Universidad De Guadalajara

Centro Universitario De Ciencias Exactas E Ingenierías

División De Electrónica Y Computación

Departamento De Ciencias Computacionales

Ingeniería Informática

Seminario de Estructuras de Datos II

Profesor: Luis Felipe Mariscal

Sección: D19

Reporte practica #4

Nombres Del Equipo:

- López Martínez Ballardo Axel
- Ruiz Ortiz Valentin Alejandro
- Silva Ruiz Diego Adan
- Waldo Salazar Alejandro
- Lalo Eduardo Perez Meza

Contenido

PRACTICA 4:

MARCO TEÓRICO

- **Abrir un archivo**

La primera operación que generalmente se realiza en un objeto de una de estas clases es asociarlo a un archivo real. Este procedimiento es conocido como *abrir un archivo*. Un archivo abierto se representa dentro de un programa mediante una *secuencia* (es decir, un objeto de una de estas clases) y cualquier operación de entrada o salida realizada en este objeto de secuencia se aplicará al archivo físico asociado a eso.

Para abrir un archivo con un objeto de secuencia, usamos su función miembro open: Donde es una cadena que representa el nombre del archivo que se abrirá, y es un parámetro opcional con una combinación de las siguientes banderas:
open (filename, mode);

filenamemode: en este caso el modo de apertura fue ios::in | ios::out para esto se explica a continuación el método para poder realizar esto.

- **Fstream**

Este tipo de datos representa el flujo de archivos en general, y tiene las capacidades de ofstream y ifstream, lo que significa que puede crear archivos, escribir información en archivos y leer información de archivos.

```
fstream archDeportEntradaSalida;
archDeportEntradaSalida.open("deportista.dat",ios::in|ios::out);
```

- **Puntos de posición de archivo**

Tanto istream y ostream proporcionan funciones miembros para reposicionar el puntero de posición del fichero. Estas funciones de miembro son seekg ("seekget") para istream y seekp ("seekput") para ostream.

El argumento para buscar y buscar normalmente es un entero largo. Se puede especificar un segundo argumento para indicar la dirección de búsqueda. La

dirección de búsqueda puede ser ios::beg (el valor predeterminado) para el posicionamiento relativo al comienzo de un flujo, ios::cur para el posicionamiento relativo a la posición actual en un flujo o ios::end para el posicionamiento relativo al final de un flujo corriente.

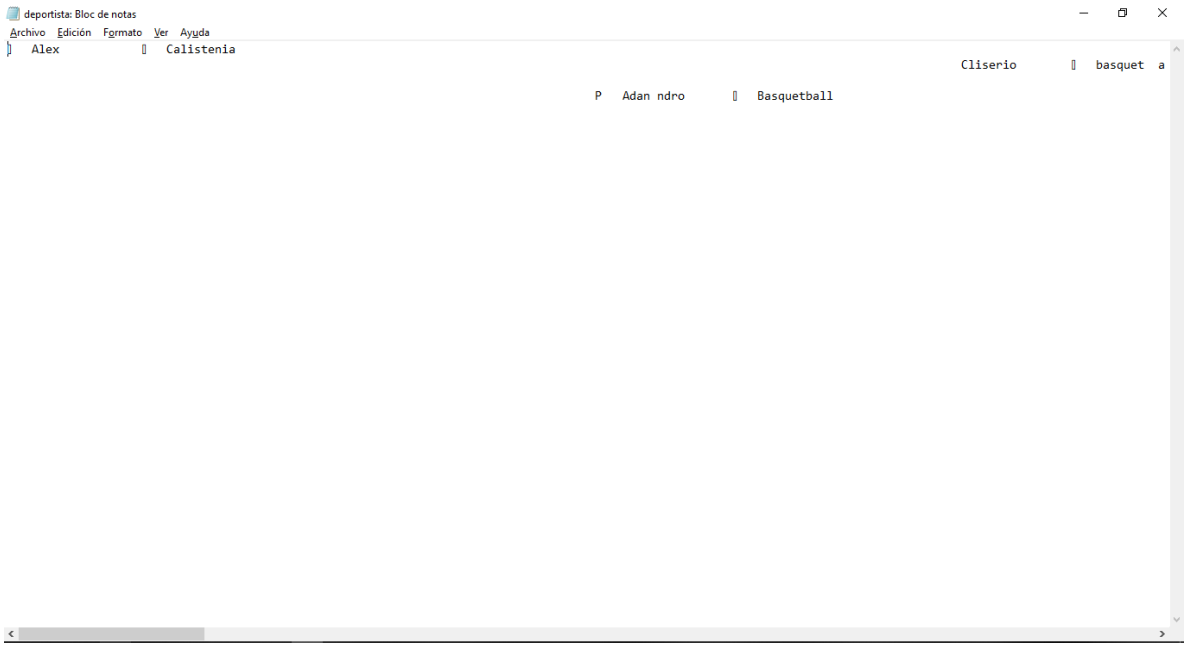
El puntero de posición del archivo es un valor entero que especifica la ubicación en el archivo como un número de bytes desde la ubicación de inicio del archivo.

```
archDeportEntradaSalida.seekp((d.obtenerNumeroDeSocio()-1) * sizeof (Deportista));
archDeportEntradaSalida.write(reinterpret_cast<const char*>(&d),sizeof (Deportista));
```

```
archDeportEntradaSalida.seekg(sizeof(Deportista)*(numeroSocioTemp-1));
archDeportEntradaSalida.read(reinterpret_cast<char*>(&d),sizeof (Deportista));
```

RESULTADOS

- **Archivo físico:**



En el siguiente archivo se muestra el resultado de guardar registros de deportistas en modo binario y a su vez en forma directa moviendo el puntero dentro del archivo físico, de esta manera se logra mandar a un registro directamente a la posición indicada por el usuario.

- **Menú**

```
Menu - Asociacion de Deportistas Tapatios
1. Alta
2. Baja
3. Cambios
4. Consultas Individuales (Numero de Socio)
5. Consuta General
6. Salir
Elige tu opcion:
```

En este menú se muestran las funciones a desarrollar y a continuación se mostrará el funcionamiento de cada una de ellas.

- **Alta**

```
Alta Deportista
Ingrese el numero de socio: 15
Teclea Nombre, Edad y Deporte
Alan 20 Natacion
```

```
ID    Nombre    Edad    Deporte

1     Alex        20    Calistenia

15    Alan        20    Natacion

30    Cliserio    20    basquet

50    Axel        20    Tenis

80    Adan        24    Basquetball

100   Alejandro   20    Natacion

Presione una tecla para continuar . . .
```

Antes de dar de alta al registro se pide un numero de socio que a la vez se va a utilizar como indicador para mandarlo en posición del archivo, una vez obtenido el numero de socio se pide los campos nombre, edad y deporte, entonces una vez con todos los campos llenos se mandan al archivo físico en el almacenamiento secundario.

- **Baja**

```
Elige tu opcion: 2
Ingrese el numero de socio a eliminar: 15
```

```
ID    Nombre    Edad    Deporte

1     Alex        20    Calistenia

30    Cliserio    20    basquet

50    Axel        20    Tenis

80    Adan        24    Basquetball

100   Alejandro   20    Natacion

Presione una tecla para continuar . . .
```

La baja tiene una lógica parecida al alta de deportista, primero se solicita el numero de socio, para saber si es un deportista existente en el archivo, ahorrando la búsqueda secuencial, con el numero de socio podemos dirigir el puntero al registro y verificar si hay algún registro, si se encuentra un registro se inserta un deportista en blanco.

- **Cambios**

```
Elige tu opcion: 3
Numero de Socio a Modificar: 15
Teclea Nombre, Edad y Deporte
Beto 25 Basquet
Modificado Exitosamente Deportista
```

```
ID    Nombre    Edad    Deporte

1     Alex        20    Calistenia

15    Beto        25    Basquet

30    Cliserio    20    basquet

50    Axel        20    Tenis

80    Adan        24    Basquetball

100   Alejandro   20    Natacion

Presione una tecla para continuar . . .
```

Para poder modificar un deportista se solicita el numero de socio, para verificar si hay un registro existente, si lo hay se pide los campos nuevamente que son nombre, edad y deporte y se sustituye el registro con la nueva información.

- **Consultas Individuales**

```
Ingresa el numero de socio a buscar: 15
ID    Nombre    Edad    Deporte

15    Alan         20   Natacion

Presione una tecla para continuar . . .
```

La consulta individual se busca el numero de socio que es la llave primaria, con esta se mueve el puntero a la posición indicada por el usuario, si hay un registro en esa dirección se imprime, si no encuentra nada, quiere decir que están en blanco y imprime que no se encontró registro.

- **Consulta General**

```
ID    Nombre    Edad    Deporte

1     Alex         20   Calistenia

15    Alan         20   Natacion

30    Cliserio     20   basquet

50    Axel         20   Tenis

80    Adan         24   Basquetball

100   Alejandro    20   Natacion

Presione una tecla para continuar . . .
```

En la consulta general, como no se utilizó índices, tuvimos que hacer un recorrido secuencial en busca de registros, si encuentra un ID entonces es un registro existente y lo imprime.

CONCLUSIONES

- **Waldo Salazar Alejandro**

  En realidad esta práctica me pareció bastante interesante porque pudimos reducir el tiempo de procesamiento para leer y escribir en un archivo físico en el almacenamiento secundario pero a la vez podemos generar demasiada fragmentación dependiendo del uso del archivo por los espacios en blanco que podemos dejar inutilizables y se vio la ventaja de utilizar el fstream.

- **Ruiz Ortiz Valentin Alejandro**

  Seekp y Seekg y sus respectivos Read Y Write son funciones de la libreria fstream que la verdad sus utilidade son increibles si las usas de maneras adecuadas la escritura en c++ puede sacar el partido mas grande, pues especificar la cantidad de bytes que vamos a leer o a escribir hace el trabajo mas facil y rapido.

- **López Martínez Ballardo Axel:**

  Esta practica me deja impresionado por la gran función que se implementa haciendo mas optima la lectura y escritura en el archivo y dejándome en mas que pensar al dejar los espacios en blanco los cuales podrían reducirse al tamaño de la cadena.

- **Diego Adán Silva Ruíz**

Concluyo con esta práctica que la librería fstream da bastantes alternativas para la apertura de archivos, de la mano con otras herramientas, resulta ser más sencillo de lo que parece guardar nuestros registros en archivos y extraer los datos fácilmente para recuperarlos dentro de nuestro programa, inclusive si el archivo tiene un encriptado ligero tipo dat, tipo bin, etc. Espero ver a futuro todo el potencial de las funciones de esta clase para manejar de manera más compleja los archivos.

- **Jesús Eduardo Pérez Meza**

La implementación de las funciones seekg y seekp optimiza aún más el proceso de búsqueda e inserción de información en un archivo gracias a la opción de poder manejar el puntero a nuestra conveniencia. Opino que estas nuevas funciones hacen que el programa vaya tomando más forma ya que la manera de manejar los datos se vuelve más ordenada y eficaz.

- **Referencias:**

Guardatti, S. d. (2010). *estructura de datos* (3ª ed.). México, México: Pearson.