**Universidad Simón Bolívar.**

**Ingeniería de Computación.**

**Laboratorio de Algoritmos y Estructuras II CI-2692.**

**Proyecto: Variantes de Registros**

**Académicos con Doble Indexación**

**Profesora:** Carmen Rosseline Rodríguez

**Integrantes:**

Argenis Chang 0810220

Vicente Santacoloma 0811044

**INTRODUCCION**

El problema atacado en el proyecto es el de construir dos variantes de implementación (o modelo concreto) para el TAD de un Registro Académico, mediante el lenguaje de programación Java.

El primer modelo concreto o variante es el de Arreglos, el cual lo implementa mediando dos arreglos que llevan la misma información académica pero en diferente orden, esto es, uno ordenado por Estudiante/Asignatura y el otro por Asignatura/Estudiante. El segundo es el de Árboles Binarios de Búsqueda, el cual de manera similar al de arreglos, utiliza dos árboles que contienen la misma información y con el mismo orden.

Ambos modelos funcionan a partir de una especificación con modelo abstracto TAD RegAcad.

Estas dos implementaciones de modelo concreto son utilizadas a partir de un programa cliente el cual es accesado por el usuario y le permite realizar una serie de operaciones referentes al registro académico.

En el informe a continuación, se presentaran tres aspectos importantes respecto al desarrollo del proyecto. Estas son: Diseño, el cual explica las decisiones tomadas que contribuyeron a la forma en que fue realizado el proyecto; Detalles de implementación, donde se comentan peculiaridades respecto a la utilización de Java, y el Estado Actual, que señala la operatividad del programa y da una explicación de todas la operaciones que intervienen en el proyecto.

**DISEÑO**

En los métodos Listar Asignaturas Por Estudiante y Listar Estudiantes Por Asignatura en la variante de árboles del TAD RegAcad, se decidió implementarla mediante iteración debido a que se nos exigía el uso de iteradores en estos métodos, y estos son los que nos permitía recorrer la estructura del árbol dado que cada elemento no estaba indexado.

En la función esCodigo de ambas implementación, fue construida como estática ya que una vez que esta recibe un valor, este no se modifica, y las operaciones se realizan utilizando ese valor.

Esta función no fue duplicada ya que es importada de un paquete regAcadUtil, en ambas implementación, sin embargo, si esta hubiese sido colocada en ambos modelos, funcionaria de igual manera ya que esta trabaja con los datos de entrada referentes a la información académica, y no con el modelo en sí, es decir, no depende de la estructura del modelo concreto.

Dado que se está importando el paquete de regAcadUtil, Java nos permite trabajar sobre las clases que se encuentren en dicho paquete, y por lo tanto como esCodigo es una de sus clases, este es heredado por las dos implementaciones.

En la implementación del TAD Árboles Binarios de Búsqueda, el uso de doble enlace no contribuyó en la construcción del iterador, ya que este método funcional hace el recorrido desde la raíz hasta el final de árbol, y no requiere de ninguna información previa de los progenitores, solo requiere la información académica del nodo y la de sus hijos.

Algunas de las decisiones empleadas por nosotros para el diseño del proyecto fue la de incluir los métodos de Listar Todas Estudiante/Asignatura y Listar Todas Asignatura/Estudiante dentro nuestros modelos concretos de Arreglos y Árboles. Estos lo hicimos ya que consideramos que dichas operaciones son propias del TAD y no del programa cliente.

Adicionalmente diseñamos el programa cliente mediante la construcción de varios métodos en lugar de uno solo que realizara todas esas operaciones. Esto con la finalidad de que le fuera más fácil al lector entender el código, y además para seguir las instrucciones indicadas en la guía de estilos de programación de Java, en la cual se establece que si un método es muy largo, este debe ser divido en otros sub-métodos.

**DETALLES DE IMPLEMENTACIÓN**

Con respecto al paso de parámetros en Java, en cada uno de los métodos de las implementaciones solamente se reciben los datos ingresados por el usuario, dado que dentro de los TADs se trabaja de manera implícita (referente al .this).

Otras peculiaridades de Java fue la herencia, es decir, hacer la importación de un paquete y luego poder acceder a sus clases. Esto en realidad fue favorable porque nos permite evitar duplicar ciertos métodos que se necesitan en ambas implementaciones. Por ejemplo: esCodigo y esCarne son necesarios en ambos modelos concretos (Arboles y Arreglos).

También cabe destacar los métodos funcionales de iteradores. Estos métodos nos permiten recorrer cualquier estructura sin tener que hacer uso de su modelo concreto en sí (en el proyecto esto fue lo que nos permitió recorrer la estructura de arreglos y árboles).

Además vale la pena mencionar el uso de excepciones, los cuales nos permiten construir programas robustos, es decir, el programa funciona bajo todos los parámetros incluyendo los casos en donde puedan generarse errores.

**ESTADO ACTUAL**

Actualmente el programa funciona perfectamente, sin embargo, hay problemas en los métodos de listar y eliminar, en donde no se cumple la precondición establecida de la interface. Pero según la explicación de varios profesores es un error de JML-JAVA (D.Mosquera).

**CONCLUSIÓN**

En la realización del proyecto hemos adquirido habilidades para la creación de manera eficiente de algoritmos que nos resuelvan un determinado problema. Además, mejoramos nuestra capacidad de entendimiento de un programa y el saber cuando realmente funciona.

En el aspecto teórico aprendimos la implementación de TADs, por medios de estructuras ya conocidas como los arreglos, y mediantes otros tipos de estructura que hayamos creado como los infoAcad, y además mediante el uso de apuntadores.

Por otra parte aprendimos nuevas herramientas que nos son de utilidad en la programación, como son: las excepciones que nos permiten pasar de un programa parcial a uno robusto; y los iteradores que nos sirven para recorrer diversos tipos de estructuras.

Por último, pero sin quitarle importancia, adquirimos conocimientos referentes al lenguaje de programación Java.

A lo largo de la realización del proyecto tuvimos algunas dificultades referentes a JML, donde se nos generaban problemas entre la interface y nuestras implementaciones del TAD Registro Académico (dichos problemas son naturales de JML). Además tuvimos dificultad para la creación de los iteradores por la complejidad que estos nos presentaron. Por otra parte nos costó cómo hacer para que el cliente manejara ambas implementaciones a la vez, pero sin embargo, esto lo pudimos solucionar.

Finalmente, algunas recomendaciones que damos es no demorar tanto en la entrega de las clases o paquetes que vayamos a utilizar ya que, esto nos retrasa en la realización del proyecto por la importancia que estos tienen en el mismo. Y de la misma manera el ser más claro en las cosas que tengamos que hacer, ya que al comienzo no nos dijeron que la interface nos la iban a dar y muchas personas comenzaron a hacerla.

**BIBLIOGRAFÍA**

* T.H. Cormen, C.E. Leiserson, R.L. Rivest & C. Stein, Introduction to Algorithms, The MIT Press, 2da. edición, 2002.
* B. Liskov con J. Guttag, Program Development in Java – Abstraction, Specification, and Object-Oriented Design, Addison-Wesley, 2001.