**MÉTODO DE LA INGENIERÍA: APLICACIÓN PARA LA SOLUCIÓN DE LA**

**TAREA INTEGRADORA 2**

**Rodríguez Amilcar Steban , Plaza Juan Felipe, Jojoa Jorge Eduardo y Nieto Juan Diego.**

**Universidad Icesi, Facultad de Ingeniería, Departamento de TIC, Computación y estructuras discretas 1.**

**Santiago de Cali, 2 de mayo del 2022.**

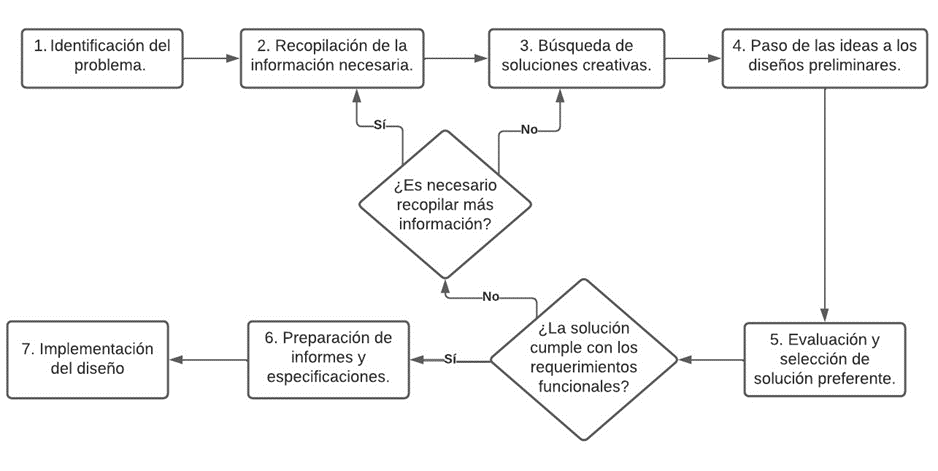
**Contexto problemático:**

Los profesores de la Facultad de Ingeniería contratan a su equipo de trabajo para que realicen una monitoria en un proyecto de investigación interna de la Universidad. El subproyecto para analizar consiste en el desarrollo de un prototipo de software que permita gestionar eficientemente las operaciones CRUD (Create, Read, Update y Delete) sobre una base de datos para simular la creación de un número de registros de personas de nuestro continente.

**Desarrollo de la Solución**

Para resolver la situación anterior se eligió el Método de la Ingeniería para desarrollar la solución siguiendo un enfoque sistemático y acorde con la situación problemática planteada.

Teniendo en cuenta la definición del método de la ingeniería del libro “Introducción a la Ingeniería” de Paul Wright, se definió el siguiente diagrama de flujo, cuyos pasos se seguirán para el desarrollo de la solución.



**Paso 1. Identificación del problema**

Se distinguen de forma concreta las necesidades de la situación problema y las condiciones bajo las cuales debe ser resuelta.

Identificación de las necesidades de la situación problema:

* Simular la creación de un número de personas para este continente con los siguientes datos: código (autogenerado), nombre, apellido, sexo, fecha de nacimiento, estatura, nacionalidad y fotografía.
* Buscar una persona en el programa. El usuario selecciona uno de los cuatros métodos de búsqueda para ingresar a la ventana de búsqueda de personas.
* Eliminar una persona del programa de simulación. Cabe resaltar, para realizar este proceso, se debe implementar el punto anterior sobre la búsqueda de una persona.
* Editar el registro de una persona. El usuario puede editar cualquier atributo de cualquier persona.

*Definición del problema*

Los profesores de la Facultad de Ingeniería requieren simular un prototipo de software que permita gestionar eficientemente las operaciones CRUD (Create, Read, Update y Delete) sobre una base de datos para la creación de registros de personas.

**Paso 2. Recopilación de la Información**

Para el planteamiento de la solución, es necesario conocer los conceptos involucrados en el desarrollo de esta. Por ende, a continuación, se hará énfasis en las definiciones de algunos tipos de estructuras de datos que se consideran, anticipadamente, harán parte de una posible solución final. Cabe resaltar, las definiciones que se presentan son tomadas de fuentes confiables y reconocidas.

***Definiciones.***

*Referencias:*

[*https://dataestructureii.files.wordpress.com/2015/11/arboles-n-arios.pdf*](https://dataestructureii.files.wordpress.com/2015/11/arboles-n-arios.pdf)

[*http://informatica.uv.es/iiguia/AED/teoria/apuntes/cuatr2/tema14.pdf*](http://informatica.uv.es/iiguia/AED/teoria/apuntes/cuatr2/tema14.pdf)

***Árboles n-arios:*** Un árbol n-ario puede tomarse como un conjunto finito de uno o más nodos, tal que existe un nodo especial, llamado nodo raíz, y donde los restantes nodos están separados en n≥0 conjuntos disjuntos, cada uno de los cuales es a su vez un árbol (llamados subárboles del nodo raíz). Se pueden encontrar tres clases de recorridos para este tipo de árboles: preorden, inorder y postorden.

***Árboles binarios:*** Un árbol binario es un conjunto finito de nodos que puede estar vacío o consistir en un nodo raíz y dos árboles binarios disjuntos, llamados subárbol izquierdo y subárbol derecho, es decir, que cada nodo puede tener como máximo dos nodos (izquierdo y derecho).

***Árboles binarios de búsqueda:***  cumple con las características de un árbol binario con las siguientes condiciones:

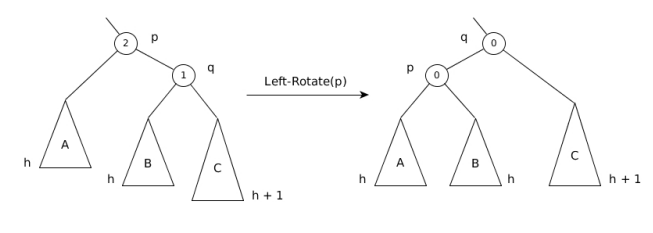
1. Todos los nodos están identificados por una clave y no existen dos elementos con la misma clave.
2. Las claves de los nodos del subárbol izquierdo son menores que la clave del nodo raíz.
3. Las claves de los nodos del subárbol derecho son mayores que la clave del nodo raíz.
4. Los subárboles izquierdo y derecho son también árboles binarios de búsqueda.

***Árboles binarios AVL:*** Es un árbol binario de búsqueda, pero posee unas condiciones sobre equilibrio que lo convierte en un caso especial de los árboles binarios:

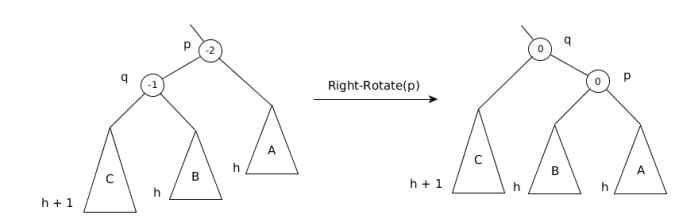
Cuando un nodo del árbol tiene una altura menor o igual a 1 pero mayor o igual a -1, se considera que el árbol AVL está en equilibrio.

Cuando la altura de un nodo es menor a -1 o mayor a 1, se considera que el árbol está en desequilibrio y se tienen que seguir una serie de pasos para balancear dicho árbol. Estos pasos se les conoce como rotaciones simples o compuestas y se pueden realizar tanto a la derecha como a la izquierda del árbol.

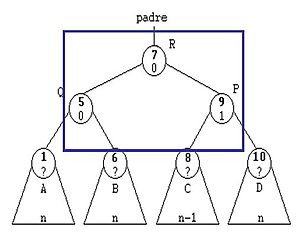
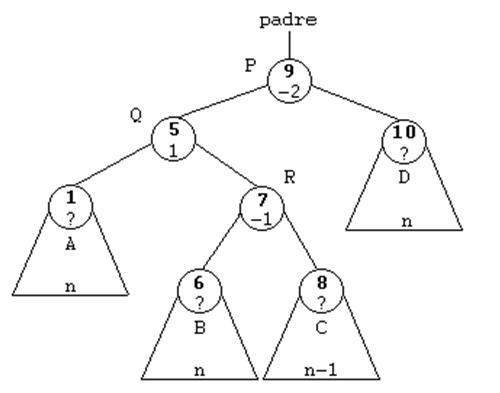
* **Rotaciones simples:**
  + **Rotación simple a la izquierda:** para trasladar un poco del peso del subárbol derecho hacia el izquierdo.

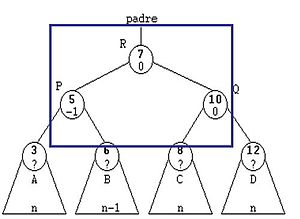
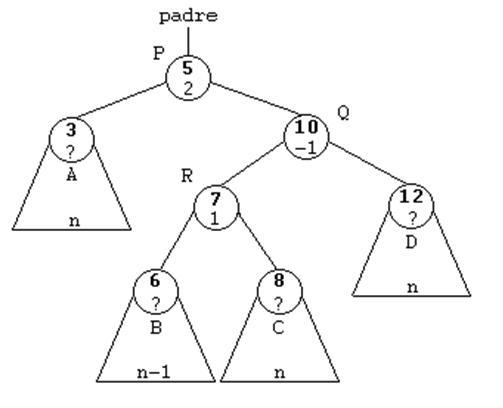


* + **Rotación simple a la derecha:** contrariamente, para llevar peso del subárbol izquierdo hacia el derecho



* **Rotaciones compuestas:**
  + **Rotación compuesta a la derecha:** La Rotación doble a la Derecha son dos rotaciones simples, primero rotación simple izquierda y luego rotación simple derecha.



* + **Rotación compuesta a la izquierda:** La Rotación doble a la Izquierda son dos rotaciones simples, primero rotación simple derecha y luego rotación simple izquierda.

***Persistencia:***

En la persistencia de objetos la información que persiste en la mayoría de los casos son los valores que contienen los atributos en ese momento, no necesariamente la funcionalidad que proveen sus métodos.

La persistencia permite almacenar, transferir y recuperar el estado de los objetos. Para esto existen varias técnicas:

* **Serialización:** consiste en un proceso de codificación de un objeto en un medio de almacenamiento (como puede ser un archivo, o un buffer de memoria).
* **Motores de persistencia:** Un motor de persistencia (DBMS: database managment system) es una aplicación o componente que resuelve la persistencia pero soporta además un número significativo de las siguientes características:
  + Encapsulamiento del medio persistente
  + Integridad
  + mecanismos que permitan mantener la correctitud de la información
  + Comunicación
  + lockeos
  + seguridad (autenticación y autorización)
  + Mecanismo para recuperarse
  + logs
  + backups
  + auditoría
  + Performance y escalabilidad.

**Paso 3. Búsqueda de Soluciones Creativas**

En primer lugar, se propone hacer un programa en el lenguaje y versión Java SE 8 con una interfaz gráfica de usuario. Teniendo en cuenta lo anterior, se elige la herramienta de diseño Scene Builder para la construcción del entorno gráfico. En segundo lugar, se propone crear una barra de menú que permita al usuario navegar a través de la aplicación, seleccionando opciones como simular la creación de un determinado número de personas, buscar el registro de una persona o eliminar el registro de una persona. En tercer lugar, se plantea guardar los registros de cada persona por medio de su nombre, apellido, nombre completo y código en diferentes tipos de árboles binario. Lo anterior, con el objetivo de facilitar el proceso de búsqueda de una persona en el programa. Además, se decide crear un paquete dentro de la clase model llamada “photos”, dado a que de esta forma el usuario puede simular el proceso de creación de personas, teniendo ligado una imagen de perfil para cada una de ellas.

**Paso 4. Transición de la Ideas a los diseños preliminares**

* **Scene Builder:** Una herramienta que básicamente nos ayudaría a crear el diseño de la interfaz gráfica, además una de las facilidades que nos brinda esta herramienta es que nos ayuda a crear el esquema de la programación de cada botón o cada componente implementado en el diseño.
* **Barra de menú:** Para facilitar la interfaz de usuario, se diseñó un menú amigable visualmente donde se puede crear, eliminar y buscar por nombre, apellido, nombre y apellido y código.
* **Sección de imágenes:** Para darle una identidad distinguida a cada persona, se le implementó al programa un paquete con una serie de imágenes que se le atribuyen de manera aleatoria a cada persona que se va creando.

**Paso 5. Evaluación y selección de la mejor solución.**

Para comenzar con este paso del método, cabe resaltar que ya se tiene una base para comenzar el proyecto. Por otro lado, para el manejo de datos se plantean dos posibles soluciones.

***Mejor solución.***

Para el manejo correcto de los datos se decidió utilizar el tipo de árbol llamado AVL. Principalmente porque es un árbol binario de búsqueda que satisface la condición de estar balanceado, de tal manera que la altura de cada nodo no pueda superar 1. Esta opción es muy útil al momento de manejar una gran cantidad de datos, así que las personas que se crearan, se irán almacenando en este árbol AVL y al mismo tiempo se irán organizando respetando la característica de los árboles binarios de búsqueda para facilitar la el manejo o la modificación de algún dato.

***Alternativa Arbol Rojo-Negro.***

Inicialmente se puede apreciar este tipo de árbol muy eficiente, ya que usa colores para marcar su equilibrio, incluso todos sus balanceos tienen como máximo 3 o 4 rotaciones, algo muy eficiente y rápido al momento de insertar y eliminar datos, sin embargo se decidió no usarlo, ya que se necesita un árbol estrictamente equilibrado y este claramente no lo hace con detallada rigurosidad.

**Paso 6. Preparación de Informes y Especificaciones**

*Especificación del Problema* (en términos de entrada y salida)

* Problema:
  + Prototipo de software que pueda gestionar eficientemente operaciones CRUD. Simular la creación de un gran número de registros con datos según unas condiciones dadas.

* Entradas:
  + Número de personas a crear.
  + Código, nombre, apellido o nombre completo de la persona para buscarla.
* Salidas:
  + Personas creadas, acorde con el número dado por el usuario del programa.
  + Resultado de la búsqueda de la persona.

*Consideraciones:*

Se debe tener en cuenta las siguientes consideraciones:

1. El código de la persona debe ser autogenerado al crear.
2. Al modificar o actualizar los datos de una persona todos sus campos se pueden modificar con excepción del código.
3. Los nombres deben ser tomados de una base de datos: <https://data.world/alexandra/baby-names>. Así mismo con los apellidos: <https://data.world/uscensusbureau/frequently-occurring-surnames-from-the-census-2010>. La edad : <https://www.indexmundi.com/es/estados_unidos/distribucion_por_edad.html>. Y por último la estatura: https://www.kaggle.com/datasets/tanuprabhu/population-by-country-2020.
4. La fotografía de la persona debe ser tomada de un generador aleatorio de rostros: <https://thispersondoesnotexist.com/>.

*Diagrama de clases:*

Diseño de clases del programa elegido para la solución.

