntados. No es necesario que el fichero esté ordenado.

17.- La Delegación de Educación tiene almacenados en un fichero los gastos realizados por cada instituto. Cada registro tiene los campos siguientes:

cod_instituto, *cod_gasto*, *gasto_en_ptas*.

Se posee un fichero en el que están almacenados todos los institutos con los campos:

cod_instituto, *direccion* y *gasto_permitido*

El campo *gasto_permitido* estará a 1 si dicho instituto puede realizar un gasto y a 0 en caso contrario. El instituto de código n estará en la posición n.

Realizar un programa que cree un fichero ordenado por *cod_gasto*, que contenga los campos:

cod_gasto, *gasto_total*

donde *gasto_total* será la suma de todos los gastos de institutos que puedan realizar gastos. El código de gasto es un número entero entre 1 y 25. El número de lecturas y escrituras en ficheros debe ser mínimo. Se recomienda utilizar un vector para almacenar los gastos.

18.- Se tiene un fichero de texto con varias líneas. Cada línea tiene 30 caracteres. Los caracteres del 1 al 3 indican el nombre de un elemento

químico, del 5 al 7 el número atómico, del 9 al 16 el peso atómico y del 17 al 25 la proporción con respecto al hidrógeno. Los datos están separados por blancos.

Realizar un programa que lea el fichero y cree un fichero binario con registros del tipo: **struct** {

```
    char nom[3];
int num;
float peso, propor;
};
```

Ejemplos ficticios de líneas del fichero de texto:

H 1 1.34 1.003

Ti 45 45.234 44.955

19.- Se tienen los siguientes ficheros:

*f*_distancias con los campos:

cod_estacion_origen, cod_estacion_destino,

distancia_entre_ambas.

f_estaciones con los campos:

cod_estacion, nombre, num_linea.

Donde el *cod_estacion* para cualquier registro coincide con su posición en el fichero. Los *cod_estacion* varían de 0 a 100 y los *num_linea* de 1 a 10.

Realizar un programa que permita:

a) Mostrar en pantalla: los nombres de las dos estaciones que están a mayor distancia y el número de la línea que tiene más estaciones.
b) Leer el fichero *f_nuevos_nombres* que tiene los campos: *cod_estacion* y *nuevo_nombre*, y cambiar en el *f_estaciones* los nombres de las estaciones por los que aparecen en el *f_nuevos_nombres*. El *f_nuevos_nombres* no está ordenado ni tiene todas las estaciones
c) Leer de teclado una serie de *cod_estacion*, como máximo 25, y mostrar en pantalla cuál es la distancia recorrida para ir de la primera a la última pasando por las demás. Si entre dos *cod_estaciones* dados consecutivamente no hay conexión

se escribirá *NO_SE_PUEDE_LLEGAR*. Para realizar este cálculo se recorrerá el fichero *f_distancias* sólo una vez.

Ejemplo del contenido de los ficheros:

f_distancias	f_estaciones	f_nuevos nombres
1 2 540	0 Ventas 2	22 Lista
3 4 121	1 Goya 2	3 Sol
3 5 232	2 Listas 4	5 Ulsera
4 5 342	3 Soles 2	
	4 Opera 6	
	5 Usada 6	

20.- Se tiene el fichero *f_hospital* con los campos:

num_cama: número entero entre 1 y 200

planta: número entero entre 1 y 7

libre: entero cuyo valor es 0 si la cama no está libre

Realizar un programa que permita mediante un menú efectuar las siguientes operaciones:

a) Escribir en pantalla un listado con una línea por planta y los campos: *planta*, *num_camas_libres*, *num_camas_ocupadas*
b) Pedir por pantalla el nombre de un enfermo, un número de planta y un día de ingreso (entre 1 y 365) y grabar un registro en el fichero *f_enfermos* que tiene los campos:

nombre_enfermo, *num_cama*, *num_dia_ingreso*

Al enfermo se le debe asignar una cama libre de la planta solicitada y si no existiese ninguna en esa planta se le asignará una en planta lo más cercana posible.

c) Pedir al usuario el número del día actual (entre 1 y 365) y mostrar en pantalla el nombre del enfermo que lleva más tiempo ingresado.

21.- Realizar un programa que actualice un fichero maestro *f_maestro* que tiene los códigos y nombres de los socios de un club con el fichero de movimientos *f_movimi* con las nuevas altas (A) , bajas (B) o modificaciones (M) . Ambos ficheros son de texto y están ordenados por el código.

Ejemplo:

fmaestro	*fmovimi*	*fmaesnew*
5 Juan	3 A Oscar	3 Oscar
10 Pepe	5 M Julian	5 Julian
15 Ana	10 B Pepe	15 Ana
20 Susana	10 B Pepe	17 Marta
	20 Susana	

Se supondrá que no existe ningún error en los ficheros (no se modifica o da de baja un registro no existente en el fichero *f_maestro*). El programa recorrerá los ficheros sólo una vez.