UNIVERSIDAD DE GUADALAJARA

Red Universitaria e Institución Benemérita de Jalisco

Rodriguez Rentería Jesus Alejandro

Código:215510307

Materia: Seminario de solución de problemas de estructuras de datos II

Profesor: Gómez Anaya David Alejandro

Sección: D16

Sede: Centro Universitario de Ciencias Exactas e Ingenierías (CUCEI)

Fecha:06 de diciembre del 2020

1.Planteamiento del problema

Diseñar un programa capas de crear cuentas, modificar y eliminar. Debido a que existirá una cantidad grade de jugadores se requiere que las cuentas este en memoria secundaria y solo este un objeto de clase usuario en principal.

Se creará un archivo de índices para poder manejar el archivo de cuentas que almacene las direcciones de memoria y la llave primaria del objeto el usuario. La organización del índice es por orden alfabético del usuario

2.Objetivos

Crea un archivo de índices con las direcciones de memoria del fichero

Diseñar un algoritmo que pueda reconocer y buscar los objetos de manera óptima utilizando el fichero

Crear un algoritmo que no deje registrar usuarios repetidos.

3.Marco teórico

Para el desarrollo de esta actividad Se hace uso de almacenamiento en memoria secundaria, lista estática para los índices, lectura de ficheros

Lista estática: Aquella cuyo tiempo de vida corresponde al tiempo de ejecución del programa. Su inicialización será recordada y, aunque se abandone el bloque en que esta es visible, seguirá estando en memoria disponible para el ámbito en que fue declarada.

Lectura de ficheros: Es simplemente un flujo externo que se puede abrir para entrada (dando lugar a un flujo de archivo de entrada que, para simplificar, llamaremos simplemente archivo o fichero de entrada), para salida (dando lugar a un flujo de archivo de salida que, para simplificar, llamaremos simplemente archivo o fichero de salida). Apéndice 1

4.Desarrollo

El código cuenta con 5 clases diferente en las cuales son: Clase Menu, Clase User, clase StaticList, Clase Index, clase FileList.

Clase menú: esta clase es encargada de hacer las impresiones del menú principal y sub menús

Clase User: cuenta con cuatro atributos. Guarda el usuario, nombre, correo y password del jugador cuenta con los siguientes métodos:

constructores

- User();
- User(char nombre[20],char usuario[20],char correo[20],char password[20]);
- User(string nombre_,string usuario_,string correo_,string password_);

Se cuenta con tres constructores sin parámetros, con string y char. Debido a que los atributos de la clase son char Array tuve la necesidad de crear ambos constructores para que sin importar el datos de entrada pueda funcionar la instancia de la clase

- string toString(); Imprime los atributos de la clase
- string get_usuario() devuelve un string en el cual están los datos del usuario
- string get_password() devuelve un string en el cual están los datos del password
- void set_usuario(string aux) Ingresa un string para cambiar el usuario (se usan en la parte de modificar)
- void set_nombre(string) Ingresa un string para cambiar el nombre (se usan en la parte de modificar)
- void set_correo(string) Ingresa un string para cambiar el correo(se usan en la parte de modificar)
- void set_password(string) Ingresa un string para cambiar el password (se usan en la parte de modificar)

Clase StaticList cuenta con tres atributos un coutn que es un contador de los elementos de la lista, un apuntador a index(clase index) y n que es el largo de la lista estática. Lo métodos con los que cuenta son los siguientes:

- StaticList();
- StaticList(int N);
  Construcores con y sin parámetros. Se utiliza con parámetros para la creación de las lista estatica.
- void insertaordenado(Index d): obtiene un Índex, lo agrega a la lista en la última posición y por ultimo llama la función de orden(burbuja).
- int get_count() Obtiene el contador de elementos de la lista y lo regresa
- void guardar_indices(Index d) función se llama al iniciar el programa instancia los datos guardados en memoria secundaria a primaria

Clase Index cuenta con dos atributos un indice entero y la llave primaria tipo string los métodos con los que cuenta son

- Index();
- Index(int index,string primaryKey);

Constructor con y sin parámetros. El principal uso es el de parámetro ya que se utiliza cuando se agrega un nuevo elemento de la clase User

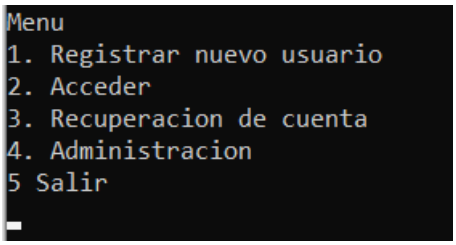
- string toString(); Función que regresa un string para imprimir los elementos de la clase
- int getIndex() optenemos el índice de la clase
- string get_primarykey() obtenemos la llave primaria de la clase
- bool operator > (Index right) Operador útil para hacer comparaciones (si la llave primaria es mayos a la llave primaria siguiente)
- bool operator == (Index right) Compara si la llave primaria es igual a la siguiente llave primaria del arreglo

- string guardar(); regresa un compimido de los datos de la clase par poder ser guardados por otro metodo
- void set_primaryKey(string aux) ingresa una llave primaria

Clase FileList el gestor de los archivos del programa cuenta con dos atributos una StaticList llamda inexList y un string filename que es donde se gurad el nombre del ficheto binario.

- FileList(string filename);Construcor de la clase encargado de inicalizar el nombre del filename
- void add(User obj): obtiene un objeto de tipo User, abre un fluo de entrada de daros a un arhcivo binario y almacena el objeto. Ademas crea un índice en la lista estatica que hace referencia al mismo objeto.
- void showallindex() Imprimeun index
- void showallorder() Imprime todos los objetos de tipo usuario que se encuentran en el archivo binario
- int showIndexList(); imprime la lista de indices y sus llaves primarias
- bool validacion_user(char[20]); valida que el usuario nuevo no sea repetido y regreso un valor booleano
- void guardar_indices() Abre un archivo de texto donde almacena los indices
- void carga_indices() obtiene una línea y separa los atributos para poder instanciar un indice de la clase Index
- bool inicio_seccion(string,string); Valida el inicio de sección del usuario y password correctos
- void mostrar_uno(string); Muestra un solo objeto de la clase User
- string modificar(string,int,string);pide el atributo a modificar ya busca el en fichero para poder hacer el cambio.
- void borrar_arch(); borra el archivo de indices cuando ya esta todo cargado en memoria principal
- void block_temporal(string); bloqueo de cuentas de manera temporal
- void block_perma(string); eliminación de cuentad el archivo binario
- void recupera_u(string); recupera las cuentas bloqueadas de manera temporal

5.Pruebas y resultados.



Inicio del programa menú principal.

Crear un objeto de tipo usuario

```
Ingresa usuario
jesus
Ingresa su nombre
jesus
Ingrese correo
jesus@jesus
Ingrese password
1
```

parte cuatro mostrar todos por índices

```
Administracion
1. Mostrar todos por indice (llave primaria y direccion en archivo).
2. Mostrar todos en archivo
3. Generaci¾n automßtica de cuentas
1
Index   160  Llave primaria  aron
Index   0  Llave primaria  jesus
Index   80  Llave primaria  jose
Presione una tecla para continuar . . .
```

Parte cuatro mostrar todos en un archivo

```
Administracion
1. Mostrar todos por indice (llave primaria y direccion en archivo).
2. Mostrar todos en archivo
3. Generaci¾n automßtica de cuentas
2
Nombre: aron, Usuario: aron, Correo: aron@aron, Password: 1

Nombre: jesus, Usuario: jesus, Correo: jesus@jesus, Password: 1

Nombre: jose@jose, Usuario: jose, Correo: jose@jose, Password: 1

Presione una tecla para continuar . . .
```

Inicio de sección exitosos mostrando la opción 1

```
Acceder a una cuenta
1. Mostrar
2. Modificar
3. Eliminar
4. salir
1
Nombre: jesus, Usuario: jesus, Correo: jesus@jesus, Password: 1
Presione una tecla para continuar . . .
```

Modificando un usuario

```
Modificar
Nombre: jesus, Usuario: jesus, Correo: jesus@jesus, Password: 1
Que desea modificar
1. Usuario
2. nombre
3. correo
4. Password
5.salir
1
Ingrese la palabra de remplazo
mario
```

6.Conclusion.

La creación de este programa hace que las cosas este diseñadas de manera que al momento de ejecutarse consuma poca memoria RAM independiente mente de los usuarios registrados en la aplicación, se forzar a que la computadora sacrifique algo de tiempo de búsqueda en memoria secundaria para ahorrar el almacén en la memoria principal. El problema de las consultas a disco se trata de resolver con un listado de índices mucho mas liviano que la clase usuario que hace referencias a lugares de memoria donde se almacenan están almacenados los datos de la clase User. Este diseño de programa busca mantener un equilibrio entre tiempos de búsqueda y poco uso de memoria principal para que la ejecución se muy buena.

7.Apendice.

Las funciones para el de archivos en c++ se encuentra en la biblioteca fstream. Para este programa en específico utilizamos el ofstream y ifstream recuerda que la ejecución de archivo en memoria secundaria es costoso, pero se ahorra RAM