**Análisis**

A fin de realizar una simulación de una red de metro con las herramientas vistas hasta el momento en curso y con un enfoque de orientación a objetos se empezó con una breve visualización y comprensión de las partes esenciales que componen una red de metro y cómo interactúan entre ellas. Para ello se realizaron algunas preguntas de sonda cómo:

* ¿De qué elementos se compone o contiene esencialmente una red?
* Si existe, ¿Cuál es la relación de cardinalidad que existe entre estas partes?
* ¿Interactúan las partes de otras maneras entre sí?
* Para efectos de este desafío, ¿Qué atributos componen cada parte?
* De los subprogramas mencionados en el documento presentación, ¿A qué parte o clase pertenece cada uno de ellos?
* ¿Es necesario el reúso de alguna clase externa?

Adicionalmente se tienen en cuenta las consideraciones iniciales mencionadas en el documento presentación y en la clase conversatorio sobre el mismo, como el no uso de STL, que una línea que tenga un estacón de transferencia no puede ser eliminada, no hay bifurcaciones ni bucles, etc...

Por otro lado se considera el uso de apuntadores como argumentos de los métodos y memoria dinámica para el almacenamiento de los arreglos de las líneas y las estaciones, con el fin de cumplir con el criterio de eficiencia.

**Clases**

* **Diagrama de clases:**

Para la solución se prevé la codificación de tres clases: Red, Linea y Estación, además del reúso de la clase string para almacenar los nombres de cada uno de los objetos de cada clase:

Diagrama

Descripción generada automáticamente

* **Descripción de los subprogramas:**
* **Descripción de los subprogramas:**

1. **Agregar una estación a una línea, en los extremos o en posiciones intermedias**

* **Entradas:**

1. Nueva estación (\*string).
2. Estación inmediatamente anterior en la línea o extremos (-1 o 1), (\*string).
3. Valor del tiempo para llegar a la estación anterior(\*int).
4. Valor del tiempo para llegar a la estación siguiente(\*int).
5. Indicador de si es de transferencia o no (\*bool).

* **Procedimiento:**

1. Contar el número de espacios del array que se encuentran ocupados.
2. Si no hay espacios disponibles en el arreglo, se crea un nuevo arreglo dinámico con n+2 espacios.
3. Asignar las estaciones en el mismo orden.
4. Asignar la nueva estación en su posición.
5. Modificar los tiempos de las estaciones adyacentes.
6. Completar el arreglo.
7. Liberar memoria del arreglo anterior si se creó uno nuevo y almacenar el nuevo apuntador al arreglo y a la última posición del mismo.

* **Salida:** Ninguna (Se modifica el apuntador o el arreglo del atributo directamente)

1. **Eliminar una estación de una línea. No se pueden eliminar estaciones de transferencia.**

* **Entradas:**

1. Estación a eliminar (\*string).
2. Tiempo entre las dos estaciones conexas a la que se va a eliminar. (\*int)

* **Procedimiento:**

1. Recorrer la línea en busca de la estación.
2. En la posición de la estación a eliminar se asigna la siguiente y se modifica el tiempo de las respectivas estaciones conexas.
3. Completar el arreglo asignando a la estación anterior la siguiente.
4. Se ingresa NULL en la última posición.

* **Salida:** Ninguna (Se modifica el apuntador o el arreglo del atributo directamente)

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

* **Entradas:** Ninguna
* **Procedimiento:**

1. Recorrer el arreglo dinámico de líneas.
2. Contar las líneas hasta que se encuentre NULL o se llegue a la posición final del arreglo.
3. Devolver el contador.

* **Salida:** Contador con la cantidad de líneas (int).

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

* **Entradas:**

1. Línea a la cual se le quiere contar las estaciones (string)

* **Procedimiento:**

1. Recorrer el arreglo dinámico de líneas en busca de la línea.
2. En la linea, recorrer el respectivo arreglo dinámico de estaciones.
3. Contar las estaciones hasta que se encuentre NULL o se llegue a la posición final del arreglo.
4. Devolver el contador.

* **Salida:** Contador con la cantidad de estaciones (int).

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

Primero tenemos que tener en cuenta que ya vamos a poseer un nombre de la estación tipo Estación y un arreglo tipo Estación, ahora vamos a realizar una búsqueda iterando sobre este arreglo para identificar si el nombre de la estación se encuentra en este arreglo tipo Estación, dependiendo de esto, se retornara un valor verdadero si el nombre de la estación fue encontrada y falso si el nombre de estación no fue encontrada

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

Al invocar esta función se realizará la creación de 2 objetos tipo  Estación y  se van a almacenar en un arreglo tipo Estación, finalizando asi la creación del objeto  Linea. Ahora la agregaremos a la red entonces, se realizará la creación de un arreglo dinámico tipo Linea con un valor superior al que ya teníamos antes y también crearemos un puntero tipo linea para borrar el contenido de este arreglo, se realizará un proceso el cual pasa los datos del primer arreglo de menor espacio al de mayor espacio quedando un espacio libre, después de esto borraremos el arreglo de menor espacio con el puntero así liberando este espacio y luego teniendo agregaremos el objeto tipo Linea a este arreglo tipo Linea.

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

En esta función se  creara un arreglo de tipo linea de un espacio menor al anterior y se identificara en el arreglo tipo Linea cual es la linea que debe ser eliminada, luego pasaremos la información de arreglo grande al arreglo pequeño evitando la información del objeto que debe ser eliminado, obteniendo asi el arreglo pequeño sin el objeto linea que deseamos eliminar, después de esto se borrara el arreglo grande mediante punteros.

1. **Saber cuántas líneas tiene la red.**

Se realizará la iteración sobre los objetos tipo Linea, se pasara sobre estos con un contador para que al finalizar la cuenta nos entre el numero de cuantos objetos conforman el arreglo

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

Se realizará la iteración sobre los objetos tipo Linea, se pasara sobre estos con 2 contadores, uno para identificar la cuenta  de las estaciones totales y el segundo para identificar la cuenta de las estaciones de trasferencia , al final restaremos a las estaciones que encontramos con las estaciones de transferencia dándonos asi las estaciones totales.