MÉTODO DE DISEÑO DE LA INGENIERÍA

COMPUTACIÓN Y ESTRUCTURAS DISCRETAS I

TAREA INTEGRADORA I

By:

Juan David Bahamon Rodriguez

A00375826

Carlos Javier Bolaños Riascos

A00377995

Samuel Hernandez Espitia

A00375392

David Esteban Peñaranda Scarpetta

A00375827

FASE 1: IDENTIFICACIÓN DEL PROBLEMA

La empresa Discreet Guys. Inc recurren a nuestro equipo de desarrolladores de software porque presentan la necesidad de "**simular el funcionamiento** de los ascensores de su nueva línea de edificios inteligentes" en la ciudad de Cali, Colombia.

Los edificios inteligentes de dicha empresa tienen diferentes tipos por lo que es de gran importancia determinar cómo se debe abordar el funcionamiento de los ascensores, ya que son un sistema de transporte que de no ser monitoreado y planeado de buena manera, puede presentar fallas y causar caos en la movilidad de las personas.

El programa requerido debe permitir simular los elementos importantes del proyecto de los edificios inteligentes, como: los edificios, pisos, oficinas, y por supuesto la movilidad de las personas. Para esto, se debe tener en cuenta las siguientes condiciones:

1. Las personas viajan a una determinada oficina, por lo que se necesita direccionar a una persona a un piso y una oficina en específico dada una entrada.
2. Ya que las personas viajan a diferentes pisos, se debe tener un control en el orden de llegada para direccionar y movilizar a las personas a los respectivos pisos del edificio.
3. Al igual que al control de llegada, se debe tener un control en la dirección en la que viaja el ascensor (puede ir en subida o bajada) para evitar posibles fallas e inconformidades con las personas que usen el sistema de elevación.
4. Por las condiciones del edificio, se debe tener control en el ingreso a las oficinas. Solo puede haber una persona en cada oficina, por lo que si una persona quiere ingresar a un módulo ya ocupado, se debe restringir su acceso. Las oficinas de cada edificio son numeradas en orden "decreciente", es decir que las del piso 1 son las que tienen mayor numeración.

5. Al ser un programa simulado, se necesita visualizar todos los movimientos o prohibiciones de acceso en la consola para poder evidenciar y probar el funcionamientos de los ascensores en los edificios dadas unas entradas ajustadas a casos reales que se podrían presentar en el día a día.

FASE 2: RECOPILACIÓN DE INFORMACIÓN NECESARIA

Para iniciar el proceso de diseño y desarrollo es necesario tener una base o contexto del funcionamiento de un ascensor, las herramientas que permiten desarrollar el sistema, y las estructuras de datos que nos facilitarian la implementación del simulador, para ello se debe de tener en cuenta la siguiente información:

- **Un ascensor** es un aparato elevador que sirve para transportar personas o cosas de unos pisos a otros en un edificio.
- Los elementos de **la cola** se añaden y se eliminan de tal manera que el primero en entrar es el primero en salir. La adición de elementos se realiza a través de una operación llamada encolar (enqueue), mientras que la eliminación se denomina desencolar (dequeue). La operación de encolar inserta elementos por un extremo de la cola, mientras que la de desencolar los elimina por el otro. https://docs.oracle.com/javase/7/docs/api/java/util/Queue.html
- **La pila** es una secuencia de elementos en donde el acceso de la pila se realiza siempre sobre un único extremo. Su funcionamiento consiste en introducir un nuevo elemento sobre la cima (push) y la de extraer el elemento situado en la cima (pop). https://docs.oracle.com/javase/7/docs/api/java/util/Stack.html
- **Una cola de prioridad** es una cola especial en la que todos los elementos de la cola se ordenan según el orden natural o utilizando un comparador personalizado se denomina cola de prioridad. https://runebook.dev/es/docs/openjdk/java.base/java/util/priorityqueue
- **Una Hashtable** es una estructura de datos que utiliza una función hash para identificar datos mediante una llave o clave (ej. Nombre de una persona). La función hash transforma una llave a un valor índice de un arreglo de elementos. https://docs.oracle.com/javase/8/docs/api/java/util/Hashtable.html

FASE 3: BÚSQUEDA DE SOLUCIONES CREATIVAS

El simulador puede ser implementado de varias formas con las estructuras de datos aprendidas en el curso de "Computación y estructuras discretas I" por lo que las posibles ideas de solución son:

Alternativa 1:

Desarrollo del sistema utilizando estructuras de datos (colas, pilas, colas de prioridad, tablas hash y LinkedList) ya implementadas por el lenguaje de programación java, de tal manera que satisfaga todos los requerimientos. El modelo del programa debe tener paquete ui y model, cada uno con sus respectivas responsabilidades y clases necesarias.

Alternativa 2:

El sistema de simulación será diseñado con las estructuras de datos (colas, pilas, colas de prioridad, tablas hash y LinkedList) construidas por el equipo de desarrolladores. Las estructuras seguirán los TAD respectivos para tener un buen funcionamiento. Además, se tendrá en cuenta la división de responsabilidades entre paquetes y clases, siguiendo las buenas prácticas de la ingeniería de software.

Alternativa 3:

Desarrollar el programa creando cada uno de los métodos necesarios y realizando sus respectivas asignaciones de los atributos sin necesidad de la implementación de estructuras de datos.

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

Alternativa 1: Esta idea tendrá específicamente la interacción de las estructuras de datos :

- Pilas
- Colas
- Hash table

- LinkedList
- Prioridad de colas

Cada una de estas contendrá los objetos de las clases Edifice, Elevator, Floor, Office y Person. Así, podemos desarrollar e implementar el comportamiento del ascensor en los edificios ingresados por el usuario.

Alternativa 2:

- El equipo de desarrolladores se enfoca en diseñar sus propias estructuras de datos para ser implementadas en la simulación.

Alternativa 3: Por falta de coherencia y funcionalidad, se ha descartado esta idea. Nuestro cliente, solicita que el comportamiento del simulador sea lo más parecido a la realidad, por lo que la herramienta eficaz para llegar a esta meta es utilizar las estructuras de datos vistas en el curso de **Computación y Estructuras discretas I**

FASE 5: EVALUACIÓN Y SELECCIÓN DE LA MEJOR SOLUCIÓN

Teniendo en cuenta las diferentes posibles soluciones se ha realizado un análisis de la solución más factible, con base en los siguientes criterios:

Criterio A. Dirección de la persona: Permite conocer hacia dónde se dirige la persona, es decir, a qué edificio y oficina se dirige.

Criterio B. Orden de ingreso y salida: Permite realizar el ingreso de las personas de acuerdo al orden de llegada del ascensor. Además, se realiza de forma correcta la salida de las personas de orden inverso.

Criterio C. Dirección del ascensor: El direccionamiento del ascensor es de forma correcta, se dirige en el orden en el que las personas pulsan el botón, teniendo en cuenta la dirección hacia donde va el ascensor.

Criterio D. Apartado de consultas: Permite el acceso de forma rápida a las personas que están en determinada oficina.

Criterio E. Estructura de datos: Se implementa la estructura de datos más eficiente y que brinde la mejor solución acorde a la problemática.

Evaluación: Siendo 0 muy malo y 5 muy bueno.

	Criterio A	Criterio B	Criterio C	Criterio D	Criterio E	Total
Alternativa 1	**5**	**5**	**5**	**5**	**5**	**25**
Alternativa 2	**5**	**5**	**5**	**5**	**2**	**22**

La mejor solución es la alternativa 1.

FASE 6: PREPARACIÓN DE INFORMES Y ESPECIFICACIONES

Especificación del problema (entradas - salidas)

Problema: Simular el movidad en el edificio de las personas

Entradas: Se van a ingresar muchas líneas según sea el caso. En la primera, ingresa el número de edificios. En la segunda, se ingresa el identificador de cada edificio acompañado del número de personas que están en el edificio, la cantidad de pisos y por último la cantidad de oficinas por piso. Según sea la cantidad de personas en el edificio se ingresa el nombre de la persona, el piso en el que está actualmente y finalmente, al piso que se dirige. Las entradas del simulador se ingresarán según sea el caso que haya ingresado el usuario. Por ejemplo:

2 → Número de edificios

A 4 2 2 → Identificador edificio, cantidad de personas, cantidad de pisos, oficinas por piso

Carlos 1 2 → Nombre de la persona, piso en el que está, piso al que se dirige.

David 2 1

Samuel 1 2

Juan 2 1

B 5 2 2

Esteban 1 2

Javier 2 1

Diego 1 2

Felipe 2 1

Camilo 2 1

Salidas: Se muestran los movimientos por cada edificio de forma ordenada. Por ejemplo:

Movimientos edificio A:

Carlos se mueve a la oficina: 1

Samuel se mueve a la oficina: 2

David se mueve a la oficina: 3

Juan se mueve a la oficina: 4

Estado final de las oficinas en el edificio A

[Carlos, Samuel, David, Juan]

Movimientos edificio B:

Esteban se mueve a oficina: 1

Diego se mueve a oficina: 2

Camilo no puede ser incluido en las oficinas del edificio

Javier se mueve a oficina: 3

Felipe se mueve a oficina: 4

Estado final de las oficinas en el edificio B

[Esteban, Diego, Javier, Felipe]

Consideraciones

1. En este caso, como los edificios tienen 2 pisos, se enumeran las oficinas del 1 hasta el 4. Se empieza con pisos menores, es decir que en el piso 1 estarán las oficinas 3 y 4; en el piso 2 estarán las oficinas 1 y 2.
2. En caso de que una persona no pueda acceder a una oficina porque esta ya está ocupada, se debe notificar en la consola.

FASE 7: IMPLEMENTACIÓN DEL DISEÑO

La implementación del diseño se desarrolla en el lenguaje de programación: Java. Haciendo uso de las estructuras de datos: colas, pilas, colas de prioridad y tablas hash.

Lista de tareas a implementar:

1. Conocer a qué oficina se dirige la persona.
2. Realizar el ingreso de la persona por orden de llegada del ascensor.
3. Realizar la salida de la persona en orden inverso.
4. El ascensor se dirige al piso que ha sido llamado por una persona, teniendo en cuenta hacia dónde se dirige el ascensor.
5. Se tiene un acceso rápido de las personas que se encuentran en cierta oficina.
6. Validar que la oficina tenga disponibilidad.

   6.1. Notificar el nombre de la persona que se quedó sin oficina en caso de que esta se encuentre llena.