UNIVERSIDAD DE ANTIOQUIA
Facultad de Ingeniería

Informática II
Semestre 2024 – 1
Desafío II. 24 de abril de 2024

Desafío II: Simulación de una red de Metro

Un programa de simulación pretende modelar una situación del mundo real con el objetivo de estudiarla y aprender algo acerca de ella. La simulación está delimitada por un contexto, en el cual los objetos y acciones de interés en la situación real, tiene una representación en el programa.

Si el programa representa un modelo apropiado del mundo real, entonces los resultados de su ejecución reflejan, con cierta confianza, los resultados reales de las acciones simuladas.

Objetivos

- Desarrollar la capacidad de solución de problemas en los estudiantes, enfrentándolos a problemáticas de la vida cotidiana.
- Verificar si el estudiante adquirió las destrezas y conocimientos fundamentales de la programación orientada a objetos en C++: abstracción, encapsulación, relaciones básicas, diseño de diagrama de clases, funciones amigas, sobrecarga y uso de plantillas.

El objetivo de esta actividad es poner a prueba sus destrezas en análisis de problemas y manejo de la POO en C++. Si usted ha llevado un proceso disciplinado de aprendizaje a lo largo del semestre, esta es una oportunidad de demostrarlo. Podrá plantear una solución viable con un resultado satisfactorio. En caso contrario, podrá identificar sus debilidades y tomar medidas a fin de ser apto para abordar situaciones similares a las planteadas en este desafío.

Trate de valorar la verdadera complejidad del problema planteado, no se rinda antes de intentarlo o de plantear los posibles escenarios de solución. Se dará cuenta que si bien, al principio le puede parecer difícil, realmente ya ha tenido la oportunidad de enfrentarse a problemas similares. Si se toma el tiempo adecuado para analizar, el proceso de codificación será eficaz y no le tomará mucho tiempo.

Esperamos que disfrute del desafío propuesto. Lea primero todo el documento antes de comenzar y asegúrese de entender muy bien las instrucciones antes de desarrollar esta actividad evaluativa.

Fue revisado por los profesores Aníbal Guerra y Augusto Salazar.

Introducción

El Metro es uno de los sistemas de transporte más utilizado en las grandes ciudades, siendo eficaz para trasladar grandes masas de usuarios en poco tiempo. Se requiere que usted diseñe e implemente un simulador de Metro que permita modelar algunas características del funcionamiento de este sistema de transporte.

Figura 1. Red del Metro de Ciudad de México. Tomado de : https://www.metro.cdmx.gob.mx/

Un Metro puede concebirse como una red (la red Metro). Una red, como la que se muestra en la Figura 1, es un conjunto de líneas dentro de un área específica. Una línea se identifica por su nombre. A cada línea se le especifica un tipo de transporte: tren o tranvía, así como la secuencia de estaciones que comunica. Una línea no tiene bifurcaciones ni bucles, sin embargo, el tránsito dentro de cualquier línea es bidireccional.

Una estación está identificada por su nombre. Para cada estación, también se conoce el tiempo en segundos que toma llegar desde la estación actual, hasta la estación siguiente o a la estación anterior dentro de su misma línea. Algunas estaciones pueden tener una categoría especial, llamándose estaciones de transferencia. Las estaciones de transferencia representan puntos de intersección entre dos líneas diferentes de la red. Por ejemplo en el metro de Medellín la estación San Antonio es una estación de transferencia donde confluyen San Antonio A, San Antonio B y San Antonio-Tran.

A efectos de esta simulación de metro, existen elementos de tipo estación, línea y red, con una relación jerárquica clara entre ellos. Tenga presente:

- Una estación puede pertenecer a varias líneas (si es una estación de transferencia). Los nombres de las estaciones de transferencia se conforman concatenando el nombre de la estación con el nombre de la línea donde se encuentra.
- Una línea sólo puede pertenecer a una red.
- Una estación sólo puede estar una vez en una línea.
- Una línea sólo puede estar una vez en una red.

Para efectos de este simulador, sólo se construirá una instancia de red Metro.

El modelo de datos a desarrollar para el simulador debe basarse en el paradigma de POO. Adicional a los constructores, destructores, getters, setters y operaciones de despliegue necesarias; incluya subprogramas que permitan:

A. Agregar una estación a una línea, en los extremos o en posiciones intermedias.
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 líneas tiene la red.
I. Saber cuántas estaciones tiene una red Metro (precaución con las estaciones de transferencia).

Por último se necesita un subprograma que permita hacer una sencilla simulación del funcionamiento de la red. Dicha simulación (“Cálculo del tiempo de llegada”), toma la hora actual como tiempo de salida de un tren desde una estación de la red, y se debe predecir el tiempo que tardaría ese tren específico en llegar a otra estación dentro de la misma línea. Este subprograma solo realiza esta tarea entre estaciones de la misma línea.

Especificaciones

El cliente requiere que ud desarrolle un prototipo de simulador de Red de Metro. El simulador debe permitir representar la estructura de una red metro cualquiera, así como ejecutar las operaciones básicas especificadas en este enunciado a través de un menú.

Se sugiere altamente delimitar el desarrollo a lo estrictamente requerido ya que la dimensión real del problema puede tener una complejidad muy superior al tiempo estipulado para la entrega. En caso de duda sobre los requerimientos, consulte con el cliente.

De acuerdo con lo anterior, entre otras cosas, usted deberá:

1. **[10%]** Contextualice el problema, analícelo y finalmente diseñe el diagrama de clases correspondiente a su solución. Refleje adecuadamente las relaciones implícitas en la problemática. Recuerde utilizar la notación UML simplificada impartida en las clases teóricas. A pesar de tener una baja ponderación, la entrega del diagrama de clases es obligatoria y sin ella no se dará lugar la sustentación.
2. **[10%]** Seleccione debidamente los tipos y estructuras de datos que le permitirán implementar las respectivas clases planteadas en 1.
3. **[10%]** Previo a la implementación, verifique el cumplimiento del requisito de eficiencia especificado en la sección 3 del apartado “Requisitos mínimos” de este documento.
4. **[10%]** Implementar un subprograma que permita ejecutar la funcionalidad “Cálculo del tiempo de llegada”.
5. **[60%]** Presente la implementación del simulador, cuya interacción se centra en un menú que permita acceder de manera independiente a las funcionalidades correspondientes a cada clase. Considere las prelaciones implícitas en la lógica del problema. Por ejemplo, una estación no se puede crear en el aire, sólo puede crearse si está adscrita a una línea existente en la cual dicha estación no exista ya.

Requisitos mínimos

A continuación, se describen los requisitos que se deben cumplir. El incumplimiento de cualquiera de ellos implica que su nota sea cero.

1. Genere un informe en donde se detalle el desarrollo del proyecto, explique entre otras cosas:

a. Análisis del problema y consideraciones para la alternativa de solución propuesta.
b. Diagrama de clases de la solución planteada. Adicionalmente, describa en alto nivel la lógica de las tareas que usted definió para aquellos subprogramas cuya solución no sea trivial.
c. Algoritmos implementados debidamente intra-documentados.
d. Problemas de desarrollo que afrontó.
e. Evolución de la solución y consideraciones para tener en cuenta en la implementación.

2. La solución debe ser implementada en lenguaje C++ y debe basarse en el paradigma de POO.
3. La implementación debe considerar el criterio de eficiencia y el buen diseño tanto de las estructuras de datos como de los módulos implementados. Considere además en este apartado la utilización de referencias vs la realización de copias innecesarias sobre objetos estructurados.
4. Se debe crear un repositorio público en el cual se van a poder cargar todos los archivos relacionados a la solución planteada por usted (informe, código fuente y otros anexos).
5. Una vez cumplida la fecha de entrega no se podrá hacer modificación alguna al repositorio.
6. Se deben hacer *commits* de forma regular (al menos uno al día) de tal forma que se evidencie la evolución de la propuesta de solución y su implementación.
7. Se debe adjuntar un enlace de *youtube* a un video que debe incluir lo siguiente:
    a. Presentación de la solución planteada. Análisis realizado y explicación de la arquitectura del sistema (3 minutos máximo).
    b. Demostración de funcionamiento del sistema. Explicar cómo funciona: ejemplos demostrativos (3 minutos máximo).
    c. Explicación del código fuente. Tenga en cuenta que debe justificar la elección de las variables y estructuras de control usados. Por qué eligió uno u otro tipo de variable o estructura de control en cada caso particular y que ventaja ofrecen estos en comparación de otras que también podrían haber sido usados (5 minutos máximo).
    d. La duración total del video no debe exceder 11 minutos ni ser inferior a 5 minutos.
    e. Asegúrese que el video tenga buen sonido y que se puede visualizar con resolución suficiente para apreciar bien los componentes presentados.
8. El plazo de entrega se divide en dos momentos:
    a. El día 2 de mayo para adjuntar la evidencia del proceso de análisis y diseño de la solución.

b. El día 8 de mayo para adjuntar la evidencia del proceso de implementación.

9. Se deben adjuntar **dos enlaces**: uno al repositorio y otro al video, nada más.
10. Para la evaluación del desafío se realizará una sustentación oral en un horario concertado con el profesor. La asistencia a la sustentación es obligatoria.