Zapata Santiago. TUP comisión 1, 1er año.

Trabajo de investigación

Laboratorio de computación

Ejercicio N°1: Definir los siguientes conceptos

- **Archivo**: Un archivo es un conjunto de datos en una colección de entidades denominadas registros, que son de igual tipo y constan a subes de diferentes entidades de nivel más bajo denominado campos.
- **Mezcla de Archivos**: La "mezcla de archivo" se refiere a la combinación de varios archivos en un solo archivo o documento. Puede involucrar la fusión de archivos de diferentes formatos, como documentos de texto, hojas de cálculo, imágenes o archivos de audio, en un solo archivo que contenga todos los elementos combinados.
- **Apareo de Archivos**: Consiste en combinar o fusionar dos o más archivos de datos en uno solo, seleccionando registros que cumplen con ciertas condiciones o criterios predefinidos.
- **Corte de Control**: Es un algoritmo que permite analizar información, generalmente provista mediante registros, agrupándolos según diversos

criterios. Como precondición se incluye que la información debe estar ordenada según los mismos criterios por los que se la quiera agrupar.

Ejercicio N°2: Describir las siguientes operaciones

1. Mezcla de Archivos: Es una operación que combina dos o más archivos ordenados en uno solo. Los archivos de entrada deben estar ordenados por la clave de búsqueda y así, el resultado también lo estará.
2. Apareo de Archivos: Es una operación que combina dos o más archivos ordenados en uno solo, utilizando técnicas para buscar y comparar los registros en cada archivo y combinarlos en uno solo.
3. Corte de Control: Es una técnica utilizada para verificar la precisión y validez de los datos almacenados en un archivo. Se selecciona un registro aleatorio del archivo y se verifica si cumple con ciertos criterios predefinidos. Si el registro no cumple con los criterios, se realiza una investigación para determinar la causa del problema y corregirlo si es necesario.

Ejercicio N°3:

- ¿Qué diferencia hay entre un arreglo y un archivo?

Un arreglo es una estructura de datos que solo almacena su valor en memoria mientras se ejecuta el programa, mientras que un archivo puede ser guardado en un medio de almacenamiento secundario y es persistente en el tiempo.

- ¿Qué diferencia hay entre mezcla y apareo?

La mezcla se basa en un campo común para combinar registros, mientras que el apareo se basa en una relación específica entre los registros, definida por claves o condiciones.

- ¿Cómo se clasifican los archivos según el método de acceso?

Los archivos se clasifican según el método de acceso en:

- Ficheros de Acceso Secuencial: En donde se acceden a los elementos del archivo en el mismo orden que fueron situados. Es decir, si queremos llegar a un registro, debemos recorrer todos los anteriores.
- Ficheros de Acceso Directo: Es posible acceder directamente a un determinado registro si le especificamos un índice, el cual determina la posición del registro respecto al origen del archivo.

- ¿Cómo se clasifican los archivos según el tipo de contenido?

Los tipos de archivo se pueden clasificar según el tipo de contenido en archivos de texto, o a archivos binarios. Pero hay que tener en cuenta que la única forma de guardar datos en el disco es en el formato binario.

- ¿Qué diferencia hay entre un archivo de texto y un archivo binario?

Un archivo de texto es una secuencia de caracteres que se guarda en binario pero se interpreta como texto, mientras que un archivo binario puede contener datos con diferentes formatos, como caracteres mezclados con enteros y flotantes. Además, los archivos de texto son planos y todas las letras tienen el mismo formato, mientras que los archivos binarios pueden tener una estructura más compleja y requieren el uso de estructuras para trabajar con ellos.

- ¿Por qué no se puede abrir un archivo?

Algunas razones por las que no se puede abrir un archivo son:

- No existe el mismo
- El disco se encuentra lleno
- El nombre es incorrecto
- El directorio al que se intenta acceder no es válido.

- ¿Por qué es necesario cerrar los archivos al final del proceso?

Cerrar un archivo garantiza que todos los datos que se han escrito en el archivo se guarden correctamente en el disco. Si no se cierra, es posible que algunos datos no se guarden y se pierdan. Además, el no cerrar un archivo puede ocasionar problemas con otros programas o procesos que intenten acceder al mismo.

Indique en C las instrucciones para realizar las siguientes operaciones

1. ¿Qué significa este tipo de apertura?

➢ r: Abre un archivo de texto como lectura.
➢ w: Crea un archivo de texto para escritura.
➢ a: Abre un archivo de texto para añadir.
➢ r+: Abre un archivo de texto para lectura / escritura.
➢ w+: Crea un archivo de texto para lectura / escritura.
➢ a+: Añade o crea un archivo de texto para lectura / escritura.
➢ rb: Abre un archivo binario para lectura.
➢ wb : Crea un archivo binario para escritura.
➢ ab: Abre una archivo binario para añadir.
➢ r+b: Abre un archivo binario para lectura/escritura.
➢ w+b: Crea un archivo binario para lectura/escritura.

➢ a+b: Añade o crea un archivo binario para lectura/escritura.

2. ¿Qué significa lo siguiente?
    - t: modo texto. Normalmente es el modo por defecto. Se suele omitir: es el modo por defecto y se utiliza para leer o escribir datos de texto. En este, los caracteres especiales como saltos de línea y tabulaciones se manejan de manera especial para que sean legibles por humanos.
    - b: modo binario: se utiliza para leer o escribir datos binarios, como imágenes, audio o video. En este modo, los datos se manejan como una secuencia de bytes sin procesar y no se realizan conversiones especiales.

3. Indicar para que se usan las siguientes funciones

Función	Significado
fopen()	Abre un archivo
fclose()	Cierra un archivo
fgets()	Lee una cadena de un archivo
fputs()	Escribe una cadena en un archivo
fseek()	Busca un byte específico en un archivo
fprintf()	Escribe una salida con formato en el archivo
fscanf()	Lee una entrada con formato desde el archivo
feof()	Devuelve true si se llega al final del archivo
ferror()	Devuelve true si se produce un error
rewind()	Coloca el localizador de posición del archivo al principio del mismo

remove()	Borra un archivo
fflush()	Vacía un archivo

4. ¿Qué significa “FILE *nombrearchivo”?

   FILE *nombrearchivo define una variable llamada "nombrearchivo" que puede almacenar y escribir datos en la dirección de memoria de un archivo.

5. ¿Qué significa el siguiente trozo de programa?¿Qué tipo de archivo se está abriendo?

```
FILE *fich;
    if ((fich = fopen("nomfich.dat", "r")) == NULL) printf ( " Error en
la apertura. Es posible que el fichero no exista \n");
```

   Esta parte del programa abre un archivo llamado "nomfich.dat" en modo lectura y comprueba si la operación fue exitosa. Si el archivo no existe o no se puede abrir, se muestra un mensaje de error. El tipo de archivo que se está intentando abrir es un archivo de texto, ya que se está usando “r” en el modo de lectura.

6. ¿Qué tipo de apertura tienen los siguientes archivos?

```
    FILE * datos;
    datos = fopen ("nombres.dat","r");
    datos = fopen ("nombres.dat", "w");
    datos = fopen ("nombres.dat", "a");
    datos = fopen ("nombres.dat", "ra");
```

- El primero con “r” , tiene apertura de tipo lectura.
- El segundo con “w” tiene apertura de tipo escritura.
- El tercero con “a” tiene apertura de tipo añadir.
- El cuarto tiene una apertura inválida.

7. ¿Qué representa el siguiente trozo de programa?

```
FILE *parch;
    if ((parch = fopen("c:\\banco.dat", "rb")) == NULL) // Se abre en
modo lectura
        printf("\nEl archivo no puede ser abierto");
    if ((fclose(parch)) == -1) // Se cierra el archivo
        printf("\nNo se pudo cerrar el archivo");
    else
        printf("\nEl archivo se cerro exitosamente");
```

El programa comienza declarando un puntero a FILE para almacenar la referencia al archivo abierto. Después intenta abrir el archivo banco.dat en modo de lectura binaria, y si este no puede ser abierto, devuelve el mensaje de que no puede ser abierto.

Luego, intenta cerrar el archivo, y se verifica si este pudo ser cerrado correctamente, en caso de que no, devuelve el mensaje de que no se pudo cerrar, y en caso de que sí, devuelve el mensaje de que se cerró exitosamente.

Ejercicio N° 4: Resolver los siguientes ejercicios

1. Diseñar un programa que permita generar un archivo de texto que permita cargar cinco nombres separados por punto. Mostrar el contenido del archivo un nombre abajo del otro.

```
#include <stdio.h>
#include <string.h>
int main()
{
    FILE *archivo;
    char nombres[255]; /*Creamos una cadena de caracteres*/
    char nombresArreglo[5][15] = {{""}, {""}, {""}, {""}, {""}}; /*Creamos
un arreglo para almacenar los nombres de la cadena*/
    int nombreActual = 0, j = 0; /*Inicializamos en 0 dos variables que
nos ayudara a iterar*/
    archivo = fopen("archivo.txt", "w+"); /*Abrimos o creamos en caso  de
```

```
que no exista, el archivo archivo.txt en formato de escritura*/
    if (archivo == NULL){ /*Sentencia if por si existen errores*/
        printf("\n No se pudo crear o abrir el archivo.");
    }
    else{
        printf("\n Ingrese cinco nombres separados por un punto: "); /*Se
solicita al usuario ingresar 5 nombres separados por un punto*/
        scanf("%s", nombres); /*Se guarda los nombres en la cadena
nombres*/
```

2. Dado el siguiente programa, se pide:

I. Correr el programa y corregir los errores

II. Agregar al programa , mostrar el contenido del archivo cargado previamente mediante el siguiente formato

```
include<stdio.h>
#include<conio.h>
#include<string.h>
#define a "c:\\arch.dat"
struct registro
{
int cliente;
char nombre[20];
float saldo;
};
int main()
{
FILE *arch;
struct registro reg;
char seguir;
if ((arch=fopen(a,"wb"))==NULL)
printf("No se pudo abrir el archivo");
do
{
printf("\nIngrese numero de cliente: ");
scanf("%d",&reg.cliente);
printf("\nIngrese el nombre: ");
scanf("%s",&reg.nombre);
printf("\nIngrese el saldo: ");
scanf("%f",&reg.saldo);
```

```
fwrite(&reg,sizeof(reg),1,arch);
printf("desea terminar s/n: ");
scanf("\n%c",&seguir);
}
while(seguir=='n');
fclose(arch);
getch();
}
```

Programa corregido:

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
#define a "arch.dat"
struct registro{
    int cliente;
    char nombre[20];
    float saldo;
};
int main(){
    FILE *arch;
    struct registro reg;
    char seguir;
    if ((arch = fopen(a, "wb")) == NULL)
        printf("No se pudo abrir el archivo");
    do{
        printf("\nIngrese numero de cliente: ");
        scanf("%d", &reg.cliente);
        printf("\nIngrese el nombre: ");
        scanf("%s", reg.nombre);
        printf("\nIngrese el saldo: ");
        scanf("%f", &reg.saldo);
        fwrite(&reg, sizeof(reg), 1, arch);
        printf("desea terminar s/n: ");
        scanf("\n%c", &seguir);
    } while (seguir == 'n');
```

```
    fclose(arch);

/*Abrimos el  archivo para leerlo*/
     if ((arch = fopen("arch.dat", "rb")) == NULL) {
        printf("No se pudo abrir el archivo");
        return 1;
    }else{

       while (fread(&reg, sizeof(reg), 1, arch) == 1) { /*Utilizamos
fread y mostramos los valores del archivo*/
        printf("Cliente: %d\n", reg.cliente);
        printf("Nombre: %s\n", reg.nombre);
        printf("Saldo: %.2f\n\n", reg.saldo);
    }

    }
    fclose(arch);
    getch();
    return 0;

}
```

Ejercicio Nº 5: A partir de un archivo binario, generar el siguiente informe:

Listado de Alumnos

DNI	Apellido	Nombre	Nota

Total de Alumnos:

Alumno con la nota más alta:

Alumno con la nota más baja:

Promedio general de los alumnos:

Cantidad de alumnos con nota mayor e igual a 6

Cantidad de alumnos con nota menor a 6

Porcentaje de alumnos con nota mayor e igual a 6

Promedio de alumnos con nota menor a 6

```
#include <stdio.h>
#include <string.h>
struct alumnos{ /*Creamos la estructura alumnos con DNI, apellido, nombre
y nota*/
    int DNI;
    char apellido[20];
    char nombre[20];
    float nota;
};
 int main(){

    FILE *archivo;
     /*Instanciamos la estructura alumnos como al, notaAlta y notaBaja*/
    struct alumnos al;
    struct alumnos notaAlta;
    struct alumnos notaBaja;
    /*Inicializamos varios contadores en 0*/
    int cantidadAlumnos  = 0, notaMayor6 = 0, notaMenor6 = 0, sumaNotas=0,
notasMenoresA6 = 0;
    if((archivo = fopen("alumno.dat","rb"))==NULL){ /*Abrimos el archivo
alumno.dat como lectura binaria y verificamo si no hay errores*/
        printf("No se pudo abrir el archivo");
    }else{
        notaAlta.nota = 0; /*Iniciamos el valor nota de notaAlta en 0*/
        notaBaja.nota = 11; /*Iniciamos el valor nota de notaAlta en 11
(El valor maximo)*/
        printf("\t LISTA DE ALUMNOS \n");
        printf("DNI \t  Apellido \t Nombre \t Nota");
        while(fread(&al,sizeof(al),1,archivo) == 1){  /*Mientras que se
```

```
lea el  archivo*/
          /*Mostramos la tabla con todos los datos*/
            printf("\n%i ", al.DNI);
            printf("   %s ", al.apellido);
            printf("      %s ", al.nombre);
            printf("         %.2f ", al.nota);
            cantidadAlumnos++; /*Aumentamos la cantidad de alumnos en 1*/
            sumaNotas+=al.nota; /*Sumamos las notas*/
            if(notaAlta.nota<al.nota){ /*Si la nota actual es mas alta que
la nota guardada como la  mas alta, entonces remplazamos  sus valores*/
                notaAlta.DNI = al.DNI;
                strcpy(notaAlta.apellido,al.apellido);
                strcpy(notaAlta.nombre,al.nombre);
                notaAlta.nota = al.nota;
            }
            if(notaBaja.nota>al.nota){ /*Si la nota actual es mas baja que
la nota guardada como la  mas baja, entonces remplazamos  sus valores*/
                notaBaja.DNI = al.DNI;
                strcpy(notaBaja.apellido,al.apellido);
                strcpy(notaBaja.nombre,al.nombre);
                notaBaja.nota = al.nota;
            }
            if(al.nota>=6){ /*Si la nota actual es mayor a 6, aumentamos
el contador en 1*/
                notaMayor6++;
            }else{ /*Sino, sumamos la nota actual y el contador de notas
menores*/
                notasMenoresA6+=al.nota;
                notaMenor6++;
            }

        }
    /* Limpiamos y cerramos el archivo */
        fflush(archivo);
        fclose(archivo);
        /*Mostramos todo lo que se pide en la consigna*/
        printf("\nLa cantidad de alumnos es %i ", cantidadAlumnos);
```

```
/*Mostramos la cantidad de alumnos*/
        printf("\nEl alumno con la nota mas alta fue %s %s ",
notaAlta.nombre, notaAlta.apellido); /*Mostramos a quien pertenece la nota
mas  alta, llamando a la estructura en donde la guardamos*/
        printf("\nEl alumno con la nota mas baja fue %s %s ",
notaBaja.nombre, notaBaja.apellido); /*Mostramos a quien pertenece la nota
mas  baja, llamando a la estructura en donde la guardamos*/
        printf("\nEl promedio general es: %i", (float)sumaNotas /
(float)cantidadAlumnos); /*Sacamos el promedio general dividiendo la suma
de las notas por la cantidad de alumnos*/
        printf("\nLa cantidad de alumnos con nota mayor o igual a 6 es %i
", notaMayor6);    /*Mostramos la cantidad de alumnos con nota mayor a 6
*/
        printf("\nLa cantidad de alumnos con nota menor a 6 es %i ",
notaMenor6);  /*Mostramos la cantidad de alumnos con nota menor a 6*/
        printf("\nEl porcentaje de alumnos con nota mayor a 6 es %.2f %%",
((float)notaMayor6/(float)cantidadAlumnos)  *100); /*Mostramo el
porcentaje que ocupa los alumnos con nota mayor a 6 con  respecto al
total*/
        printf("\nEl promedio de alumnos  con nota menor a 6 es de %.2f ",
(float)notasMenoresA6 / (float)notaMenor6); /*Mostramos el promedio de los
que sacaron notas menores a 6*/
    }
    return 0;
}
```