Método de ingeniería: Tarea Integradora 2
(Gestión Eficiente de Base de Datos de Personas)

Integrantes:

- *Cristian Felipe Perafan Chilito - A00378035*
- *Danna Alexandra Espinosa Arenas- A00378613*

1. IDENTIFICACIÓN DEL PROBLEMA:

Se requiere trabajar en el desarrollo de un prototipo de software que permita gestionar eficientemente las operaciones CRUD (Create, Read, Update and Delete) sobre una base de datos de personas de nuestro continente. La población del continente americano se estima, en 2020, en poco más de mil millones de personas, representando cerca del 13% del total mundial.

Se requiere además tener funciones tales como actualizar, borrar y buscar la información de una persona, para buscar se aplica mediante filtros tales como Nombre, Nombre Completo, Apellido y finalmente código, el cual se generará de forma automática y de manera única para cada persona que genere el sistema.

Requerimientos funcionales:

R1. El programa debe permitir generar y registrar a las personas con sus datos correspondientes(código (autogenerado), nombre, apellido, sexo, fecha de nacimiento, estatura, nacionalidad y fotografía.) con ayuda de las bases de datos sugeridas.

R2. El programa debe permitir funciones básicas del almacenamiento persistente (Crear, Leer, Actualizar y Borrar) con los datos del sistema, en especial los datos de las personas.

R3. El programa debe permitir actualizar los datos de una persona.

R4. El programa debe permitir eliminar a la persona seleccionada.

R5. El programa debe permitir crear a una persona con sus respectivos datos.

R6. El programa debe permitir realizar la búsqueda por Nombre y desplegar el listado de personas encontradas

R7.El programa debe permitir realizar la búsqueda por Apellido desplegar el listado de personas encontradas

R8. El programa debe permitir realizar la búsqueda por Nombre Completo desplegar el listado de personas encontradas

R9.El programa debe permitir realizar la búsqueda por código único y desplegar el listado de personas encontradas.

2.RECOPILACIÓN DE LA INFORMACIÓN NECESARIA:

La solución a la que va enfocada este método es a la solución del problema de buscar las personas que coincidan con los caracteres que se ingrese en la caja de texto en el árbol.

Marco Teórico:

Palabras Claves:

- ***Base de datos:***
  Una base de datos es una recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático.

  Fuente: https://www.oracle.com/co/database/what-is-database/
- ***Operaciones CRUD:***
  CRUD (Create, Read, Update, Delete) es un acrónimo para las maneras en las que se puede operar sobre información almacenada. Es un nemónico para las cuatro funciones del almacenamiento persistente. CRUD usualmente se refiere a operaciones llevadas a cabo en una base de datos o un almácen de datos, pero también pude aplicar a funciones de un nivel superior de una aplicación como soft deletes donde la información no es realmente eliminada, sino es marcada como eliminada a tráves de un estatus.

  Fuente:https://developer.mozilla.org/es/docs/Glossary/CRUD

- ***Árboles binarios de búsqueda:***
  Un árbol binario de búsqueda también llamado BST (acrónimo del inglés Binary Search Tree) es un tipo particular de árbol binario que presenta una estructura de datos en forma de árbol usada en informática.

  Fuente:https://es.wikipedia.org/wiki/%C3%81rbol_binario_de_b%C3%BAsqueda

- ***Árboles Trie:***
  Un trie es una estructura de datos de tipo árbol que permite la recuperación de información (de ahí su nombre del inglés reTRIEval). La información almacenada en un trie es un conjunto de claves, donde una clave es una secuencia de símbolos pertenecientes a un alfabeto. Las claves son almacenadas en las hojas del árbol y los nodos internos son pasarelas para guiar la búsqueda. El árbol se estructura de

forma que cada letra de la clave se sitúa en un nodo de forma que los hijos de un nodo representan las distintas posibilidades de símbolos diferentes que pueden continuar al símbolo representado por el nodo padre.

Fuente:https://es.wikipedia.org/wiki/Trie

- ***TextField:***
  En los programas de computadora, una caja de texto, cuadro de texto o caja de entrada de texto es un elemento común de una interfaz gráfica de usuario, también como el correspondiente tipo de widget usado al programar GUIs. El propósito de la caja de texto es permitir al usuario la entrada de información textual para ser usada por el programa.

  Fuente:https://es.wikipedia.org/wiki/Caja_de_texto

- ***Thread:***
  Un hilo es un hilo de ejecución en un programa. La máquina virtual de Java permite que una aplicación tenga varios subprocesos de ejecución que se ejecutan simultáneamente. Cada hilo tiene una prioridad. Los subprocesos con mayor prioridad se ejecutan con preferencia a los subprocesos con menor prioridad. Cada subproceso puede o no estar marcado como un demonio. Cuando el código que se ejecuta en algún subproceso crea un nuevo objeto Subproceso, el nuevo subproceso tiene su prioridad establecida inicialmente igual a la prioridad del subproceso de creación, y es un subproceso demonio si y sólo si el subproceso de creación es un demonio.

  Fuente:https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.html

- ***ObservableList:***
  Una lista que permite a los oyentes seguir los cambios cuando se producen

  Fuente:https://docs.oracle.com/javase/8/javafx/api/javafx/collections/ObservableList.html

3. BÚSQUEDA DE SOLUCIONES CREATIVAS:

- **Posible solución #1:**
  Ingresar los objetos personas en un arraylist y buscar dentro de este arraylist,retornar a la persona con la que coincida la búsqueda.
- **Posible solución #2:**
  Usar árboles trie para la búsqueda de las palabras que coincidan.
- **Posible solución #3:**
  Mediante hilos, ejecutar el algoritmo de búsqueda de los árboles, pasar los nodos que devuelva el método y agregarlos en un arrayList para posteriormente estos valores pasarlos a un observable List, para mostrar los resultados en una uitableview.
- **Posible solución #4:**
  Comparar con el metodo .contains si existe ese string en el arraylist de personas que ya antes se creó en el generador de personas

- **Posible solución #5:**
- **Posible solución #6:**

4. TRANSICIÓN DE LA FORMULACIÓN DE IDEAS A LOS DISEÑOS PRELIMINARES:

Posible solución #1 DESCARTADA.

La posible solución #1 se descarta debido a que no se emplean árboles, ni mucho menos los métodos de búsqueda de estos árboles.

Posible solución # DESCARTADA.

Posible solución # DESCARTADA.

5. EVALUACIÓN Y SELECCIÓN DE LA MEJOR SOLUCIÓN:

Rúbrica de selección:
Criterio A. Precisión de la solución
[2] Completa
[1] Parcial

Criterio B. Eficiencia.
[4] Constante
[3] Mayor a constante
[2] Logarítmica
[1] Lineal

Criterio C. Completitud.
[3] Todas
[2] Más de una si las hay, aunque no todas
[1] Sólo una o ninguna

Criterio D. Facilidad en implementación algorítmica.
[2] Compatible con las estructuras de datos de java.
[1] No compatible completamente con las estructuras de datos de java que se deben usar para la implementación de la solución.

	Criterio A	Criterio B	Criterio C	Criterio D	Total
Solución 2	**2**	**2**	**3**	**1**	**8**
Solución 3	**2**	**4**	**2**	**2**	**10**
Solución 4	**1**	**1**	**1**	**1**	**4**

La solución elegida es la solución 3 debido a que cumple con la mayoría de requisitos impuestos y tienen mejor puntaje dentro de la rúbrica.

Nota 1: No se empleó la búsqueda caracter por caracter debido a la complejidad.
Nota 2: El árbol puede generar hasta 1.000.000 de datos sin embargo se demora, para mayor eficiencia el límite es de 100.000 datos.