Tecnología de la Programación - Curso 2017/2018

Examen de Junio (06/06/2018) - Duración: 3 horas

Puntuación máxima del examen: 10 puntos

Puntuación mínima en el examen para poder aprobar la asignatura: 4 puntos

Grados en Ingeniería Informática, del Software y de Computadores (Grupos B, C, D)

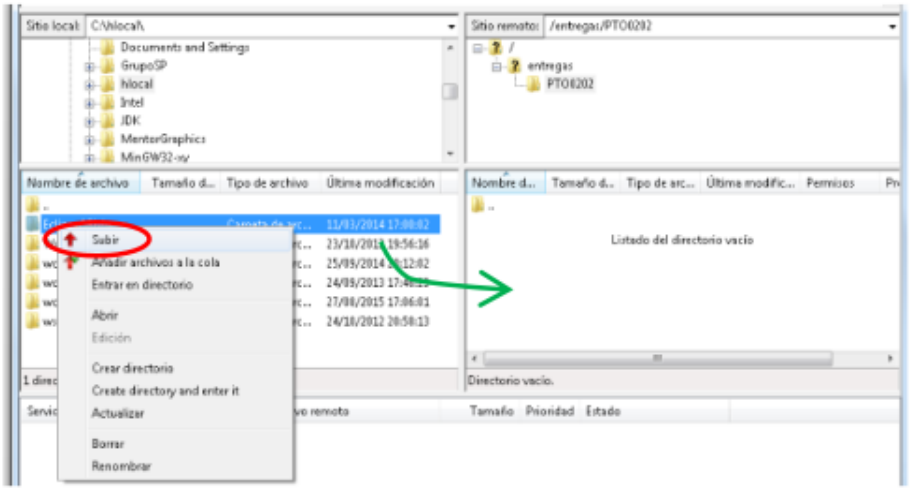
Instrucciones

- Descarga tu Práctica 5 ó 6 del Campus Virtual, ponla a funcionar en Eclipse y comprueba que funciona correctamente.
- **Crea un fichero de texto `cambiosExamen.txt` en la raíz de tu proyecto (dentro de `src`).** En este fichero deberás incluir tu **nombre completo** y el nombre de todos los ficheros `.java` que modifiques. Además puedes incluir otros comentarios relevantes sobre tu solución del examen. Por ejemplo, si sabes cómo hacer algo, pero no sabes cómo escribirlo, describe en este fichero lo que querías hacer (en lugar de entregar código que no funciona; o peor, ni siquiera compila).
- El código entregado ***debe compilar***, y no debe ***romper la encapsulación de las clases*** (acceso a atributos privados y protegidos desde clases externas, utilización de atributos públicos, etc.). Si el código no compila o rompe encapsulación de clases, tendrá la ***calificación de 0 puntos*** en la pregunta correspondiente.
- Para implementar cada una de las preguntas, tienes que cumplir las restricciones que exige la pregunta, y que aparecen en negrita al final de la misma. El incumplimiento de estas restricciones supondrá la ***calificación de 0 puntos*** en dicha pregunta.
- En la corrección del examen se valorará el funcionamiento, la claridad del código, el uso conveniente de los medios proporcionados por la Programación Orientada a Objetos (herencia, polimorfismo y vinculación dinámica), **la elección apropiada de colecciones** y el buen uso de comentarios.

Instrucciones de entrega

- Para entregar la solución al examen, crea un fichero `NombreApellido1Apellido2.zip`. En él debes incluir todo el proyecto una vez limpiado de archivos intermedios (es decir, **sin** el directorio `bin/`, que contiene los `.class`)
- Haz doble clic en el icono del escritorio denominado: **"EXAMENES en LABs entregas..."**, y dentro de la ventana que aparece, doble clic en **"ALUMNOS entrega de practicas y examenes"**.

  Se abre otra ventana en la que debes seleccionar el archivo zip en el panel inferior izquierdo y arrastrarlo al panel inferior derecho (o utiliza el botón derecho del ratón y la opción **Subir**), como se indica en la figura.



- Antes de abandonar el laboratorio debes pasar por el puesto del profesor para asegurarte de que lo que se ve en el puesto del profesor es lo que has entregado (*no cierres tu sesión hasta haberlo verificado*) y **firmar en la hoja de entregas**.

Enunciado

1. **[3,5 puntos]** Implementa un nuevo vehículo, *vehículo electrico*, con el siguiente comportamiento:

   - Tiene una batería con capacidad de C Kwh con $C > 0$ (recibida como entrada).
   - Inicialmente, el nivel de su de batería es C Kwh, es decir, totalmente cargado.
   - Cada kilómetro que recorre consume 1 Kwh de su batería, y cuando el nivel de su batería disminuye a 0 entra en un estado de *recarga* durante T unidades de tiempo (siendo T un valor aleatorio entre 1 y $\frac{C}{10} + 1$).
   - Cuando está en estado de recarga, en cada unidad de tiempo la batería aumenta el nivel en 10 Kwh, pero no puede exceder la capacidad C. El vehículo debe salir del estado de recarga si su nivel de batería llega a C antes de consumir todas las unidades de recarga.
   - Cuando está en estado de recarga, el vehículo no puede avanzar, y su velocidad debería ser 0 (pero, para simplificar, sí puede cruzar un cruce, si el cruce se lo pide).
   - Cuando está en estado de recarga, no puede estar defectuoso. Es decir, cuando se recarga, cualquier llamada a makeFaulty no debe alterar el estado actual del vehículo.
   - Su velocidad no puede exceder el nivel actual de la batería.

   Los eventos asociados a ese tipo de coche incluyen una sección `type` con el valor `electric` y una sección `capacity` con el valor C (la capacidad). Los informes incluyen una sección `type` con el valor `electric`, una sección `battery` con el nivel actual de la batería y una sección `charging` con el tiempo que queda para finalizar la recarga. Por ejemplo,

   ```
   [new_vehicle]
   time = 0
   id = v1
   itinerary = j1,j2,j3
   max_speed = 20
   type = electric
   capacity = 100
   ```

   ```
   [vehicle_report]
   id = v1
   time = 20
   speed = 0
   kilometrage = 160
   faulty = 0
   location = (r2,60)
   type = electric
   battery = 30
   charging = 2
   ```

   NOTA: Para implementar este apartado puedes añadir nuevas clases, pero no debes modificar ninguna otra clase de la práctica (excepto aquella(s) donde se registran ó instancian los tipos de eventos).

2. **[3,5 puntos]** Modifica la Práctica 5 para que, en el modo consola (-m batch), *una vez que la simulación termina*, se muestre, para cada vehículo, el tiempo en el que ha llegado a su destino (muestra NO en caso que no haya llegado).

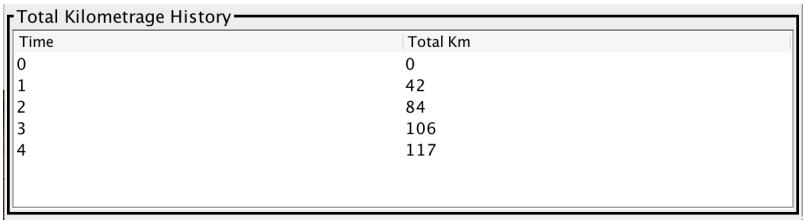
   Así, una salida (que se debe hacer vía `System.out.println`, y *no* usando el `OutputStream` del simulador) podría ser la siguiente:

   ```
   [vehicles_info]
   arrival = (v1,5), (v2,4), (v3, NO), (v4,6)
   ```

NOTA: Para implementar este apartado puedes añadir nuevas clases, pero no debes modificar ninguna otra clase de la práctica (excepto la clase `Main`), salvo la introducción de algún método "get" en caso de que no lo tuvieras implementado en tu práctica. Ten en cuenta además que esta información se muestra cuando la simulación ha terminado, y no en cada paso de la simulación.

En la corrección de la pregunta se valorará fuertemente el diseño y el uso de las colecciones apropiadas.

3. **[3 puntos]** Añade una nueva tabla a la ventana principal de tu práctica, en la que se muestre en cada paso el total kilometros recorridos entre todos los vehículos: cada fila mostrará el paso de simulación y el total de kilómetros recorridos. Un ejemplo de posible tabla sería:



NOTA: Puedes modificar clases relacionadas con la vista, pero no debes modificar ninguna clase del modelo en la implementación de este apartado), salvo la introducción de algún método "get" en caso de que no lo tuvieras implementado en tu práctica.

En la corrección de la pregunta se valorará fuertemente el diseño y el uso de las colecciones apropiadas.