EDA (ETS d'Enginyeria Informàtica). Curs 2020-2021

Pràctica 1. La Llista Amb Punt d'Interés d'una aplicació d'apostes de la Primitiva: implementació i ús (1 sessió)

Departament de Sistemes Informàtics i Computació. Universitat Politècnica de València

1. Objetius

El principal objectiu d'aquesta pràctica és que l'alumne aplique al disseny d'una aplicació concreta els conceptes Java per a Estructures de Dades (EDAs) estudiats en el Tema 1 de l'assignatura. Específicament, en acabar aquesta pràctica l'alumne haurà de ser capaç d'implementar i utilitzar eficaçment la jerarquia Java de la *Llista Amb Punt d'Interés* usada en una aplicació d'apostes de la Primitiva.

Al mateix temps, en realitzar aquesta pràctica, l'alumne haurà de ser capaç de crear i manejar l'estructura bàsica de llibreries d'usuari *BlueJ* en la qual, de manera incremental, anirà situant les diferents classes Java que es desenvolupen durant el curs.

2. Descripció del problema

La Loteria Primitiva és un joc d'atzar regulat per Loteries i Apostes de l'Estat en el qual cada aposta consisteix a triar sis números diferents entre l'1 i el 49 (combinació); bàsicament, una aposta resultarà premiada si coincideix amb la combinació guanyadora del sorteig corresponent.

En l'actualitat existeixen diferents aplicacions per a jugar *online* a la Primitiva encara que, òbviament, totes elles comparteixen una mateixa funcionalitat bàsica: fer una aposta. Precisament per això, en aquesta pràctica l'alumne abordarà la implementació d'una senzilla aplicació d'apostes de la Primitiva que tan sols permet realitzar una aposta aleatòria, i.e. seleccionar de manera aleatòria sis números diferents entre l'1 i el 49, i emmagatzemar-los en una Llista amb Punt d'Interés a manera de resguard virtual; a petició de l'usuari, els números de l'aposta poden emmagatzemar-se, bé en ordre de generació, bé en ordre Ascendent.

Les classes de la aplicació

Seguint les anteriors indicacions, les dues classes de l'aplicació a implementar són:

- `NumeroPrimitiva`, que representa un número seleccionat aleatoriament entre 'l'1 i el 49. Per això, un `NumeroPrimitva TÉ UN int numero`.

  Quant als mètodes d'aquesta classe, cal assenyalar ara que ...

  - Com un `NumeroPrimitiva` pot figurar en una llista ordenada Ascendentement, la classe ha de sobrescriure el mètode `compareTo` de la interficie `Comparable` de la següent manera: retorna un `int` negatiu si `this NumeroPrimitiva` té un valor menor que el de l'`altre`, un `int` positiu si `this` té un valor major que el de l'`altre` i `0` si els valors de `this` i `altre` coincideixen.
  - Com els números d'una combinació de la Primitiva han de ser diferents per definició, la classe ha de sobreescriure el mètode `equals` de la classe `Object` com segueix: retorna `true` si `this NumeroPrimitiva` té el mateix valor que `un altre`, o `false` en cas contrari.

- `ApuestaPrimitiva`, que representa una aposta aleatòria de la Primitiva; per això, una `ApuestaPrimitiva TÉ UNA ListaConPI combinacio` de sis `NumeroPrimitiva`, diferents i generats aleatòriament.

  És important destacar ara que, el mètode constructor d'aquesta classe té un paràmetre `boolean ordenada` per a determinar si els números de la `ApuestaPrimitiva` a crear han de ser emmagatzemats en ordre Ascendent (`ordenada == true`) o no (`ordenada == false`); per a aconseguir-ho, la Llista Amb Punt d'Interés que els emmagatzema s'implementarà mitjançant una `LEGListaConPIOrdenada` o una `LEGListaConPI`, respectivament.

  Així mateix, atés que els números d'una aposta han de ser diferents, cada vegada que es genera aleatòriament un nou número de l'aposta s'ha de comprovar que encara no figura en la combinació actual. Per

a això, s'ha d'invocar a un mètode auxiliar de la classe denominat `posicionDe`; en concret, i assumint que el primer element d'una combinació està en la seua posició 0 i l'últim en la 5, aquest mètode retorna la posició d'un `NumeroPrimitiva n` donat en una `ApuestaPrimitiva`, o `-1` si `n` no forma part de la combinació actual.

3. Activitats a realitzar

En tractar-se de la primera sessió de pràctiques, abans de dur a terme les activitats que es proposen en aquest apartat és necessari que l'alumne organitze el seu espai de treball tal com es mostra en la Figura 1.

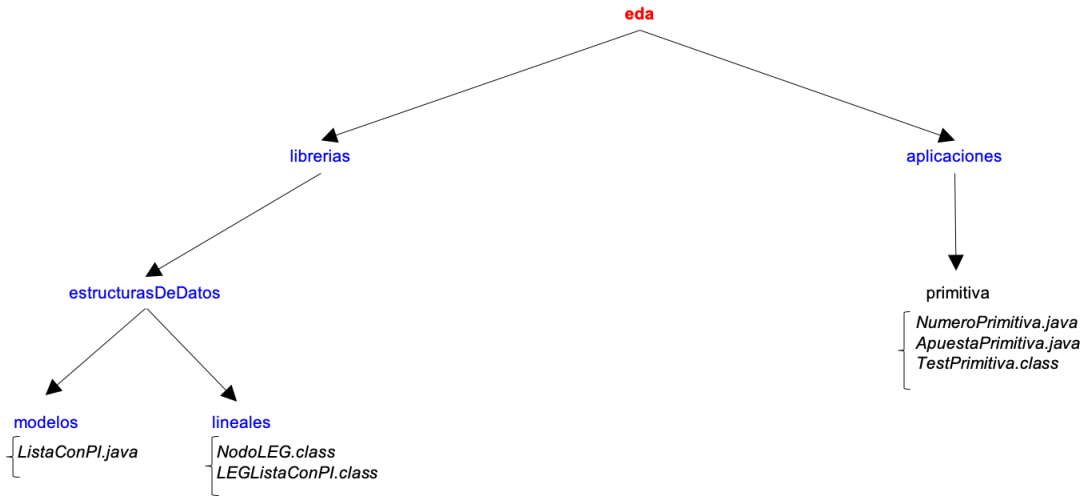

Figura 1: Estructura de l'espai de traball de l'alumne

Per a això, ha de dur a terme els següents passos, en l'ordre al qual s'indiquen:

- Crear un projecte *BlueJ* denominat *eda* en una carpeta del teu ordinador (disc local).
- Dins del projecte *eda*, crear dos nous paquets denominats *librerias* i *aplicaciones*. D'ara en avant, en *librerias* se situaran els paquets que contenen les classes d'utilitats i estructures de dades que s'usaran en les pràctiques de l'assignatura, mentre que en *aplicaciones* estaran els paquets amb les classes de les aplicacions específiques que les usen.
- Obrir el paquet *aplicaciones* i crear en ell un nou paquet de nom *primitiva*, que contindrà les classes de l'aplicació a desenvolupar en aquesta pràctica.
- Obrir el paquet *librerias* i crear en ell un nou paquet de nom *estructurasDeDatos*.
- Obrir el paquet *estructurasDeDatos* i crear en ell dos nous paquets: *modelos* i *lineales*. D'ara en avant, en aquests paquets se situaran, respectivament, els models Java (interfícies) i les Implementacions Lineals que es realitzen d'ells en les pràctiques de l'assignatura.
- Eixir del *BlueJ* **seleccionant la opció *Eixir* de la pestanya *Proyecte***.
- Descarregar els fitxers disponibles en *PoliformaT* en els seus corresponents directoris, tal com s'indica en la Figura 1: `ListaConPI.java` en el directori *librerias.estructurasDeDatos.modelos*; `NodoLEG.java` i `LEGListaConPI.java` en *librerias.estructurasDeDatos.lineales*; etc.
- Entrar en el projecte *BlueJ eda* i compilar la classe del seu paquet *librerias.estructurasDeDatos.modelos*.
- Tancar el paquet *modelos* **seleccionant la opció *Tancar* de la pestanya *Projecte***.
- Obrir el paquet *librerias.estructurasDeDatos.lineales* i compilar les classes `NodoLEG` i `LEGListaConPI`.

3.1. Implementar la classe `LEGListaConPIOrdenada`

En aquesta activitat l'alumne ha de dissenyar en el subpaquete *lineales* la classe `LEGListaConPIOrdenada`, una implementació de la interfície `ListaConPI` que manté ordenats ascendentment els elements d'una *Llista Amb Punt d'Interés*. Per a això, n'hi ha prou amb sobreescriure el mètode `inserir` fent ús, via herència, dels mètodes de `ListaConPI` implementats en la classe `LEGListaConPI`; d'aquesta forma, cada element s'inserirà en la seua posició ordenada en la Llista i no abans del seu Punt d'Interés. A més, atés que els elements d'una `LEGListaConPIOrdenada` són de tipus `Comparable`, en la sobrescritura del mètode `inserir` s'ha d'usar el mètode de comparació genèric `compareTo` de la interfície `Comparable`.

Dos q uestions basiques a tindre en compte a l'hora d'implementar la classe LEGListaConPIOrdenada :

- Els elements d'una llista sols es poden ordenar si són comparables. Pel qué, el tipus genèric dels elements de la classe s'han de restringir a aquells que implementen la interficie `Comparable` de tal manera que es puga usar el mètode `compareTo`.
- La sobrescriptura del mètode `insertar` heretat de la classe `LEGListaConPI` s'ha de realitzar única i exclusivament amb els mètodes definits a l'interficie `ListaConPI`.

3.2. Completar la implementació de les classes de la aplicació d'apostes de La Primitiva

Per a dur a terme aquesta activitat, en primer lloc l'alumne ha d'obrir el paquet *aplicaciones.primitiva* del projecte *BlueJ eda*. Després, seguint els comentaris que apareixen en el codi de les classes d'aquest paquet, així com la descripció que d'elles s'ha realitzat en l'apartat 2 d'aquest butlletí, ha de...

- Completar la capçalera de la classe `NumeroPrimitiva` i els cossos dels seus mètodes `equals` i `compareTo`.
- Completar els cossos dels mètodes de la classe `ApuestaPrimitiva`: constructor amb paràmetre `boolean ordenada`, `posicionDe`, que usa el constructor, i `toString`.

3.3. Validar el codi desenvolupat en la pràctica

Per a comprovar la correcció del codi que ha implementat durant la sessió, l'alumne ha de realitzar dues proves usant les classes del paquet *aplicaciones.primitiva* del seu projecte *BlueJ eda*.

En la primera d'elles, ha de crear dos objectes de tipus `ApuestaPrimitiva` en l' *Object Bench* de *BlueJ*, `aposta` i `apuestaOrd`, executant el constructor de la classe amb arguments `false` i `true` respectivament. Després, usant el *Code Pad* de *BlueJ*, ha d'aplicar a cadascun d'ells el mètode `toString` per a comprovar que, tal com mostra la Figura 2, si bé tots dos objectes contenen sis números diferents entre l'1 i el 49, els de `apuestaOrd` estan ordenats Ascendentment -perquè ha sigut creat usant una `LEGListaConPIOrdenada`- i els de `aposta` no -perquè ha sigut creat usant una `LEGListaConPI`.

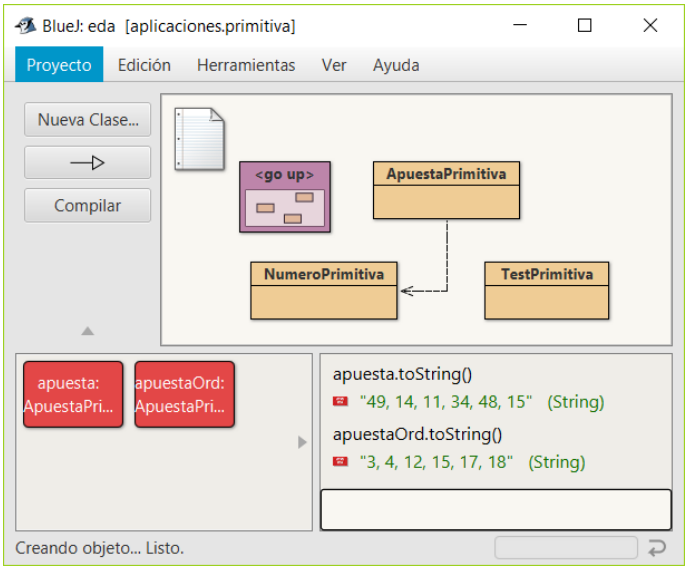

Figura 2: Generació de dos apuestes de La Primitiva

La segona prova que ha de realitzar l'alumne és executar el mètode `comprobar` de la classe `TestPrimitiva`; en fer-ho, es comprova mitjançant una bateria de tests el correcte funcionament de la classe `LEGListaConPIOrdenada`, els mètodes `equals` i `compareTo` de `NumeroPrimitiva` i, finalment, els mètodes constructor, `posicionDe` i `toString` de `ApuestaPrimitiva`.