



ARCHIVOS BINARIOS Y ARCHIVOS DE TEXTO

OBJETIVOS:

- ✚ Que el alumno, Conozca los diferentes tipos de archivos y su manejo en C.
- ✚ Que el Alumno, Conozca, comprenda el manejo y las operaciones con archivos.
- ✚ Que el alumno, Adquiera prácticas con estos tipos de archivos.
- ✚ Que el alumno, Reconozca el tipo de archivo y los comandos que se debe utilizar en cada caso.
- ✚ Que el alumno, Ejercite habilidades prácticas con operaciones sobre diferentes tipos de archivos.

TEMAS:

- ✚ Apertura de Archivo.
- ✚ Registro de un archivo.
- ✚ Declaración de la Variable, de tipo puntero, para la gestión del archivo.
- ✚ Puntero de registro.
- ✚ Comando de apertura y cierre de archivo.
- ✚ Estructura de un registro en un archivo.
- ✚ Ingresos de datos (Alta).
- ✚ Borrado de un registro (Baja, Eliminación Física y Lógica).
- ✚ Modificación en archivos. (búsqueda y reposicionamiento)
- ✚ Búsqueda y visualización de datos. (consultas)
- ✚ Ordenamiento.
- ✚ Listados de Archivos.

PROBLEMAS RESUELTOS

EJEMPLO 01

OBJETIVO: *Definición y Manejo de ARCHIVOS BINARIOS con datos de tipo Registros.*

ENUNCIADO:

En un negocio se registran las ventas que tuvieron sus empleados de los artículos que se comercializan en un archivo Binario. La configuración *lógica* del registro es:

-código Artículo,
-importe de Venta.

SE PIDE:

- Definir de manera apropiada el registro y realizar la carga de las diferentes ventas por medio de una función *Sin Tipo*.
- En una función *Con Tipo*, recorrer el archivo determinando la cantidad de Artículos cuya venta supere los \$ 1000 y por otro lado, las ventas menores a esta cifra, mostrar ambos resultado en main.
- En una función *Sin Tipo*, Realizar el listado del archivo.

CODIFICACIÓN:

```
#include <stdio.h>
#include <stdlib.h>

/*----- Estructura Lógica del Registro Venta -----*/
struct venta{
    int    cod;
    float  imp;
};

/*----- Prototipos de Funciones -----*/
void cargarArchivo(FILE *arch1);
int  contarMayorMenorQue(FILE *arch1, int &mayorQue);
void listarArchivo(FILE *arch1);

/*----- Bloque Principal -----*/
main (void)
{
    FILE *arch;           //Define "arch" de tipo puntero a un archivo.
    int  menorQue=0, mayorQue=0; //Define los contador de mayores y menores que 1000.

    /*----- Abre y Determina si se creó bien el archivo ----*/
    arch = fopen("articulo.dat","w+b"); //Crea el archivo de Escritura.
    if(arch==NULL){
        printf("\n\n Ocurrido un error al intentar abrir el archivo.....\n");
        system("PAUSE");
        exit(1);
    }

    /*----- Realiza la carga del archivo -----*/
    cargarArchivo(arch); //Llamada.

    /*----- Cuenta la cantidad de ventas mayores y menores a 1000 -----*/
    menorQue = contarMayorMenorQue(arch, mayorQue); //Llamada.

    /*----- Muestra los valores obtenidos -----*/
    printf("Hay %d productos con importe menor a $1000\n",  menorQue);
}
```

```

printf("Hay %d productos con importe mayor a $1000\n", mayorQue);
system("PAUSE");
system("CLS");

/*----- Realiza el listado del archivo -----*/
listarArchivo(arch); //Llamada.
fclose(arch); //Cierra el archivo.

} //Fin de función main.

/*-----*/
void cargarArchivo(FILE *arch1){
    venta regVenta; //Define "regVenta" de tipo Venta(estructura lógica).
    char seguir = 'n'; //Define "Seguir" para determinar el final de la carga.
    int nroReg = 0; //Define "nroReg" para saber cantidad de registro/s ingresados.
    do{
        system("CLS");
        printf("\n\n*****\n");
        printf("Registro Nro: %d", ++nroReg);
        printf("\n\n*****\n\n");
        printf("Codigo : "); scanf("%d", &regVenta.cod);
        printf("Importe: "); scanf("%f", &regVenta.imp);
        fwrite(&regVenta, sizeof(venta), 1, arch1); //Graba el registro.
        _flushall();
        printf("\nIngresar mas datos (s/n):"); scanf("%c", &seguir);
    } while( seguir == 'S' || seguir == 's' );
    printf("\n\n*****\n");
    printf("\n      Fin de la Carga.....");
    printf("\n\n*****\n\t");
    system("PAUSE");
    system("CLS");
}

/* -----
Nombre      : contarMayorMenor(...)
Descripción : Cuenta las ventas mayores e iguales a 1000 y las menores a 1000.
Tipo Función: Con Tipo.
Retorna     : El cantidad de valores que fueron menores a 1000 y por referencia mayor de 1000
Parámetros  : Recibe dos (1)- El puntero del registro. (2)- Variable por referencia. Ventas ma-
yores a 1000.
-----*/

int contarMayorMenorQue(FILE *arch1, int &mayorQue)
{
    venta regVenta;
    rewind(arch1);
    fread(&regVenta, sizeof(venta), 1, arch1); //Leer el primer registro.
    int menorQue = 0;
    mayorQue = 0;
    while( !feof(arch1)) //Termina después de leer el último registro.
    {
        if (regVenta.imp < 1000)
            menorQue = menorQue + 1;
        else
            mayorQue = mayorQue + 1;
        fread(&regVenta, sizeof(venta), 1, arch1); //Continua leyendo.
    }
    Return menorQue;
}

```

Parámetro por referencia.

```

/* -----
Nombre      : listarArchivo(...)
Descripción : Lista todas las ventas Realizadas (registradas).
Tipo Función : Sin Tipo.
Retorna     : Ningún Valor.
Parámetros  : Recibe uno
               1- El puntero del registro.
-----*/

void listarArchivo(FILE*arch1){
    venta regVenta; //crea la variable "regVenta" de tipo venta.
    rewind(arch1);   //Ubica el puntero en el primer registro del archivo.
    fread(&regVenta, sizeof(venta), 1, arch1); //Lee un registro.
    printf("\n\n*****\n");
    printf("\n      Listado del Archivo ");
    printf("\n*****\n");
    printf("\tCodigo \tMonto de Venta \n");
    printf("\n*****\n");
    while(!feof(arch1)) //Termina después de leer el último registro.
    {
        printf("\t %d \t %.2f \n", regVenta.cod, regVenta.imp);
        fread(&regVenta, sizeof(venta), 1, arch1); //Continúa leyendo.
    }
    printf("\n*****\n\t");
    system("PAUSE");
    system("CLS");
}

```

EJEMPLO 02

OBJETIVO: Manejo de *archivo Binario Realizando Alta, Baja Física y Lógica de registros.*

ENUNCIADO:

En una empresa de taxis se registra la información en un archivo binario llamado Taxi.dat, que posee la siguiente configuración de registro:

- ✚ Número de licencia.
- ✚ Patente.
- ✚ Apellido y Nombre del Chofer
- ✚ Fecha de Incorporación o compra del vehículo por la empresa. (*Registro Jerarquizado*).

SE PIDE:

- a) Crear la estructura adecuada. Y utilizando un menú de opciones realizar
- b) Realizar utilizando una función el Ingreso de los datos de una cantidad no determinada de vehículos.
- c) Realizar utilizando una función, el Ingreso un número de licencia y borrar físicamente el registro del archivo. Si el número de licencia, no existe mostrar un mensaje indicando esta situación.
- d) Realizar el Listado de los datos guardados en el archivo.

CODIFICACIÓN:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

```

```

/*----- Estructura -----*/
struct fecha
{
    int dia;
    int mes;
    int anio;
};

/*----- Estructura Jerarquica -----*/
struct RegTaxis
{
    int    nroLicencia;
    char  ApeNomb[40];
    fecha fec; //Miembro "fec" de tipo estructura fecha.
};

/*----- Prototipos Auxiliares -----*/
int menuPrincipal();
void mensaje(char const *cadena);

/*----- Prototipos que dan solución al enunciado -----*/
void registrarTaxis(FILE *taxis); //PUNTO 01.
bool borradoTaxis(FILE *taxis); //PUNTO 02. Devuelve un valor lógico 1si fue borrado.
void listarTaxis(FILE *taxis); //PUNTO 03.

/*----- Bloque Principal -----*/
main()
{
    int N=0, nroLic=0, nOp = 0;
    int nroOp = 0;

    /*----- Abrir y verificar apertura del archivo -----*/
    FILE *ArchTaxis; //Se abre de lectura para conservar
    ArchTaxis = fopen("taxis.dat", "w+b"); //los anteriores registros.
    if(ArchTaxis == NULL)
    { // Evalua, Si hubo error, muestra mensaje y termina.
        system("CLS");
        mensaje("Ocurrio un error en la apertura del Archivo...");
        exit(1);
    }
    do
    {
        nOp = menuPrincipal(); //Llama la función que muestra el menú.
        switch(nOp)
        {
            case 1:
                registrarTaxis(ArchTaxis);
                break;
            case 2:
                bool band;
                band = borradoTaxis(ArchTaxis);
                if(band)
                    ArchTaxis = fopen("taxis.dat", "r+b"); //Se abre de nuevo el archivo, porque
                                                            //Fue cerrado en el borrado físico.
                break;
            case 3:
                listarTaxis(ArchTaxis);
                break;

            case 4:
                system("CLS");
                mensaje("F i n   d e l   P r o g r a m a");
                break;
            default:
                system("CLS");
                mensaje("Ha ingresado un numero no valido");
        }
    }
}

```

```

        break;
    } //Fin del switch().
}while(nOp != 4); //Fin del Ciclo Do
} //Final del main().

/*-----*/
void registrarTaxis(FILE *Taxis)
{
    RegTaxis reg; //crea la variable "reg" de tipo RegTaxis.
    int nroReg=0; //Variable contador del nro de registros almacenado.
    char continua='N';
    do{
        system("CLS");
        printf("\t*****\n");
        printf("\t** Registro Nro %3.0d **\n", ++nroReg);
        printf("\t*****\n\n");
        printf("\n\t Nro Licencia : "); scanf("%d",&reg.nroLicencia);
        _flushall();
        printf("\n\t Apellido y Nombre: "); gets(reg.ApeNomb);
        printf("\n\t Fecha: \n");
        printf("\n\t Dia: "); scanf("%d",&reg.fec.dia);
        printf("\n\t Mes: "); scanf("%d",&reg.fec.mes);
        printf("\n\t Añio: "); scanf("%d",&reg.fec.anio);
        fwrite(&reg, sizeof(RegTaxis), 1, Taxis); //Graba el registro lógico.
        _flushall();
        printf("\n\nContinua Registrando Taxis (S/N): ");
        scanf("%c", &continua);
    }while(continua == 'S' || continua == 's');
    system("CLS");
    mensaje("F i n de la c a r g a"); //Muestra el mensaje.
} //Fin de la función registrarTaxis.

/*-----*/
bool borradoTaxis(FILE *Taxis){
    int nroLicencia;
    bool bandEncontrado; //Variable para indicar si fue borrado o no.
    bandEncontrado = false; //Se inicializa en falso.
    RegTaxis reg;

    /*----- Se crea un archivo Temporal -----*/
    FILE *ArchTemp;
    ArchTemp = fopen("taxisTemp.dat","w+b"); //Se crea un archivo de respaldo.

    /*----- Se muestra un título -----*/
    system("CLS");
    printf("\n\t*****");
    printf("\n\t**");
    printf("\n\t** B O R R A D O **");
    printf("\n\t**");
    printf("\n\t*****");

    /*----- Se pide el ingreso del nro de licencia a buscar -----*/
    printf("\n\n\t Ingrese Nro de Licencia: ");
    scanf("%d", &nroLicencia);
    if (nroLicencia > 0 ){//Si es mayor a cero se busca, en otro caso sale.
        rewind(Taxis); //Ubica el puntero en el primer registro del archivo.
        fread(&reg, sizeof(reg), 1, Taxis); //Lee el primer registro.
        while(!feof(Taxis)){ //Repetir hasta el último registro.
            if(nroLicencia != reg.nroLicencia){ //Si son diferentes, Se guarda en el
                fwrite(&reg, sizeof(reg), 1, ArchTemp); //archivo temporal el registro leído.
            }else{ //si son iguales
                bandEncontrado = true; //Pone en verdadero a la variable, indicando que
            } //fue encontrado y no guarda en archivo temporal el registro
        }
    }
}

```

```

        fread(&reg, sizeof(RegTaxis), 1, Taxis); //Continua leyendo.
    }
} else {
    fclose(ArchTemp); //Cierra el archivo temporal.
}
if (bandEncontrado) {
    fclose(ArchTemp); //Cierro archivo Temporal.
    fclose(Taxis); //Cierro archivo Maestro.
    remove("taxis.dat"); //Borro el archivo Maestro.
    rename("taxisTemp.dat", "taxis.dat"); //Cambio de nombre del Archivo Temporal.
    //Por el del maestro.

    mensaje("Nro de Licencia Encontrada, ha sido eliminado fisicamente");
} else {
    mensaje("Nro de Licencia No Encontrada, No se realizo el borrado");
    fclose(ArchTemp); //Cierro archivo Temporal.
}
system("CLS");
mensaje("F I N    D E L    B O R R A D O ");
remove("taxisTemp.dat"); //Elimina el archivo temporal creado.
return bandEncontrado;
} //Fin de la función borradoTaxis.

/*-----*/
void listarTaxis(FILE *Taxis) {
    RegTaxis reg; //Registro logico.

    /*----- Titulo del listado -----*/
    system("CLS");
    printf("\t*****\n");
    printf("\t**  L I S T A D O   D E   T A X I S   R E T I R A D O S   **\n");
    printf("\t*****\n\n");
    printf("\t*  Licencia \t Chofer\t \t Fecha      **\n\n");
    printf("\t*****\n\n");

    /*----- Listado de los Taxis -----*/
    rewind(Taxis); //Ubica el puntero en el primer registro del archivo.
    fread(&reg, sizeof(reg), 1, Taxis); //Leer el primer registro.
    if (feof(Taxis)) {
        system("CLS");
        mensaje("El Archivo esta vacio\n No se Registro Informacion.");
    } else {
        while (!feof(Taxis)) { //Repite hasta el último registro.
            printf("\t %d      ", reg.nroLicencia);
            printf("\t %s      ", reg.ApeNomb );
            printf("\t %d/%d/%d      ", reg.fec.dia, reg.fec.mes, reg.fec.anio);
            printf("\n");
            fread(&reg, sizeof(RegTaxis), 1, Taxis); //Continua leyendo.
        }
    }
    mensaje("F i n    d e l    L i s t a d o");
} //Fin de la función listarTaxis.

/*-----*/
int menuPrincipal()
{
    int opc=0;
    system("CLS");
    printf("\t*****\n");
    printf("\t**          M E N U       P R I N C I P A L          **\n");
    printf("\t*****\n");
    printf("\t**\n");
    printf("\t**      1-> Registrar      **\n");
    printf("\t**\n");

```

```

printf("\t*****\n");
printf("\t**\n");
printf("\t**      2-> Borrar\n");
printf("\t**\n");
printf("\t*****\n");
printf("\t**\n");
printf("\t**      3-> Listar Choferes\n");
printf("\t**\n");
printf("\t*****\n");
printf("\t**\n");
printf("\t**      4-> S A L I R   del   S I S T E M A\n");
printf("\t**\n");
printf("\t*****\n\n");
printf("  Seleccione Opcion: ");
scanf("%d", &opc);
return opc; //retorna el número de opción seleccionada.
} //Fin de la función menuPrincipal.

/*-----*/
void mensaje(char const *cadena)
{
printf("\n\n\n*****\n");
printf("\n      %s ", cadena );
printf("\n\n*****\n\n\t");
system("PAUSE");
} //Fin de la función mensaje().

```

EJEMPLO 03

OBJETIVO: Manejo de archivo Binario de valores enteros.

ENUNCIADO:

Crear un archivo binario (Multiplo.Dat) e ingresar una serie indeterminada de números entero.

SE PIDE:

- Deberá guardar en el archivo, solo aquellos valores que sean múltiplos de 3.
- Deberá mostrar, luego del ingreso, los valores guardados y
- Deberá mostrar, la cantidad de valores que no se guardaron.

CODIFICACIÓN:

```

#include <stdio.h>
#include <stdlib.h>
main()
{
FILE *archMultiplo; //Se crea un puntero de archive, para manejar los datos de un archivo.
int valIngresado=0;
int canNumNoGua =0;
char opcSalida = 'N';

archMultiplo = fopen("Multiplo.dat", "w+b"); // Se abre el archivo, se asigna el puntero a la variable creada anteriormente.

```



```

if(archMultiplo == NULL) // Si tuvo problema y no se abrió se muestra un mensaje y se cierra el programa.
{
    system("CLS");
    printf("\n\n Se produjo un ERROR al intentar abrir el archivo\n");
    printf("comuníquese con el administrador del Sistema. Gracias");
    system("PAUSE");
    exit(1);
}
do{ // hacer
    printf("\nIngrese un Nro : ");
    scanf("%d", &valIngresado);

    if ( valIngresado%3 == 0) //Determina si es múltiplo de 3
    {
        fwrite(&valIngresado, sizeof(valIngresado), 1, archMultiplo); //Guarda el valor ingresado
    }else{
        canNumNoGua = canNumNoGua + 1; //Cuenta los no guardados
    }
    printf("\n\t Continua: (S)i - (N)o, Ingrese Opcion: "); //Pregunta si continua
    _flushall();
    scanf("%c", &opcSalida); //recibe la respuesta por teclado.
}while ((opcSalida == 'S') || (opcSalida == 's')); //mientras "opcSalida" sea 'S' o 's'.

system("CLS");
printf("\n\n\t\t LISTADO DEL ARCHIVO.....\n\n");
rewind(archMultiplo); //Ubica el puntero en el primer valor que tiene el archivo.
fread(&valIngresado, sizeof(valIngresado), 1, archMultiplo); //Lee un valor.

while ( !feof(archMultiplo) ) //Mientras no se fin de archivo hacer
{
    printf("%d - ", valIngresado);
    fread(&valIngresado, sizeof(valIngresado), 1, archMultiplo);
}

fclose(archMultiplo); //cierra el archivo.
printf("\n\n\t\t\t La cantidad de valores no Guardados es: %d \n\t\t", canNumNoGua);
system("PAUSE");
}

```

Guarda Datos en Archivo

Realiza el listado de datos guardados en el archivo

EJEMPLO 04

OBJETIVO: Manejo de archivo Binario de valores reales.

ENUNCIADO:

Crear un archivo binario e ingresar una serie de **N** de números reales.

SE PIDE:

a) Listar Los valores ingresados.

CODIFICACIÓN:

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    FILE *Arch;
    int N, i;
    float valor;
    /* Abre el archivo de lectura */
    Arch = fopen("archReal.dat", "W+b");
    /* Si no se pudo abrir muestra el mensaje y cierra el programa.*/
    if (Arch == NULL)
    {
        system("CLS");
        printf("\n\n Se produjo un ERROR al intentar abrir el archivo\n");
        system("PAUSE");
        exit(1);
    }
    /* Solicita la cantidad de valores a ingresar */
    printf("Cuantos Valores Reales Desea Ingresar: " );
    scanf("%d", &N);
    /*Procede a la carga y su posterior guardado de los valores en el archivo.*/
    for(i=0; i<N; i++)
    {
        printf("\nIngresa un Nro : ");
        scanf("%f", &valor);
        fwrite(&valor, sizeof(float), 1, Arch);
    }
    /* Limpia la pantalla y se procede al listado de los valores guardado en el archivo. */
    system("CLS");
    printf("\n\n\t\t\t LISTADO DEL ARCHIVO \n\n");
    rewind(Arch); //ubica el puntero en el primer valor.
    fread(&valor, sizeof(float), 1, Arch); //Lee el primer valor del archivo.
    while ( !feof(Arch) ) //Inicia el listado de los datos hasta el fin del archivo
    {
        printf("%.2f - ", valor);
        fread(&valor, sizeof(float), 1 , Arch);
    }
    /* Inicia procedimiento de cierre del archivo y fin del programa.*/
    fclose(Arch); //Cierra el archivo.
    printf("\n Fin del Listado.....\n" );
    system("PAUSE");
}
```

EJEMPLO 05

OBJETIVO: Manejo de Archivo de Texto. Guardar datos tipo texto en un archivo.

ENUNCIADO:

Escriba Un programa que pida al usuario que teclee frases, y las almacena en el fichero frases.txt. Acabará cuando el usuario pulse Intro sin teclear nada. Después deberá mostrar el contenido del fichero.

CODIFICACIÓN:

```
#include<stdio.h>
#include <string.h>
main(){
    FILE *fichTexto;    //Crea un variable de tipo puntero a un archivo.
    char frase[61];     //cadena de 60 caracteres para almacenar la frase introducida.
    int i=0;

    fichTexto= fopen("frases.txt", "w");    //Abre el archivo de modo escritura.
    printf(" PROGRAMA para ESCRIBIR y almacenar FRASES en un archive de texto.\n");
    printf("Cuando quiera salir, simplemente pulse \"Intro\".\n\n");

    //Recibe las frases y las almacena en el archivo.
    do{
        puts("\nEscriba una FRASE:\n(o pulse \"Intro\"). \n");    //Muestra en pantalla.
        gets(frase);    //Recibe una frase, como máximo de 60 caracteres.
        fprintf(fichTexto, "%s\n", frase);    //graba la frase introducida en el archivo.
    }while (strcmp(frase, "") != 0);    //compara la frase con una cadena vacía.
    //Continua si no es vacía la frase.

    printf("He aqui lo que escribio:\n\n");
    fclose(fichTexto);    //cierra el archivo.
    fichTexto = fopen("frases.txt", "r");    //Abre el archivo de solo lectura.

    //Muestra las frases introducidas.
    do{
        fgets(frase, 60, fichTexto);    //Lee del archivo 60 caracteres y lo asigna a var. frase.
        puts(frase);    //Muestra el contenido de la variable frase.
    } while (!feof(fichTexto));    //Sigue leyendo, hasta el final del archivo.
    getchar();
    fclose(fichTexto);    //Cierra el archivo de texto.
}
```

EJEMPLO 06

OBJETIVO: Manejo de Archivo de Texto. Ordenar texto, listado, determinar tamaño de la cadena

ENUNCIADO:

En un archivo de texto se tiene almacenado una lista de **N** nombres de personas que pertenecen al equipo vóley de un determinado colegio.

SE PIDE:

- A) Usando una función **Sin Tipo**, almacenar los Apellidos y nombres de las personas en el archivo.
- B) Usando una función **Sin Tipo**, Realizar un listado.

- C) Usando una función **Sin Tipo**, Ordenar alfabéticamente de menor a mayor las información en el archivo. Como NO SE PUEDE ordenar directamente en el archivo, se deberá usar una estructura que almacene los datos y luego ordenar.
- D) Crear otra lista con los nombres, de menor tamaño a mayor tamaño, en un vector y realiza su listado.

CODIFICACIÓN:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

//Declaración Prototipos de las Funciones Utilizadas.
void cargarLosDatos(int N, FILE *ficApeNom);
void listarArchivo(FILE *ficApeNom);
void ordenarApeNom(int N, FILE *ficApeNom);
void ordenarPorLargoDeCadena(int N, FILE *ficApeNom);
/*----- Bloque Principal -----*/
Tipo de Función: Sin Tipo.
Devuelve: Ningún Valor.
Parámetros: Ninguno.
Descripción: Bloque principal, comienza a ejecutarse el programa.
-----*/
main(){
    int N=0;
    FILE *ficApeNom;

    printf("Cuantos Nombres Desea Ingresar: ");
    scanf("%d",&N);

    cargarLosDatos(N, ficApeNom); //Llamada, cargar los nombres al archivo.
    listarArchivo(ficApeNom);     //Llamada, listar Archivo.

    ordenarApeNom(N, ficApeNom); //Llamada, para ordenar alfabéticamente.
    listarArchivo(ficApeNom);     //Llamada, listar Archivo.

    ordenarPorLargoDeCadena(N, ficApeNom); //Llamada, ordena arreglo por tamaño de cadena

} //Final de la función Principal main().

/*-----*/
void cargarLosDatos(int N, FILE *ficApeNom){
    int i;
    char nombre[20]; //Define variable para recibir los nombres a guardar.
    ficApeNom = fopen("Nombres.txt", "w"); //crea el archivo de texto.
    system("CLS");
    printf("\n\n\t\t\tIngrese los nombres a Registrar\n");
    printf("\t\t\t===== \n\n");
    for(i=0; i<N; i++){ //Ingresa N nombres en el archivo.
        printf("\n\t Nombre %d : ",i+1);
        _flushall();
        gets(nombre);
        fprintf(ficApeNom, "%s\n", nombre); // Guarda y salta a la linea siguiente con (\n).
        printf("\n");
    }
    fclose(ficApeNom);
    printf("Final de la carga.....\n");
    system("PAUSE");
    system("CLS");
} //Fin de la función cargarNombres().
```

```

/*-----*/
void listarArchivo(FILE *ficApeNom){
    char nombres[20];
    ficApeNom = fopen("Nombres.txt", "r"); //Abre el archivo de solo lectura.
    fgets(nombres, 20, ficApeNom); //Lee del archivo 20 caracteres o hasta fin de línea y se lo asigna a la variable "nombres".
    printf("Se Procede a listar el archivo.....\n");
    while (!feof(ficApeNom)){ //Sigue leyendo, hasta el final del archivo.
        puts(nombres); //Muestra el contenido de la variable nombres.
        fgets(nombres, 20, ficApeNom); //Lee del archivo 20 caracteres o hasta fin de
    } //el fin de línea o de archivo.
    fclose(ficApeNom);
    printf("Final del Listado.....\n");
    system("PAUSE");
    system("CLS");
} //Fin de la función listarArchivo().

/*-----*/
void ordenarApeNom(int N, FILE *ficApeNom){
    typedef char nombPers[20]; //Defino un tipo nuevo de dato.
    nombPers vecAux[N]; //arreglo Auxiliar que contendrá los nombre a ser ordenado.
    nombPers cadAux; //cadena auxiliar. Usada para ordenar el arreglo.
    int i;
    bool bandOrdenado = false; //Variable bandera, usada para saber si esta ordenado el arreglo.
    ficApeNom = fopen("Nombres.txt", "r"); //Abre de lectura el archivo.
    system("CLS");
    printf("\n\n INICIO del Ordenamiento Alfabético de menor a mayor.....\n");
    system("PAUSE");
    /*----- Pasar los elementos del archivo al vector auxiliar (vecAux). -----*/
    i=0;
    fgets(vecAux[i], 20, ficApeNom); //Lee el primer nombre y lo guarda en el arreglo.
    while (!feof(ficApeNom)){ //continua leyendo hasta el final del archivo.
        i++; //incrementa para la siguiente posición del arreglo.
        fgets(vecAux[i], 20, ficApeNom); //continua leyendo y guardando los nombre en el
    } //arreglo "vecAux".
    fclose(ficApeNom);
    /*----- Ordenar el arreglo alfabéticamente. (Utiliza 2 ciclos for) -----*/
    for(int j=0; j<N; j++){
        for(i=0; i<N-1; i++){
            if(strcmp(vecAux[i+1], vecAux[i]) < 0){ // Si vecAux[i+1] es menor alfabéticamente
                strcpy(cadAux, vecAux[i]); // al contenido del vecAux[i]. Realiza
                strcpy(vecAux[i], vecAux[i+1]); // el cambio.
                strcpy(vecAux[i+1], cadAux);
            }
        }
    } //repite hasta que todo los nombres estén ordenado.
    /*----- Ordenado el vector, transferir los nombres al archivo. -----*/
    ficApeNom = fopen("Nombres.txt", "w"); // Abre con "w" BORRANDO su contenido.
    for(i=0; i<N; i++){ // y guardar los nombres ordenados.
        fprintf(ficApeNom, "%s", vecAux[i]);
    }
    fclose(ficApeNom);
    printf("\n\nFin del Ordenamiento Alfabético de menor a mayor.....\n");
    system("PAUSE");
    system("CLS");
} //Fin de la función ordenarArchivo().

/*-----*/
void ordenarPorLargoDeCadena(int N, FILE *ficApeNom) {
    typedef char nombPers[20];
    nombPers vecAux[N]; //arreglo Auxiliar que contendra los nombre.
    nombPers cadAux; //cadena auxiliar para realizar el ordenamiento del arreglo.
    bool bandOrdenado=false; //Bandera logica.

```

```

int i;
ficApeNom = fopen("Nombres.txt", "r"); //Abre de lectura.
/*----- Pasar los nombres que tiene el archivo al arreglo -----*/
i=0; //Variable indica la posición en el arreglo.
fgets(vecAux[i], 20, ficApeNom); //Lee el primer nombre y lo guarda en el arreglo.
while (!feof(ficApeNom)){ //continua leyendo hasta el final del archivo.
    i++; //incrementa para Guardar en la siguiente posición del arreglo.
    fgets(vecAux[i], 20, ficApeNom); //continua leyendo y guardando los nombre en el arreglo "vecAux".
}
fclose(ficApeNom); //Cierra el archivo.
system("CLS");
printf("Se procede a ordenar por su largo, los diferentes nombres.....\n");
for(int j=0; j<N; j++){
    for(i=0; i<N-1; i++){
        if(strlen(vecAux[i]) > strlen(vecAux[i+1])){ // Si el nombre del vecAux[i] es mas
            strcpy(cadAux, vecAux[i]); // largo que el nombre del vecAux[i+1].
            strcpy(vecAux[i], vecAux[i+1]); // cambio los nombres.
            strcpy(vecAux[i+1], cadAux);
        }
    }
}
/*----- Mostrar el arreglo Ordenado por el largo de sus cadena -----*/
system("CLS");
printf("\n Se termino de ordenar los datos Por el largo que tiene cada nombre.....");
printf("\n\n Se procede al listado de los datos.....\n\n");
for(i=0; i<N; i++){
    printf("%d %s \n", strlen(vecAux[i])-1, vecAux[i]); //strlen(vecAux[i])- 1: Se le resta 1 para no contar el fin de cadena (\0).
}
printf("Final del Listado.....\n");
system("PAUSE");
system("CLS");
} //Fin de la función ordenarLargoCadena().





```

DATOS DEL GRUPO DE ALUMNOS		
División: 1K.....	Profesor:	Fecha de Entrega __/__/__
	Auxiliar o JTP:	
Legajo o DNI	Apellido y Nombre	

Problemas Propuestos

Dados los siguientes enunciados con manejo de registro, encuentre una solución para cada uno de los problemas propuestos.

- 1)  Se dispone del archivo binario ("*Historia_2A.dat*"), de un colegio secundario, con los siguientes datos en cada registro:

-  Nombre del Alumno.
-  Promedio del primer trimestre en una materia.
-  Promedio del segundo trimestre en una materia.
-  Promedio del tercer trimestre en una materia.


Conociendo el Promedio General (PG) se puede definir que si:

- PG** \geq 7 El alumno se exime.
- PG** \geq 4 El alumno rinde en diciembre.
- PG** $<$ 4 El alumno rinde en marzo.

SE PIDE:


- a) Defina la estructura del registro.
- b) Grabe en el archivo, un número **no determinado** de alumnos. Analice y defina como será el fin de la carga.
- c) Realice un listado por pantalla del archivo, mostrando: Nombre del alumno, Promedio General.
- d) Terminado el listado, muestre cuantos alumnos están **eximidos**, cuantos rinden en diciembre y cuantos rinden en marzo. Determine y muestre el promedio de alumnos Eximidos.

EJEMPLO DE ARCHIVOS BINARIOS DE REGISTRO

- 2)  Una empresa dedicada a la venta de vehículos nuevos, tiene en un archivo registrado los datos correspondientes a los vehículos para la venta. El registro tiene 3 campos:
- (1) Categoría o clasificación (**A**: alta gama, **B**: Baja gama, **M**: media gama)
 - (2) Nombre Vehículo.
 - (3) Precio del vehículo.

SE PIDE:

- a) Utilizando una función Sin Tipo, Realizar la carga de los registros al archivo (Ingresar 3 vehículos por Clasificación o categoría como mínimo).
- b) Usando una función Con Tipo lea el archivo y Determinar cuántos vehículos se tiene en cada categoría. Mostrar en la función main el resultado.
- c) Usando una función Sin Tipo, Ingresado un tipo de categoría, listar el archivo en pantalla los nombres y precio de vehículos disponibles en la categoría ingresada.

- 3)  Un hotel tiene en un archivo "Estadistica.dat", los datos que representan la cantidad total de turistas registrados en su hotel en los años 2010, 2011, 2012, 2013, 2014, 2015. La estructura de los datos es la siguiente: (*) Año, (*) Cantidad de turistas.


SE PIDE:

- a) Usando una función Sin Tipo, Ingresar los datos (registros) en el archivo.
- b) Usando una función Con tipo, lea el archivo y determinando cuál fue la menor cantidad de turistas ingresados y en qué año se produjo.

- c) Si el próximo año se espera un ingreso calculado en 50 turistas más que el último año y el hotel dispone de 137 camas. Determinar leyendo el archivo.

✚ ¿Si habrá camas libres o quedaran turistas sin alojar? (Indicar cantidad en cada caso).


EJEMPLO DE ARCHIVOS BINARIOS DE ENTEROS

- 4)  En una escuela se tiene un archivo binario, las edades correspondientes a sus alumnos. a partir del archivo creado.


SE PIDE:

- a) Ingresar N valores enteros correspondientes a las edades de los alumnos.
- b) Encontrar el promedio de alumnos que están entre los [12..15] y entre [8..11] años de edades.



EJEMPLO DE ARCHIVOS BINARIOS DE REALES

- 5)  Se guardaran N números reales positivo de tres dígitos en su parte entera como máximo en un archivo binario. En caso de ingresar más de 3 dígitos mostrar un mensaje de error en la carga y no guardar ese valor.

SE PIDE:

- a) Haciendo uso de una función Sin Tipo, realizar la carga de los datos, como mínimo 10 valores.
 - b) Haciendo uso de una función Con Tipo, recorrer el archivo determinando cual fue el valor más pequeño y el más grande. El valor mayor será devuelto por la función y el valor menor se utilizara parámetro por referencia.
 - c) También hacer el listado en pantalla de todos los valores ingresados al archivo.
- 6)  En un archivo binario se almacenan valores reales correspondientes a una muestra estadística.
- #### SE PIDE:
- a) Ingresar N valores al archivo binario.
 - b) Usando una función Con Tipo, calcular el promedio de la muestra y mostrar dicho valor en main.
 - c) Determinar cuántos valores se tiene del promedio hacia abajo y del promedio inclusive hacia arriba.

EJEMPLO DE ARCHIVOS TEXTO

- 7)  Ingresar N cadenas de caracteres y grabarlas en un archivo de texto. Determine y muestre a través de una función con tipo la cantidad de palabras grabadas en el archivo.
- 8)  Guardar en un archivo de texto los nombres de 20 personas en mayúsculas y luego mostrar por pantallas dichos nombres en minúsculas.

Ejercicios Adicionales (Ejercicio 09)

Este ejercicio solo debe ser realizado, por los alumnos que no presentaron el TP en la fecha estipulada. El ejercicio deberá ser presentado junto al resto de los ejercicios resuelto.

- 9) Una empresa tiene N vendedores y sus datos se encuentran guardados en un archivo binario con la siguiente estructura de registro:
- ✚ Código del vendedor. Apellido y Nombre. Sueldo base y Monto de la Comisión a recibir.

SE PIDE:

- a) Registrar los datos de N vendedores.
- b) Muestre el apellido y nombre de los vendedores y el sueldo a percibir (Sueldo Base + Comisión).
- c) Ingrese un código de vendedor y modifique el monto de la comisión del mismo.

Ejercicio Complementario

Este ejercicio es para que el alumno practique y amplíe su conocimiento y destreza.

Diseñe un que permita manejar una agenda personal de clientes y amigos, La agenda deberá contar con un sistema de menú para seleccionar las acciones a realizar.