|  |  |  |
| --- | --- | --- |
| C:\Users\afreijo\Documents\uad.jpg | **UNIVERSIDAD ARGENTINA DE LA EMPRESA** | |
| **Departamento de Tecnología**  **Informática**  PARADIGMA ORIENTADO A OBJETOS (3.4.208)  Profesor: Ignacio Colombo | |
| Examen Final  ALUMNO: LU: FECHA: 08/05/2024  CARRERA: TEMA: 2 | | |
| NOTA: EL EXAMEN ESCRITO ES UN DOCUMENTO DE GRAN IMPORTANCIA PARA LA EVALUACIÓN DE LOS CONOCIMIENTOS ADQUIRIDOS, POR LO TANTO, SE SOLICITA LEER ATENTAMENTE LO SIGUIENTE:   1. El parcial se entregará subiendo el projecto java comprimido en formato ZIP en la tarea asignada en Teams.    * Proyecto exportado con todos los archivos codificados, los diagramas de secuencia y clases en imagen (archivos legibles). 2. Los diagramas a entregar son en formato PNG. En adición, se debe entregar el proyecto (si se desarrolló con startUML) o el código en un archivo .txt (si se desarrolló en PlantUML). 3. El criterio de evaluación es del 50% del examen correcto, esto incluye desde la claridad de los nombres de las clases, métodos y atributos, su aplicación al estándar UML (asociaciones, cardinalidad, etc.) y aplicación de los conceptos y buenas prácticas de la materia. **Revisar la rúbrica de evaluación adjunta al examen.**    * Para llegar al 50%, debe tener al menos:      1. Bien el diagrama de clases,      2. Bien en al menos uno de los diagramas de secuencia, **y**      3. Bien la implementación del código 4. El tiempo estipulado de desarrollo del examen es de 3 (tres) horas.   Por último, les recuerdo que es una instancia de evaluación individual, así que el intercambio de información entre los alumnos se considerará copia con las sanciones del caso si correspondiere. | |  |

## Descripción de la situación

La Ciudad de Buenos Aires desea implementar un sistema de gestión de la línea de subtes.

## Detalles del negocio

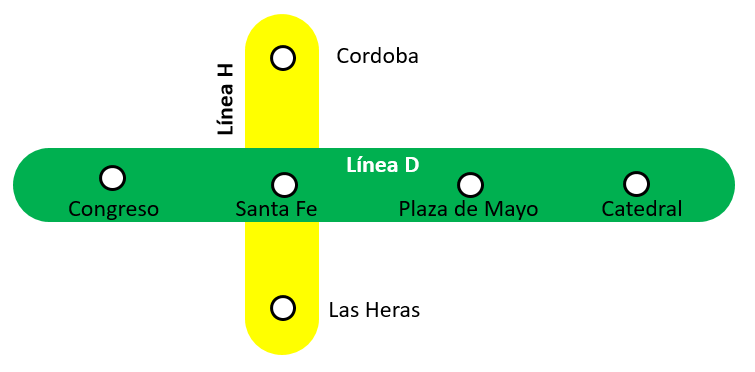
* Los subtes tienen vagones que pueden ser de distinto origen y según este depende su capacidad:

|  |  |
| --- | --- |
| **Origen** | **Capacidad** |
| Vagón Alemán | 30 Pasajeros |
| Vagón Argentino | 25 Pasajeros |
| Vagón Chino | 20 Pasajeros |

* Una línea de subte puede tener vagones de cualquier origen, pero no puede mezclarlos, su capacidad de transporte viene dada por la suma de las capacidades individuales de los vagones.
* La velocidad de un vagón está dada por la fórmula 120/capacidad del vagon
* Las estaciones de subte también tienen una capacidad y por cuestiones de seguridad la capacidad total de la línea que transitan por una estación no puede ser superior. El sistema no puede permitir agregar vagones a una línea que excedan la capacidad de alguna de las estaciones por las que ya está transitando.

|  |  |
| --- | --- |
| **Origen** | **Capacidad** |
| Estación Catedral | 90 Pasajeros |
| Estación Santa Fé | 100 Pasajeros |
| Estación Plaza de Mayo | 120 Pasajeros |
| Estación Congreso | 110 Pasajeros |
| Estación Las Heras | 100 Pasajeros |
| Estación Cordoba | 80 Pasajeros |

* El Sistema de Subtes de la Ciudad es el siguiente (considerar que en la estación Santa Fe se puede hacer combinación de la línea D con la H).



## Puntos Para Desarrollar

1. Desarrollar el Diagrama de Clases
2. Desarrollar los diagramas de Secuencia de los casos de uso:
   1. agregado de un vagón Argentino determinado a una línea del Sistema de Subtes de la Ciudad, el usuario llama al método **agregarVagonArgentino(String nombreLinea)**
   2. agregar una estación a una línea de subte, que comienza con el método: **agregarEstacionLinea(String nombreEstacion,String nombreLinea)**
3. Implementar una interfaz gráfica que solicite el nombre de una Línea de subte y que muestre por pantalla:
   1. Nombre de la línea
   2. Capacidad total (suma de las capacidades de todos los vagones de la Línea)
      1. Implementar método en LineaSubte **public int getCapacidadTotal()** el cual retorna la suma de las capacidades de todos los vagones de la linea.
   3. Nombre de todas las estaciones que componen la línea

## Rúbrica de Evaluación

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **NIVELES DE LOGRO** | | | |
| **MUY BIEN (2,5 puntos)** | **BIEN (1,5 puntos)** | **REGULAR (0,5 puntos)** | **NO SATISFACTORIO (0 puntos)** |
| **Diagrama de Clases** | Diagrama completo incluyendo todas las clases y relaciones con sus tipos (agregación, composición). Todos los tipos de relaciones son correctos. Se definen correctamente los atributos con su accesibilidad y tipo. Los métodos contienen la accesibilidad, parámetros y tipo de retorno. | Diagrama incluye todas las clases y relaciones. Pocos tipos de relaciones son omitidos o incorrectos. Se definen correctamente atributos (con accesibilidad y tipo). Se definen pocos métodos pero contienen accesibilidad, parámetros y tipo de retorno. | Diagrama incluye todas las clases. Pocos tipos de relaciones son omitidos o incorrectos. Se omiten algunos atributos o métodos claves. Algunos métodos o atributos no contienen la accesibilidad o tipo correcto. | Diagrama omite clases y/o relaciones. No presenta ningún tipo de relación o en su mayoría son incorrectos. Omite atributos y métodos claves. Define incorrectamente accesibilidad, tipos y parámetros. |
| **Diagrama de Secuencia (x2)** | Diagrama completo y con un buen nivel de detalle de los métodos. Los mensajes cumplen con el estándar (nombre, tipo de retorno, parámetros). Se respeta la secuencialidad del diagrama. Los elementos del diagrama (loop, alt, opt, etc.) están bien ubicados y siguen el estándar UML. | Diagrama completo pero omite detalle de algunos métodos. Los mensajes cumplen con el estándar. Se respeta la secuencialidad. Algunos elementos del diagrama pueden no estar bien ubicados. | Diagrama no está completo en su totalidad para resolver la funcionalidad u omite detalle de varios métodos. Las clases utilizadas son correctas y la secuencialidad se respeta. Algunos mensajes no respetan el estándar o los retornos son omitidos. Algunos elementos del diagrama pueden no estar bien ubicados. | Diagrama incompleto o no detalla ningún método. No respeta la secuencialidad. No respeta el estándar UML del diagrama. Tipos de retorno son omitidos o no utilizan la flecha correspondiente. Varios elementos del diagrama son omitidos o están mal ubicados. |
| **Código** | Codigo cumple con los patrones de diseño MCV y las buenas practicas. Se utiliza ObjectView para enviar datos a la pantalla. Codigo resuelve todo lo solicitado. | Codigo cumple con los patrones de diseño MCV y las buenas practicas. Se utiliza ObjectView para enviar datos a la pantalla. Alguno de los requerimientos del código es resuelto. | Codigo cumple con los patrones de diseño MCV y las buenas practicas. Se utiliza ObjectView para enviar datos a la pantalla. Más de un requerimiento no se cumple. | No respetan las buenas prácticas de MVC. |