MÉTODO DE INGENIERÍA.

Juan Esteban Duque Taborda (A00376778).

Sebastián Escobar Marín (A00374994).

**FASE 1: IDENTIFICACIÓN DEL PROBLEMA.**

El problema que debemos solucionar con base en lo que nos piden es modelar un sistema de edificios inteligentes donde cada uno tiene sus pisos, oficinas, ascensores y personas que trabajan en dichos edificios. Para esto se necesita modelar la cantidad de edificios, cada uno con un identificador, los pisos por cada edificio pueden variar, pero el número de oficinas de cada piso según el edificio no varía, por ejemplo, un edificio puede tener dos pisos y todos los pisos de ese edificio tener 2 oficinas, pero otro con 5 pisos y cada piso dentro de ese edificio con 5 oficinas también, la cantidad de oficinas por piso depende del edificio. Seguidamente, se modelan las oficinas que solo puede abarcar una persona, además estas van de forma ascendente como los pisos, por ejemplo, si en un edificio hay dos pisos, en cada piso hay dos oficinas, entonces en el último piso estará la oficina #1 y #2 y en el primer piso estarán las oficinas #3 y #4. También, se modelan los ascensores de cada edificio y su funcionamiento el cual tiene un sistema de prioridad dependiendo de si suben o bajan, cuando suben la prioridad va para las personas que suben y viceversa. Y por último se modelan las personas, antes de tener personas en el edificio se definen las oficinas de cada persona, pero hay un problema con esto, como se dijo anteriormente, solo puede haber una persona por oficina, entonces ¿qué pasa si dos o más personas tienen la misma oficina?, pues esta oficina será ocupada por el primero que llegue a dicha oficina y los demás quedarán por fuera y se debe mostrar la cantidad de personas que quedaron por fuera. El tiempo de estas personas se miden según el ascensor y quien pueda llegar primero.

**FASE 2: RECOPILACIÓN DE INFORMACIÓN NECESARIA**

Lista de Enlace Simple:

Una lista de enlace simple es una lista enlazada de nodos, donde cada nodo tiene un único campo de enlace. Una variable de referencia contiene una referencia al primer nodo, cada nodo (excepto el último) enlaza con el nodo siguiente, y el enlace del último nodo contiene null para indicar el final de la lista. Aunque normalmente a la variable de referencia se la suele llamar top, usted puede elegir el nombre que quiera. La siguiente figura presenta una lista de enlace simple de tres nodos, donde top referencia al nodo A, A conecta con B y B conecta con C y C es el nodo final.

Una Hashtable Java 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. En este caso a un índice de nuestra Hashtable Java.

La clase Stack es una clase de las llamadas de tipo LIFO (Last In - First Out, o último en entrar - primero en salir). Las operaciones básicas son push (que introduce un elemento en la pila), pop (que saca un elemento de la pila), peek (consulta el primer elemento de la cima de la pila), empty (que comprueba si la pila está vacía) y search (que busca un determinado elemento dentro de la pila y devuelve su posición dentro de ella).

Desde el punto de vista teórico una cola es un tipo de dato abstracto (TDA) que sigue el principio FIFO (primero en entrar, primero en salir) que implica que el primer elemento en ser agregado en la cola es también el primero en ser removido de la misma. Cuando se inserta un nuevo elemento lo hace al final de la cola, y el primer elemento es el que se elimina al ejecutar la acción eliminar.

**FASE 3: BÚSQUEDA DE SOLUCIONES CREATIVAS**

1ra idea: Manejas los edificios como un arreglo puesto que se ingresa la cantidad de edificios que se utilizarán en el programa.

2da idea: Manejar las oficinas como un arreglo puesto que se ingresa el número de oficinas por cada edificio, lo que nos permitiría tener un arreglo fijo de oficinas igual para cada piso dentro del edificio estipulado.

3ra idea: Cada oficina del edificio se puede manejar con arraylist, pero esto debe tener en cuenta que para todos los pisos el número de oficinas es igual. Además, que se tendría en cuenta que no se pueden añadir más personas de las oficinas estipuladas.

4ta idea: Tener un hashTable para modelar los pisos y las oficinas que hay en cada edificio. Esto porque se ingresa la cantidad de pisos por edificio que es el tamaño de la hashtable y dentro de esa se guardan nodos en donde está cada oficina del piso.

5ta idea: Manejar las personas en un arraylist ya que se ingresa el número de personas totales, estas tienen que ser más que la cantidad de oficinas porque siempre se llenan, pero puede haber empleados que sobren.

6ta idea: Recibir las personas en un arreglo del tamaño ingresado para luego pasarlo a una cola en la q se ordena según la persona que menos se vaya a mover en el edificio hasta la que más de mueve. Y con ese orden se agregan al ascensor, pero no solo cuando se inicia el programa sino en cada piso.

7ma idea: Usar listas enlazadas para las oficinas, así se puede tener crear una sola lista por edificio, que el identificador de la oficina cambia según la oficina que sea y pueda añadirse la misma cantidad de oficinas a cada piso, pero con su identificador correspondiente.

**FASE 4: TRANSICIÓN DE LA FORMULACIÓN DE IDEAS A LOS DISEÑOS PRELIMINARE**

Las ideas que no son factibles son la segunda idea, esto porque no es necesario tener un arraylist cuando se sabe la cantidad de edificios que se tendrán. La cuarta idea también funciona igual que la anterior, no es necesario tener este método cuando se sabe la cantidad. Las otras ideas se moldean de manera adecuada al problema y se obtendría una solución muy cercana a la deseada. Esto porque los edificios se podrían recibir en un arreglo porque nos ingresan la cantidad de edificios, dentro de este irían las hashtables de cada edificio, las cuales contendrían el modelo de los pisos y oficinas, para las oficinas se tendría una lista enlazada para conocer la oficina siguiente o la trasera en el caso de que haya, para así tener un orden más claro en cuanto a las oficinas totales de cada edificio y las cuales se les pueda asignar las personas necesarias. Cada edificio tiene su ascensor, este funcionaría según la prioridad de la persona. Esto usa un stack y la manera en la que se añaden las personas a ese stack es con un queue la cual acomodaría a las personas según al piso que deben ir. A este queue se añaden las personas luego de recibirlas en un arreglo normal con la cantidad de empleados ingresadas al inicio.

**FASE 5: EVALUACIÓN Y SELECCIÓN DE LA MEJOR SOLUCIÓN.**

Los edificios reciben un arreglo porque se ingresan la cantidad de edificios, dentro de cada edificio va una HashTable, cada una conteniendo el modelo de los pisos y oficinas, las oficinas se realizarán a partir de una lista enlazada para conocer la oficina siguiente o la trasera en el caso de que haya, para así tener un orden más claro en cuanto a las oficinas totales de cada edificio y las cuales se les pueda asignar las personas necesarias. Cada edificio tiene su ascensor, este funciona según la prioridad de la persona, que la que recorra menos pisos o la que suba. Para esto se usa un Stack y la manera en la que se añaden las personas a ese Stack es con un Queue la cual ordena las personas según al piso que deben ir, la oficina y el recorrido que se tiene que hacer según el piso en el que están. A este Queue se añaden las personas luego de recibirlas en un arreglo normal con la cantidad de empleados ingresadas al inicio.