INSTITUTO TECNOLÓGICO DE COSTA RICA

ESTRUCTURAS DE DATOS

Unidad de computación

PROYECTO DE GESTIÓN DE EXÁMENES CON ESTRUCTURAS DE DATOS

Jafeth Steven Vásquez Herrera y Kembly Quirós Araya

Sede San Carlos
Octubre 2016

0.1. Introducción

Este proyecto se ha realizado con el objetivo de conocer las diferentes estructuras en cuanto al tipo de información y su manejo, para esto ha sido necesario el estudio de diferentes estructuras de datos como las listas Simples, listas doblemente enlazadas y listas circulares.

Conocer las diferentes estructuras de datos y utilizarlas de manera adecuada de acuerdo con la necesidad o el problema que el programador deba resolver para lograr un programa eficiente y optimizado. Uno de los primeros pasos en este aspecto es conocer primero cual es la información que se necesita almacenar, luego es necesario saber como se va a almacenar y finalmente como implementar correctamente en el código la infraestructura en el código. Durante el siguiente documento se van a desarrollar algunos aspectos importantes necesarios para la ejecución del proyecto, se hablara sobre las dificultades encontradas y sobre las soluciones implementadas.

0.2. Análisis del problema

Para este proyecto se ha solicitado la creación de un sistema de administración de exámenes utilizando el lenguaje C++ así como TDA y listas enlazadas [1]. Para la realización del proyecto se debe trabajar en equipo, por lo cual es necesario trabajar en la encapsularon del código para garantizar la compatibilidad con el código del compañero.

Una de las principales dificultades del proyecto radica en que se cuentan con herramientas limitadas con respecto al manejo de la información por lo que es necesario que el programador planifique e implemente sus propias estrategias para solucionar el problema.

Otra dificultad importante es la necesidad de crear un algoritmo para la revisión de las preguntas, las cuales pueden ser de dos tipos, respuesta corta y selección única. Para la selección única es necesario manejar una lista de opciones de respuesta, el programa debe conocer la respuesta correcta pero además debe mostrar las respuestas de manera aleatoria. Por otra parte para la respuesta corta se necesita analizar un string como respuesta y analizar el nivel de coincidencia con la respuesta correcta.

Los métodos de inserción y borrado en algunas partes del programa representan también una dificultad, ya que es necesario establecer los enlaces necesarios para insertar un nuevo elemento en la lista de manera ordenada en algunos casos. De igual con el borrado es necesario liberar la memoria así como enlazar los elementos adyacente al objeto borrado.

0.3. Solución del problema

Para la Solución de este problema se utilizaron clases, en orden de la más general a la menos general la primera es la clase archivador, esta clase es la que le permite al usuario crear modificar y borrar exámenes. La siguiente clase es la clase examen, esta de manera general le permite al usuario añadir y borrar secciones, en esta clase se almacenan también las secciones en listas enlazadas cuya cabeza se almacena como un atributo de la clase. La siguiente clase es la clase sección la cual es la que almacena las preguntas con su respectiva información, además le permite al usuario añadir preguntas.

Para las preguntas de selección única se creo un struct preguntaSeleccionUnica el cual almacena las opciones en una lista circular, almacena también la cantidad de opciones, el texto de la pregunta y el puntaje. Es importante destacar que la respuesta correcta se almacena siempre en la cabeza y las otras respuestas se agregan a la lista circular. Se decidió hacer lista circular para implementar un algoritmo que permitiera mostrar las opciones siempre en orden aleatorio, el algoritmo genera un numero aleatorio entre 0 y la cantidad de opciones de la pregunta, luego se desplaza desde cabeza tantas veces como indique el numero aleatorio, finalizada la iteración se procede a imprimir las opciones y se agregan a un arreglo a como se van imprimiendo, cuando el usuario escoge una opción se verifica si la respuesta en esa posición del arreglo menos uno coincide con la respuesta correcta(cabeza) y desde ahí se define si la pregunta está correcta o incorrecta.

Para las preguntas de respuesta corta se creó un struct llamado preguntaRespuestaCorta el cual contiene la respuesta correcta, la respuesta del usuario, el texto de la pregunta y el puntaje, para calificar se utiliza un algoritmo que analiza cada char del string de la respuesta del usuario con la respuesta correcta y lleva un contador de los aciertos, en caso que el numero de aciertos se mayor o igual al sesenta por ciento de la longitud del string de respuesta correcta se considera la pregunta como correcta, en caso contrario se define como incorrecta.

Importante mencionar que ambos estruct almacenan un atributo booleano çorrecto.[el] cual después de que se califica se define como true si la pregunta es correcta o como false si la respuesta es incorrecta.

0.4. Análisis de resultados

Actividad	Finalizado	Observaciones
Crear clases archivador, sección, examen	Si	Ninguna
Crear estructuras de nodos	Si	Ninguna
Métodos en archivador	Si	Ninguna
Métodos en sección	Si	Ninguna
Métodos en examen	Si	Ninguna
Funciones en structs	Si	Ninguna
Crear examen para profesor	Si	Ninguna
Buscar, imprimir exámenes	Si	Ninguna
Crear secciones	Si	Ninguna
Crear preguntas en sección	Si	Ninguna
Preguntas selección única	Si	Ninguna
Preguntas respuesta corta	Si	Ninguna
Modificar y eliminar preguntas, secciones	Si	Ninguna
Preguntas ordenadas al agregar	Si	Ninguna
Ordenar preguntas según usuario	Si	Ninguna
Revisar preguntas según sección	Si	Ninguna
Obtener nota final	Si	Ninguna
Despliegue respuestas correctas, incorrectas	Si	Se muestra automáticamente
Criterio de evaluación respuesta corta	No	No se muestra
Validaciones	No	Incompletas

0.5. Conclusiones

Como resultado en el proyecto se puede confirmar que el uso de listas simples, dobles y circulares genera un mejor manejo de los datos en memoria, ya que manejarlos con punteros permite poder acceder con mayor seguridad y evitar duplicación de datos, perdida de información, entre otros problemas.

Se determinó que la utilización de listas simples y circulares son más sencillos que las listas dobles. Sin embargo, las listas dobles permiten un doble enlace de información, con lo que en el proceso de agregar preguntas se puede acceder a ellas ordenadas descendientemente o ascendentemente. Para el caso de selección única, es más útil manejar una lista circular que apunte a su cabeza, que en este caso fue la respuesta correcta. Para los demás recorridos se implementaron listas simples. Estos dos últimos no necesitan de un doble enlace, por lo que se pudieron implementar de esta forma.

La implementación de estructuras dentro del código ayudan al manejo de nodos de memoria para las listas. Mas como adición al proyecto se añadieron clases que entre sus atributos contiene los nodos creados por las estructuras y demás variables necesarias. Esto mejoró el proceso de acceso a los métodos y atributos que son específicos para cada clase. Además de que haya encapsulamiento de los atributos.

Con la ejecución del programa se verificó que hay un ambiente de interfaz relativamente fácil para el usuario, él puede desplazarse correctamente, crear exámenes, resolverlos y realizar cambios en ellos.

0.6. Recomendaciones

Para lograr ampliar el alcance del proyecto se pueden implementar varios aspectos. Uno de ellos es tener más flexibilidad a la hora de realizar el examen, pudiendo tener opciones para ir a la pregunta anterior y a la siguiente.

Un avance que se le puede agregar es el poder dar opciones al usuario de quienes son los que realizan el examen, ya que los estudiantes solo responden exámenes sin saber quien lo creó.

Otro agregado sería poder tener un registro de profesores y usuarios, ya que la funcionalidad de este programa permite que cualquier persona cree el examen y así mismo cualquiera pueda resolverlos.

Bibliografía

[1] Cicinelli, D., *Estructura de Datos: Lista.*, 2014.