Java Util labor

Készítette: Goldschmidt Balázs, BME IIT, 2015.

A feladatok megoldásához felhasználandó osztályok leírásait az alábbi URL-en találja meg:

http://download.oracle.com/javase/8/docs/api/

1 Kollekció alapok

Készítsen **ArrayList** felhasználásával sörnyilvántartó alkalmazást (*BeerRegister*)! Az alkalmazás az előző laborhoz hasonlóan parancsokat vár a standard bementén! Az alkalmazás első két parancsa az **add** és a **list**.

- Az **add** paranccsal új sör (*Beer*: név (*name*), jelleg (*style*), alkoholfok (*strength*)) adható a nyilvántartáshoz, pl:
  - add Guinness stout 4.2
  - add Leffe_bruin brown_ale 6.5
  - add Dr_Eher pilsner 5.2

- A **list**tel a nyilvántartás tartalma íratható ki (tetszőleges sorrendben).

2 Comparator

Bővítse az alkalmazás listázó parancsát (**list**)! A listázás sorrendjét határozza meg a parancs opcionális argumentuma:

name - név
style - jelleg
strength - alkoholfok

A feladat megoldásához készítsen minden rendezésfajtához egy-egy *java.util.Comparator* implementációt (*NameComparator*, *StyleComparator*, *StrengthComparator*). A lista rendezéséhez használja a *java.util.Collections* osztály megfelelő metódusát!

3 Szerializálás

Az alkalmazásnak legyen beolvasó és kiíratható parancsa (**load**, **save**), amelyek szerializáltan írják ki az sörök listáját a parancsok argumentumában megadott fájlba!

4 Keresés és törlés

Az alkalmazáshoz készítsen törlő parancsot (**delete**), ami a neve alapján töröl egy sört (használja a *java.util.Collections.binarySearch* metódust).

5 PQueue osztály

Implementáljon generikus **PQueue<T extends Comparable>** osztályt!
Az implementációban az elemek tárolására használjon **ArrayList**-et, a maximumkiválasztáshoz használjuk a **java.util.Collections** osztályt!
A **PQueue** interfésze az alábbi metódusokat valósítsa meg:

- *konstruktor* - létrehoz egy üres prioritási sort (**P**riority **Queue**-t)
- **void push(T t)** - t elemet a sorba helyezi
- **T pop()** - a sorban található legnagyobb elemet visszaadja és törli a sorból
- **T top()** - a sorban található legnagyobb elemet visszaadja, de nem törli
- **int size()** - visszaadja a sorban tárolt elemek számát
- **void clear()** - törli a sorban levő elemeket

A *pop* és a *top* metódusok dobjanak (általunk definiált) **EmptyQueueException**-t!

Készítsen egy **Test** osztályt, ami a **main** metódusában kipróbálja egy *String*gel parametrizált *PQueue* összes metódusát!

6 For-each

Tegye lehetővé, hogy az előző feladatban elkészített osztályra meg lehessen hívni a Java for-each parancsát. Forduljon le az alábbi kódrészlet, ami kiírja a sor tartalmát nagyság szerint csökkenő sorrendben(4,3,2,1). A feladathoz meg kell valósítani a *java.lang.Iterable* interfészt. Az iterátor sorban halad, először a legnagyobb elemet adja vissza (ha szükséges, használja a *Collections* megfelelő metódusát). A megoldáshoz nem szükséges saját *Iterator* osztályt készíteni!

```
PQueue<Integer> s = new PQueue<Integer>();
s.push(1); s.push(2); s.push(3); s.push(4);
for (Integer i : s) {
    System.out.println(i);
}
```

Az előző feladat **Test** osztályának **main** metódusában próbálja ki a fenti kódrészletet!