Programozási nyelvek (BSc, 18) Java 2019/20/2 elővizsga

Az alábbi feladatleírásban azon nyelvi elemek (osztályok, metódusok stb.) leírása szerepel, amelyeknek kötelező megjelennie a megoldásban.

Megengedett a megnevezetteken kívül további metódusok, adattagok felvétele, ha szükséges; ezek legyenek rejtettek. Ha a feladatleírásban meg van adva egy nyelvi elem neve, kötelező azt használni, különben szabadon választható. Szabad választás esetén is betartandóak a Java nyelv konvenciói. A megadottakon kívül egyetlen osztály se tartalmazzon más publikus metódust vagy adattagot, illetve egyik csomag se tartalmazzon más osztályokat. A program ne írjon ki semmit, amire a feladat nem ad utasítást. Ha bármilyen kérdés, észrevétel felmerül, azt a felügyelőknek kell jelezni, *NEM* a diáktársaknak!

A megoldás értékelése:

Az elméleti és programozási feladatra 20-20 pont szerezhető. A gyakorlati jegy a következőképpen kerül meghatározásra: 20 ponttól 2-es, 25 ponttól 3-as, 30 ponttól 4-es, 35 ponttól 5-ös.

Használható segédanyagok:

- A Canvas-ben, a Fájlok oldalon, pre_test/ könyvtár alatt található base.zip, amely tartalmazza
  - az alapfeladathoz a kiindulási fájlt
  - a JUnit tesztelő `jar` fájljait
- Java dokumentáció
- Legfeljebb egy üres lap és toll.

A feladat összefoglaló leírása

Egy számonkérésre különböző tankönyvek különböző oldalaiból kell felkészülni. Az elkészítendő program feladata ezeknek az oldalaknak a kezelése.

A feladat részletes ismertetése

Alapfeladat (9 pont)

A letölthető `study.planner.StudyPlanner` osztály kódja helytelen, ebben a feladatrészben ezt kell javítani, valamint a leírtaknak megfelelően bővíteni is.

- A javítások csak olyan mértékben módosítsák a kódot, amennyire ez feltétlenül szükséges.
- A fájlban `6` sorban `7` darab hiba van.
  - Az alábbiak elvégzendőek, nem számítanak bele a `7` hibába:
    - Az adattag felvételével járó módosítások.
    - A fájl legelejének megfelelő kitöltése.
    - A fájlt a megfelelő könyvtárba kell helyezni.

Az osztálynak legyen egy félnyilvános `bookToPages` adattagja, amely hozzárendeli egy-egy tankönyv nevéhez (ami egy szöveg) azoknak az oldalaknak a halmazát, amelyeket abból a tankönyvből meg kell tanulni.

Az osztályban egy metódus szerepel, `readPagesFromText`, amelynek a következő az elvárt működése.

- Bemenetként egy `Scanner` paramétert kap, ennek az alábbiakban leírt szöveges tartalma van. A metódus feladata a szöveget kiolvasni és feldolgozni.

    - Például a szöveges bemenet lehet a következő:

      ```
      4
      10 20 Programozasi Nyelvek Java
      150 190 Analizis
      20 130 Analizis
      55 78 Programozasi Nyelvek Java
      ```

    - Az első sor egyetlen számot tartalmaz (`lineCount`), amely megadja, hogy hány sor következik.
    - A többi sor szóközzel elválasztva tartalmazza a könyvekből feladott rész kezdő és záró oldalszámát, és a könyv nevét.
      - Feltehető, hogy a bemenet megfelelő szerkezetű, és a könyvek nevei csak a latin ábécé kis- és nagybetűit tartalmazzák a szóközökön kívül.
      - Különböző sorok jelölhetnek ki részeket ugyanabban a könyvben.
      - A feladott rész zárt intervallum: az első és az utolsó oldal is fel van adva.
    - A példában tehát a `Programozasi Nyelvek Java` könyv `10`, `11`, `...`, `20`, `55`, `56`, `...`, `78` oldalai vannak feladva, az `Analizis` könyvből pedig a `20`, `21`, `...`, `130`, `150`, `151`, `...`, `190` oldalak.
- A `readPagesFromText` metódus kiolvassa `lineCount` értékét, majd feldolgozza a következő `lineCount` sort.
    - Azt a sort, amelyik nem pontosan három részre bomlik a szóközök mentén, figyelmen kívül kell hagyni.
    - Egyébként feltételezhető, hogy minden adat helyes, nem kell további ellenőrzéseket végezni.
- A megnevezett könyvhöz tartozó feladott oldalak halmazába kell betenni a sorban megadott zárt intervallum mindegyik számát.
    - Ha a halmaz még nem létezik, mert még nem szerepelt ilyen könyv, létre kell hozni.
    - Ha a halmaz már tartalmazza az intervallum valamelyik értékét, akkor váltódjon ki egy `StudyException` úgy, ahogy le van írva a kódban.
      - Ez a kivételfajta legyen ugyanebben a csomagban. A konstruktora vegyen át egy magyarázó szöveget, és a bázisosztály szöveges konstruktorát hívja meg.
      - A kivétel kiváltásával a `Scanner` feldolgozása megszakad, ezzel most nem törődünk.
- A `Scanner` objektumot nem kell bezárni.

Készüljenek el továbbá a következő lekérdező metódusok:

- `getBookCount()`: a betöltött könyvek száma
- `pageCountOf("Analizis")`: a megadott nevű könyvből feladott összes oldal száma, a példában szereplő `Analizis` könyvre `152`
    - ha a megadott nevű könyv nem létezik, váltódjon ki `StudyException` ilyen üzenettel: `"Book Analizis is unknown"`

Az osztályhoz készüljön JUnit alapú tesztelő a `study.planner.test.StudyPlannerTest` osztályban. A tesztelő mindegyik esete hívja meg a `readPagesFromText` metódust úgy, hogy előkészít neki egy `Scanner` objektumot; a `Scanner` kapjon szöveges paramétert az alábbi leírások szerint. A tesztelő vizsgálja a következő eseteket (a sor elején a tesztelő metódusok nevei állnak):

- `noLines`: `0` darab könyv szerepel, ha a `Scanner` olyan szöveget kap, amelyben `lineCount` értéke `0`
- `exampleBookCount`: a fenti példára a könyvek száma `2`
- `examplePageCount_Analizis`, `examplePageCount_Java`: a fenti példa könyveinek hosszát ellenőrzik (`152` és `35`)
- `missingBook`: ha a `MissingBook` hosszát kérjük le, akkor `StudyException` váltódik ki

Emlékeztető:

- az `org.junit.Assert` osztályt (és/vagy annak műveleteit, pl. `assertEquals`) kell importálni, valamint a `org.junit.Test` annotációt
- a JUnit futtatása, ha a tesztesetek osztálya a névtelen csomagba tartozó `SimpleTest` (Windows rendszeren `;` az elválasztó):

```
javac -cp .:junit-4.12.jar:hamcrest-core-1.3.jar SimpleTest.java
java -cp .:junit-4.12.jar:hamcrest-core-1.3.jar org.junit.runner.JUnitCore SimpleTest
```

Bővítés: öröklődés (5 pont)

Készüljön el az alábbi két művelet is a `StudyPlanner` osztályba:

- `isStudied("Analizis", 160)`: megadja, hogy az `Analizis` könyvből fel van-e adva a megadott oldal (a példában igen)
- `isStudied("Analizis", 140, 200)`: megadja, hogy az `Analizis` könyvből a megadott (zárt) intervallum bármelyik oldala fel van-e adva
    - A megadott paraméterekkel a válasz `true`, mert pl. a `150`. oldal fel van adva.
    - Például `isStudied("Analizis", 141, 149)` hamisat ad vissza.

A `StudyPlanner` osztály leszármazottja legyen a `study.planner.StrictStudyPlanner` osztály, amely működjön úgy, mint a bázisosztálya, azonban mindegyik oldal legyen feladva a legkisebb és a legnagyobb feladott oldalszám között.

- Az eltárolt adatokon nem kell módosítani, hanem meg kell találni az alsó és a felső határt.
    - Tipp: a legkisebb indexű szakaszt úgy könnyű megtalálni, ha a `Set`-ből egy `TreeSet`-et készítünk, majd arra meghívjuk a `ceiling(Integer.MIN_VALUE)` műveletet; a legnagyobb értékhez `floor(Integer.MAX_VALUE)` használható.
- A megörökölt metódusok (`pageCountOf` és a két `isStudied`) működjenek eszerint.
- Példa: `StrictStudyPlanner` objektumba töltve a fenti példa adatait, `isStudied("Analizis", 141, 149)` igazat ad, mert a `20`. és `190`. oldal között mindegyik fel van adva.

A tesztelő ellenőrizze egy-egy tesztesettel, hogy a `pageCountOf` és a két `isStudied` különbözőképpen működik `StudyPlanner` és `StrictStudyPlanner` használatával.

Bővítés: egyenlőség (4 pont)

A `StudyPlanner` osztályra legyen értelmezett az egyenlőségvizsgálat.

Akkor tekintsünk két példányt egyenlőnek, ha ugyanazokban a könyvekben ugyanazok az oldalak vannak feladva.

- Az egyenlőségvizsgálat mellett azt a másik metódust is meg kell (értelemszerűen) valósítani, ami szerződéses viszonyban áll vele.
- Mindkét művelet implementálása során kihasználható, hogy a könyvtári típusokra definiálták ezeket a műveleteket.

Egy `studyDifferently` teszteset tesztelje, hogy két különböző módon betöltött `StudyPlanner` lehet egyenlő egymással. A fenti példa és az alábbi egyenlőek:

```
~~~
6
67 78 Programozasi Nyelvek Java
10 20 Programozasi Nyelvek Java
150 190 Analizis
20 123 Analizis
55 66 Programozasi Nyelvek Java
124 130 Analizis
~~~
```

Bővítés: összehasonlíthatóság (2 pont)

Két `StudyPlanner` legyen egymással összehasonlítható a következő módon.

Az a `StudyPlanner` nagyobb, amelyben több oldalt kell megtanuni.

Írjon a három esetre egy-egy unit test-et.