Universidad Rafael Landívar

Ingeniería en informática y sistemas

Ing. David Luna

Proyecto Segunda Parte

Kevin Barrientos

1084517

Francisco Alonzo

1197517

Guatemala, Miércoles 24 de noviembre, 2018

**INTRODUCCIÓN**

En la segunda entrega del pryecto, CORREO, consistió en implementar un archivo de Acceso indexado o acceso indizado (la traducción correcta) es un modo de organización de archivos en el cual al archivo le acompaña un índice que tiene la función de permitir el acceso directo a los registros del disco. El índice se puede organizar de diversas formas, las más típicas son: secuencial, multinivel y árbol. A través del índice podremos procesar un archivo de forma secuencial o de forma directa según la clave de indización, y esto independientemente de cómo esté organizado el archivo por sí mismo. El lenguaje de programación utilizado fue JAVA.

Los objetivos principales del proyecto consistieron en aplicar los conocimientos adquiridos en la clase teórica y laboratorio del curso Manejo e implementación de archivos, implementando estructuras de datos genéricas, a nivel de archivos, pilas y listas, para la solución del problema planteado e identificar cómo el uso de la programación es indispensable para la optimización de procesos.

El presente trabajo contiene la documentación externa del proyecto, poniendo en práctica la primera y segunda fase del método de resolución de problemas: (1) el análisis y (2) el diseño; los cuales fueron útiles para posteriormente codificar y verificar el correcto funcionamiento del programa.

**Análisis**

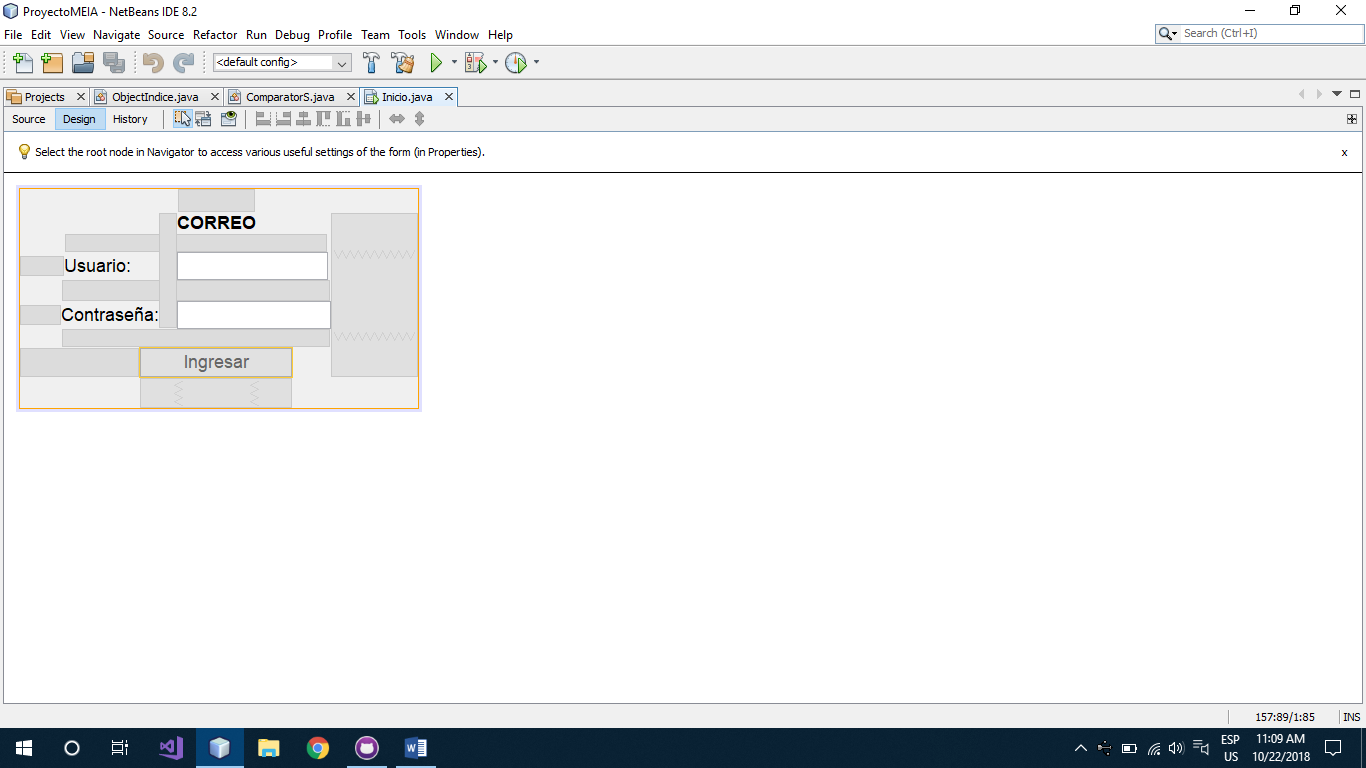
|  |  |
| --- | --- |
| **Alcances** | **Funcionalidades requeridas** |
| * Permite el acceso secuencial. * Permite el acceso directo a los registros. * Se pueden actualizar los registros en el mismo fichero, sin necesidad de crear un fichero nuevo de copia en el proceso de actualización. | * Lista de usuarios * Desc\_indice, Desc\_lista * Indice,lista |
| **Procesos** | **Salidas** |
| * Solo se puede utilizar soportes direccionales. * Tiene tendencia a que aumente el tiempo medio de acceso a los registros, cuando se producen muchas altas nuevas con claves que hay que intercalar entre las existentes, ya que aumenta el área de overflow. * Ocupa más espacio en el disco que los ficheros secuénciales, debido al uso del área de índices. | * Información(listas de usuarios) |

**Diseño**

**Identificación de clases:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | Secuencial Indexado | | **Atributos**   * private RandomAccessFile archivo; * private File DescriptorIndice; * private File Indice; * private File DescriptorMasterFile; * private File masterFile; * private int nPosicion = 1; | | **Métodos**   1. InsertarIndice() 2. InsertarLista() 3. Next() 4. Siguente() 5. Modificar() 6. Eliminar() 7. Busqueda() 8. CrearDescriptorLista() 9. CrearDescriptorIndice() 10. ObtenerInicio() 11. NuevoInicio() 12. EsInicio() 13. ObtenerDSiguiente() 14. RefactorI() 15. RefactorF() 16. PosicionRegistro() 17. CantidadRegistrosLista() 18. CantidadRegistrosInactivos() 19. CantidadRegistrosActivos() |  |  | | --- | | ObjectIndice | | **Atributos**   * private String nregistro,posicion,Nlista,Nusuario, * Usuarioasociado,siguiente,estatus; | | **Métodos**   1. Getters y Setters | | |  | | --- | | ObjectLista | | **Atributos**   * private String Nombre\_lista,Usuario,Descripcion; * private int Numero\_usuarios; * private String Fecha\_Creacion; * private boolean Estatus; * private String rightpad(String text, int length) { * return String.format("%-" + length + "." + length + "s", text); | | **Métodos**   1. Getters y Setters |  |  | | --- | | ComparatorS | | **Atributos**   * NodoLista<T> inicio * NodoLista<T> fin | | **Métodos**   1. ListaEstaVacia() 2. Cantidadelementos() 3. Insertar(T value) 4. Insertar(T value, int index) 5. Extraer(int index) 6. ObtenerArreglo() | |

**Diseño de pantalla:**



**Conclusiones**

* Se concluye que por medio de un software que permite realizar distintas acciones al usuario, se pude comprobar en este caso, la lógica computacional.
* Todo sistema es susceptible a generar errores en caso que el usuario se confunda al momento de realizar distintas acciones.
* Debido a que todos los usuarios que deseen jugar se deberán acomodar a las reglas planteadas de manera lógica en el juego.

**Bibliografía**

* Xerox. (2017). Pilas, listas y colas. 2017, de Xerox Sitio web: https://www.xerox.com/es-gt/servicios/managed-print/conocimientos/guia-de-analisis-documental?CMP=PPC-HUPEnterprice&DATE=08-06-2017&HBX\_PK=seguridad\_informatica&HBX\_OU=50&ADGRP=interest&AUD=GTM
* Ijnet. (2015). Verificadores de contenido. 2017, de Ijnet Sitio web: https://ijnet.org/es/blog/cinco-verificadores-de-contenido-gratuitos