Examen Parcial (Codi: 6 punts)

16 d'Abril de 2021

Contesta els següents exercicis creant el codi Java que es demana a continuació en un projecte Netbeans. Quan acabis, lliura el projecte a la tasca oberta al Campus Virtual amb un sol fitxer comprimit anomenat: Cognom1Cognom2Nom.zip.

Enunciat:

El projecte consisteix en un programa per fer la gestió de vaixells turístics a la Costa Brava. Es podran gestionar diferents tipus de vaixells i crear recorreguts amb una llista de vaixells disponibles per ells.

Cada vaixell (**Vaixell**) tindrà un nom (String), un número de seients (int), un tamany (float) i la informació de si està en manteniment o no (boolean). Inicialment els vaixells no estan en manteniment. Quan ho estiguin no es podran fer servir. Hi ha dos tipus de vaixells: La llanxa (**Llanxa**) és el vaixell més senzill i que només té les propietats d'un vaixell. El número de places disponibles per als passatgers es calcula com el número total de seients menys el número de membres de la tripulació (1). El iot (**Iot**) a més de les propietats d'un vaixell, té un número de cabines amb llits (int) per allotjar als passatgers. El número de places disponibles per als passatgers es calcula com el número total de seients menys el número de membres de la tripulació (5) més el número de llits de les cabines (cada cabina té 4 llits).

Un recorregut (**Recorregut**) tindrà un nom (String) i una llista dels vaixells (**ArrayList**). Per exemple, es podrà crear un recorregut amb nom "Illes medes" i amb tres vaixells disponibles. Llavors, es podrà calcular el nombre de places totals disponibles per a cada recorregut així com el número de llits de cabina.

Exercicis:

Seguint les indicacions de l'enunciat i les que segueixen a continuació, implementa en llenguatge Java les següents classes: **Vaixell**, **Llanxa**, **Iot**, **Recorregut i GestorVaixells**, i la interfície: **InVaixell**.

1. (0.5 punt)

A la interfície **InVaixell** es defineixen els següents mètodes:

- *public int totalPlacesPassatgers();* Permetrà calcular el nombre de places d'un vaixell.
- *public int tripulacio();* Retornarà el nombre de seients reservats per a la tripulació d'un vaixell

2. (1 punt)

La classe **Vaixell** és una classe abstracta que implementa la interfície **InVaixell.** A més, en aquesta classe s'ha de:

- Guardar els atributs que s'han descrit al segon paràgraf de l'enunciat.
- Implementar un sol constructor on s'inicialitzen tots els atributs de la classe amb els valors passats per paràmetre i un valor per defecte false per al boolean.
- Implementar els getters i setters per manipular els atributs.

3. (2 punt)

La classe **Llanxa** hereta de **Vaixell.** A més, en aquesta classe s'ha de:

- Implementar un sol constructor que rebrà els valors de tots els atributs de la classe com a paràmetres.
- Implementar els mètodes *totalPlacesPassatgers* i *tripulació*.

La classe **Iot** hereta de **Vaixell.** A més, en aquesta classe s'ha de:

- Guardar l'atribut que s'ha descrit al segon paràgraf de l'enunciat.
- Implementar un sol constructor que rebrà els valors de tots els atributs de la classe com a paràmetres.
- Implementar els getters i setters per manipular l'atribut propi de la classe.
- Implementar els mètodes *totalPlacesPassatgers* i *tripulació*.
- Implementar el mètode *getLlits* que calcularà i retornarà el número de llits del iot.

4. (1.5 punt)

La classe **Recorregut** guardarà el seu nom i una llista de vaixells (**ArrayList** de Java). I a més, s'han de definir els següents mètodes:

- Un mètode constructor que inicialitzarà els atributs.
- *afegirVaixell:* que permetrà afegir un vaixell a la llista.
- *placesDisponibles:* que calcularà i retornarà el número de places disponibles. Pel càlcul iterarà pels vaixells de la llista i accedirà al càlcul del número de places de cada vaixell que no està en manteniment per sumar-los.
- *llitsCabina*: que calcularà i retornarà el número de llits de cabina disponibles. Pel càlcul iterarà pels iots de la llista i accedirà al mètode que retorna el número de llits (mètode *getLlits* de la classe **Iot**) de cadascun dels iots que no estan en manteniment per sumar-los.

Nota: Per iterar sobre les llistes de tipus **ArrayList** heu de fer servir Iterators. Les solucions que no utilitzen un Iterator no seran avaluades.

5. (1 punt)

En la classe **GestorVaixells** s'ha d'implementar un mètode main on es realitzaran els següents passos:

1. Es crea un recorregut amb nom "Illes Medes".
2. Es crea una llanxa amb nom "Llanxa" de 5 places i 2 metres.
3. S'invoca al mètode del recorregut per afegir la llanxa.
4. Es crea una llanxa amb nom "Popeye" de 5 places i 2 metres.
5. S'invoca al mètode del recorregut per afegir la llanxa.
6. S'invoca al mètode per indicar que aquesta llanxa està en manteniment.
7. Es crea un iot amb nom "Luxe" de 20 places, 10 metres i una cabina.
8. S'invoca al mètode del recorregut per afegir el iot.
9. S'invoca al mètode del recorregut per calcular el número de places disponibles
10. Es mostra per pantalla aquest valor.
11. S'invoca al mètode del recorregut per calcular el número de llits de cabina disponibles.
12. Es mostra per pantalla aquest valor.