****

**Universidad De Antioquia**

**Facultad de ingeniería**

**Materia:**

**Informática II**

**Trabajo presentado por:**

**Carlos Manuel Gutiérrez Peñafiel**

**Fabio Andrés García Pérez**

**Presentado a:**

**Augusto Enrique Salazar Jimenez**

**Medellín, Colombia**

**Análisis del problema:**

Para el ejercicio planteado propone hacer una interfaz en la que un usuario pueda crear una red metro con distintas opciones como en el siguiente menú:

**A.** Agregar una estación a una línea.

**B.** Eliminar una estación de una línea.

**C.** Saber cuántas líneas tiene una red Metro.

**D.** Saber cuántas estaciones tiene una línea dada.

**E.** Saber si una estación dada pertenece a una línea específica.

**F.** Agregar una línea a la red Metro.

**G.** Eliminar una línea de la red Metro.

**H.** Saber cuántas estaciones tiene una red Metro.

para la solución pensada, se planea en una primera instancia cuando no hay ninguna línea ni estaciones en nuestra red metro hacer lo siguiente:

1. pedir al usuario que ingrese el nombre para la primera línea, está esta línea será considerada como un objeto con los siguientes atributos:

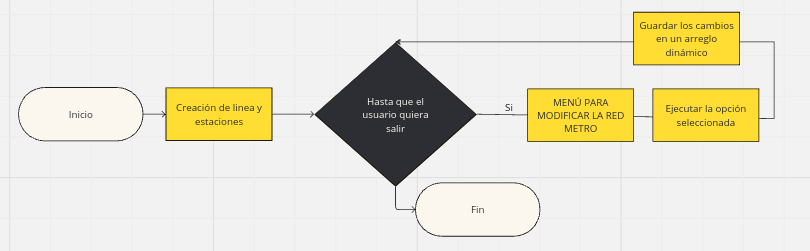
* Nombre de la línea.
* Estaciones.

1. Se pedirá al usuario que ingrese las estaciones para que la línea se pueda crear correctamente. cada estación será un objeto con los siguientes atributos:

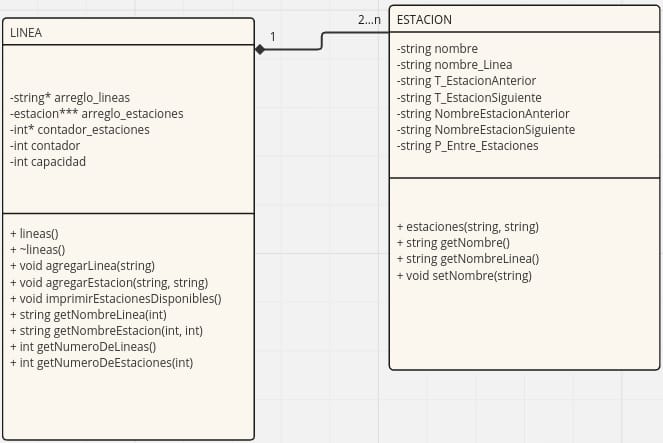
* Nombre de la línea a la que pertenece.
* Nombre de la estación.
* Tiempo de llegada hasta la estación anterior.
* Tiempo de llegada hasta la estación siguiente.
* Nombre de la estación anterior.
* Nombre de la siguiente estación.
* Entre qué estaciones desea crear la nueva estación (Solo si ya ha creado un mínimo de 2 estaciones en la línea principal).



Las estaciones se manipularán a través de un arreglo dinámico de objetos. Una vez que se tenga mínimo una línea con 2 estaciones, se entrará en un ciclo mostrando las opciones que tiene el usuario para manipular la red metro, este ciclo terminará hasta que el usuario ya no desee seguir modificando la red metro.



**Diagrama de clases:**



**Algoritmos Implementados:**

En la clase Estación se usaron los siguientes métodos:

|  |
| --- |
| 1.**String getNombre ():** Método que retorna el nombre de la estación en una variable tipo string.  2.**String getNombreLinea ():** método que retorna en una variable tipo string el nombre de la línea a la que pertenece la estación.  3.**Void setNombre (string):** método set para asignar el nombre de la estación.  4.**Void setEsTransferencia(bool):** método que tiene como parámetro un bool y modifica el valor del atributo de la estación “es\_trasferencia”, asignándole true o false dependiendo del caso.  5**.Bool getEstransferencia ():** Método get para determinar si una estación es de transferencia o no. |

En la clase Línea se tienen los siguientes métodos:

|  |
| --- |
| 1.**Void agregarLinea (String):** método que tiene como parámetro el nombre de la línea que se desea crear, verificando que no se encuentre otra línea con el mismo nombre ya creada anteriormente.  2.**Void agregarLinea ():** método usado para crear mínimo una línea y dos estaciones.  3.**Void eliminar Línea(string):** método que tiene como parámetro el nombre de la línea que se desea eliminar, verificando que no tenga una estación de transferencia.  4.**Void agregarEstacion (string, string):** método que tiene como parámetro el nombre de la estación y el nombre de la línea a la que se va a agregar, verificando que no exista otra estación con el mismo nombre.  5.**Void eliminarEstacion (string, string):** método que tiene como parámetros el nombre de la estación que se desea eliminar y en qué línea se encuentra, verificando que la estación exista.  6.**Void imprimirEstacionesDisponibles** (): método que imprime por pantalla todas las líneas creadas con sus respectivas estaciones.  7.**String getNombreLinea (int):** Método get que tiene como parámetro un entero que sirve para retornar el nombre de una línea de un arreglo en la posición del entero que esta como parámetro.  8.**String getNombreEstacion (int, int):** método get que tiene como parámetros dos enteros para acceder a los atributos de una estación a través del triple puntero.  9.**Int getNumeroDeLineas ():** método get que retorna contador con el valor del número de líneas creadas.  10.**int getNumeroDeEstaciones(int**): método get que tiene como parámetro un entero para retornar el numero de estaciones ubicadas en un arreglo en la posición del parámetro. |