Miskolci Független Vizsgaközpont

Java Junior Backend Fejlesztő

szakképesítés

programkövetelmény száma: 06135011

Vizsgafeladat

Vizsga időpontja: 2022.11.07.

Vizsga helyszíne: Budapest

Értékelés

81 - 100%	jeles
71 – 80%	jó
61 - 70%	közepes
51 - 60%	elégséges
0 - 50%	elégtelen

Projekt feladatok megvalósítására vonatkozó információk:

Készítse el a következő konzolos alkalmazásokat, amelyek megoldására 180 perc áll rendelkezésre. A projekt vázát, valamint az automata unit teszteseteket verziókövető rendszeren keresztül éri el a vizsgán megadott linken. A vizsgázó feladata a szöveges leírás megértése, a tesztesetek értelmezése. Ezek alapján meg kell terveznie a megoldást. A tervezés során meg kell határoznia a megoldást biztosító metódusokat, osztályokat és interfészeket. A tervezés után implementálnia kell a megoldást a megfelelő fejlesztőeszközben. Meg kell nyitnia a projekt vázát, melyben a tesztesetek nem fognak lefordulni. A vizsgázónak létre kell hoznia a megtervezett osztályokat és interfészeket, mely után a projekt lefordítható, de funkcionálisan nem működőképes, azaz a unit tesztesetek már lefutnak, de hibát adnak. Ezután úgy kell implementálnia a hiányzó részeket, hogy mind a leírásnak, mind a teszteseteknek megfeleljenek, azaz a tesztesetek lefuttatása sikeres legyen. A vizsgázónak figyelnie kell az objektumorientált programozás alapelveire, valamint a clean code elvekre, azaz olvasható és karbantartható kódot kell írnia. A forráskódot ellenőrizni kell a kódolási konvenciók alapján. Az alkalmazást le kell buildelni. Amennyiben a vizsgázó elkészült a munkájával, azt verziókövető rendszeren kell beadnia.

1. Feladat - Java programozási nyelv alapjai

Mezőgazdasági jóslás.

Az időjárás más és más minden évben. Így a mezőgazdaságban soha nem lehet tudni előre a várható termés mennyiségét. A korábbi évek megfigyelései alapján viszont megjósolható a várható termés. Józsi bácsi kíváncsi ember és nagyon érdekli, hogy különböző időjárási viszonyok alapján mennyi búza teremne neki az elvetett mennyiség alapján. Ezért segítsük őt a következő szimulációs programmal.

1. A feladat megoldásához nyissa meg a „**joslas**” nevű projektet. A könyvtárban lévő „Main.java” forrásfájlban dolgozzon.

2. Írjon egy függvényt „**milyenHozamVarhato**” néven, amely egy véletlenszerűen generált, az időjárási viszonyokat jelző számot (*intervalluma: [5-15]) Az év meghatározása:* 9 alatt-*átlag alatti,* 12 felett-*átlag feletti, a köztes értékek esetén-átlagos év várható*) és az elvetett búza mennyiségét kapja meg bemeneti értékként. Ezek alapján, a két szám szorzataként, visszaadja a várható hozamot.

3. Írjon egy újabb függvényt „**milyenEvVarhato**” néven, amely a várható hozam és az elvetett búza alapján visszaadja, hogy milyen év („átlag alatti”, „átlagos” vagy „átlag feletti” a 2. feladatban megfogalmazottak alapján) várható. A vizsgálatokat a függvény törzsében valósítsa meg a bemeneti paraméterek hányadosa alapján.

4. A program addig kérje be az elvetett búza mennyiségét tonnában a felhasználótól, amíg üres bemenetet nem kap! Ilyen akkor történik, ha a felhasználó egyszerűen Enter-t nyom anélkül, hogy bármit is begépelne. A program felépítését a következő leírás alapján készítse el!

5. A várható hozamot a „***milyenHozamVarhato***” függvény felhasználásával, jelenítse meg a mintán látható formátumban.

6. Írjon egy eljárást „**KiIras**” névvel, amely egy szöveges paraméterben megkapja, hogy milyen év várható. Az eljárás a paraméterben kapott (a „*milyenEvVarhato*” függvény által visszaadott) érték felhasználásával jelenítse meg az üzenetet. A megjelenítést a mintán látható formátumban végezze el.

7. Ellenőrizze a megoldását a „joslasTest” segítségével. Hibátlan megvalósítás esetén 4 darab sikeres teszt fog lefutni.

A program üzeneteinek megfogalmazásában kövesse az alábbi példát! Azokat a részeket, amiket a felhasználó gépel be, a mintában vastagított és döntött betűkkel emeltük ki.

```
Add meg az elvetett búza mennyiségét tonnában! 6
A várható hozam 60 tonna.
A hozam alapján átlagos év várható.
Add meg az elvetett búza mennyiségét tonnában! 8
A várható hozam 104 tonna.
A hozam alapján átlag feletti év várható.
Add meg az elvetett búza mennyiségét tonnában! 7
A várható hozam 42 tonna.
A hozam alapján átlag alatti év várható.
Add meg az elvetett búza mennyiségét tonnában!
C:\Users\vizsga\programok>
```

2. Feladat – Java objektumorientált programozás

Könyvtár

1. A feladat megoldásához nyissa meg a „**konyvtar**” nevű projektet. A benne lévő „Main.java” forrásfájlban írja meg a főprogramot. A további leírások alapján hozza létre a következő osztályokat és interface-t.

2. Hozzon létre egy „**rekord**” nevű, nem példányosítható osztályt. Az osztály egy örököltethető mezőt tartalmaz „**ISBN**” névvel, szöveges típussal. A mező védelmi szintjét a leírás alapján válassza meg! Ezt a mezőt csak az osztály publikus, belső metódusai érhetik csak el (setter és getter). Ezért készítsen hozzá jellemzőket, azaz gettert és settert, „**getISBN**” és „**setISBN**” névvel. A getter adja vissza az ISBN mező értékét, a setter pedig adja át neki a paraméterben kapott értéket abban az esetben, ha a paraméterében 10 vagy 13 hosszúságú, számsort kapott! A feltételnek nem megfelelő ISBN szám esetén a setter „dobjon” egy "***RangeException***” kivételt, az „*Az ISBN értéke 10 vagy 13 hosszú számsor lehet csak!*” üzenettel. Ha nem sikerül kivételt dobnia, akkor is a vizsgálatot végezze el a setter törzsében, de akkor a konzolra írja ki az üzenetet.

3. Hozzon létre egy „**konyv**” nevű osztályt, adja meg ősként a „***rekord***” osztályt. Hozza létre benne a következő 4 nyilvános védelmi szintű mezőt a megadott típussal. „**Cím**” – *szöveg*, „**Szerzo**” - *szöveg*, „**Nyelv**”-*szöveg*, „**MegjelenesiDatum**”- *egész.* Készítsen hozzá konstruktort (5 paraméterrel), amely az osztály minden mezőjét beállítja (a rekord osztály ISBN mezőjét is). Készítse el az osztály „üres” konstruktorát is.

4. Hozzon létre egy Interface-t „**IkonyvekSzama**” névvel, és deklaráljon benne egy függvényt „**ISBNperDarab**” névvel. A függvény integer típusú értékkel tér vissza, bemeneti paramétere pedig egy ISBN szám „**ISBNSzam**” névvel.

5. Hozzon létre egy „**konyvtar**” nevű osztályt, az osztály definíciójánál adja hozzá a korábban létrehozott „***IkonyvekSzama***” nevű interface-t. Az osztályban hozzon létre egy listát, amely „***konyv***” típusú objektumokat tárol „**konyvek**” névvel. A lista csak az osztályon belülről legyen elérhető! Készítsen egy nyilvános eljárást „**konyvHozzaAd**” névvel, amely paraméterben egy „***konyv***” típusú objektumot kap. Az eljárás törzsében lévő utasítással a paraméterben kapott könyvet tárolja el a „***konyvek***” nevű listában. Készítsen egy publikus gettert ”**getKonyvek**” névvel, amely visszaadja a könyvtárban lévő összes könyv adatát („***konyvek***” listát). Továbbá implementálja az interface-ben megadott függvényt. A függvény adja vissza azon könyvek számát (egész) a „***konyvek***” nevű listából, amelyek ISBN száma megegyezik a paraméterben kapott ISBN számmal.

6. Hozzon létre a főprogramban egy „**Konyvtar**” nevű „***konyvtar***” típusú objektumot. A következő két könyv létrehozásához használhatja a 2_OOP.txt-t. Hozzon létre egy „***konyv***” típusú objektumot „**konyv1**” névvel. Adja meg a következő értékeket: ISBN: 9783404142163, Cím: Egri csillagok, Szerzo: Gárdonyi Géza, Nyelv: magyar, MegjelenesiDatum: 1899. Hozzon létre egy másik „konyv” típusú elemet „**konyv2**” névvel a következő adatokkal: ISBN: 9788807900365, Cím: A Pál utcai fiúk , Szerzo: Molnár Ferenc, Nyelv: magyar, MegjelenesiDatum: 1906. A két könyvet helyezze el a könyvtárba („***Konyvtar***”) a könyv hozzáadására szolgáló metódus segítségével. Jelenítse meg a képernyőn, hogy a könyvtárban hány darab „*9783404142163*” számmal lévő könyv van, a következő formában: „*A könyvtárban 1 db 9783404142163-as ISBN számmal rendelkező könyv található.*”

7. Tesztelje a programját a csatolt JUnit tesztekkel!

3. Feladat – Java kollekciók

Napló

Az iskolákban a diákok értkélése a különböző tantárgyakból 1-től 5-ig adott érdemjeggyel történik. Nem történik ez másképp a „Programozási alapok” nevű tárgyból sem. A feladat, hogy az említett tantárgyból a programunk segítségével, véletlenszám generálás felhasználásával létrehozzuk a tantárgy éves naplózási adatait, azaz az osztályba járó diákok milyen jegyeket kaptak havonta a tanév során. A tanév szeptembertől következő év június közepéig tart. Egy diáknak havonta legalább egy jegyének lennie kell, maximum jegy 5 adható neki egy hónapban. Az érdemjegyek a következő eséllyel kerüljenek legenerálásra: 1 – 10%, 2 - 15%, 3 – 20%, 4 – 25%, 5 -30%. Az év végi jegy kerekített jegy, amely az átlagból kerül kiszámításra. 0.5 felett felfelé kerekítünk, ellenkező esetben lefelé.

1. A feladat megoldásához nyissa meg a „**naplo**” nevű projektet. A benne lévő „Main.java” forrásfájlban írja meg a főprogramot.

2. Hozzon létre egy listát, amely egy három elemből álló, egész számokat tartalmazó tömb tárolására szolgál „**Naplo**” névvel. A lista a program minden részéből elérhető/látható legyen, azaz a függvényekből is.

3. A „Main”-ben hozzon létre egy, kulcs-értékpárok tárolására szolgáló adatszerkezetet „**Diakok**” néven a „try” előtti részben. A következő hat diák adatát rögzítse és tárolja el benne a megadott kulcs-értékpárok (szöveg-egész) alapján: Edina-1, Géza-2, Réka-3, Béla-4, Zita-5, Tamás-6.

4. A jegyek generálására hozzon létre egy függvényt „**jegyGenerator**” névvel, amely [1,100] intervallumba eső (főprogramban generált) egész értéket kap bemenetként. A kapott érték alapján, a bevezetőben megfogalmazottak szerint generálja le az érdemjegyet. (pl.: *10*-->1, *11*-->2, *26*-->3, *46*-->4, *71*-->5). (Ha ez nem sikerül véletlenszám generátorral adjon vissza értéket [1,5] között, ha ezt választotta akkor a jegygenerátor tesztek hibásan futhat le!)

5. Készítsen egy függvényt „**diakEvVegiErtekelese**” névvel. A visszatérési értéke valós szám legyen. A függvény bemeneti paramétere pedig egy diák kódja legyen. A függvény a paraméterben kapott diák kódja alapján a „*Naplo*” adatbázisból adja vissza a diák évvégi átlagát. A diák kódja a listában tárolt tömb első eleme. A hozzá tartozó jegy a tömb harmadik eleme.

6. Ciklusok egymásba ágyazásával oldja meg, hogy az összes diák, minden tanítási hónapra, a bevezetőben megfogalmazottak szerinti számú jegyet kapjon. A jegyek generálásához használja a „*jegyGenerator*” függvényt. Egy diák egy jegyének az eltárolásához hozzon létre egy tömb típusú adatszerkezetet, amely 3 egész szám tárolására szolgál. A tömb első eleme a diák azonosítója, a második elem a hónap, a harmadik elem pedig a generált érdemjegy legyen. A létrehozott tömböt tárolja el a „*Naplo*”-ban minden egyes iteráció alkalmával.

7. A főprogram „try” részében kérje be egy diák nevét. „***Add meg egy diák nevét:*** ” üzenettel. A bekért név alapján határozza meg a diák kódját a „*Diakok*” adatbázisból. A kapott kóddal használja „*diakEvVegiErtekelese*” függvényt az év végi átlag meghatározásához. Az átlagot 2 tizedesjegy pontossággal, az év végi jegyet pedig a feladat leírásában megadottak alapján jelenítse meg a következő formában: pl.: „***xy Programozási alapok tantárgy átlaga: x.xx. Év végi jegye: y***”, ahol az „xy” a diák neve, a „x.xx” és az „y” a kapott és számított értékeket jelöli.

8. Tesztelje a programját a csatolt JUnit tesztekkel!

4. Feladat – Fájlkezelés Javaban

Kutyák

Az ebek mikrochippes megjelölése 2013 január 1-től, minden 4 hónaposnál idősebb kutyánál kötelező. A következő szöveges fájlban (4_kutyak.csv) ilyen mikrocsippel ellátott kutyák adatai találhatóak. A fájl első sora fejlécet tartalmaz, a további sorok egy-egy kutya adatát tartalmazzák a következő sorrendben. 1. mező – **ID**-nem folytonosan emelkedő egész, 2. mező – **KutyaNeve**, szöveges tartalom, 3. mező – **Ivar** szöveges (hím vagy nőstény), 4. mező – **ChipSzama** 15-19 számjegyből álló azonosító nemzetiséggel kiegészítve, 5. mező – **Tulajdonos**, szöveges. Ahol a tulajdonos nem ismert, ott a Tulajdonos mező üres. Ezek a kutyák jellemzően örökbefogadhatóak. Az egyes mezők tartalma a „**;**” azaz pontosvessző karakterrel van elválasztva egymástól. Azon feladatoknál, ahol képernyőre írást kér a feladat az adatok megjelenítése előtt írja ki a feladat sorszámát a következő módon: pl.: „3. feladat:”

1. A feladat megoldásához nyissa meg a „**kutyak**” nevű projektet. Itt dolgozzon! A benne lévő „Main.java” forrásfájlban írja meg a főprogramot, illetve egészítse ki a projektet a megadott osztállyal.

2. Hozzon létre egy „**Kutya**” nevű osztályt a feladat leírásában szereplő mezökkel (5 db). Készítsen hozzá konstruktort (5 paraméterrel), amely az osztály minden mezőjének értéket ad. Készítse el az osztály „üres” konstruktorát is.

3. Hozza létre a „gazdatlanKutyakSzama” nevű függvényt, amely a „kutyak” listából visszaadja a „Tulajdonos” nélküli kutyák számát.

4. Olvassa be és tárolja el a „4_kutyak.csv” fájlban lévő adatokat. Az egyes sorokat egy-egy „*Kutya*” típusú objektumban tárolja el, amelyeket egy „**kutyak**” nevű listában helyezzen el a további feladatok végrehajtásához. Figyeljen arra, hogy a fájl első sora fejlécet tartalmaz!

5. A 3. feladatban megírt függvény segítségével írja ki a képernyőre, hogy hány gazdátlan /örökbefogadható kutya adatát olvasta be a következő formátumban: „**Gazdátlan/örökbefogadható kutyák száma: x**”, ahol „x” a függvény által visszaadott értéket jelöli.

6. Az egyedi azonosítók 1-től 407-ig emelkedően kerültek kiosztásra az adatbázisban. Viszont nem minden azonosító került bele az adatbázisba. A „hianyzoIDk.txt” fájlba írja ki azokat az ID-ket egymás alá, amelyek hiányoznak a „4_kutyak.csv” fájlból. A feladat megoldásánál használjon halmazt (HashSet), amelyben a meglévő ID-ket tárolja! A kiírásnál használja fel a feladat megoldása során létrehozott halmazt!

7. Tesztelje a programját a csatolt JUnit teszttel!

Adatbázis forrás: internet (http://www.hovawartclub.hu/hovi/index.php?act=chip)

5. Feladat – Adatbáziskezelés Javaban

Budapesti irányítószámok

A következő feladatban a budapesti irányítószámokat tartalmazó adatbázissal kell dolgozni. Ebből az adatbázisból kell különböző lekéréseket, törléseket és módosításokat elvégezni. Azon feladatoknál, ahol képernyőre írást vagy adat bekérést kér a feladat, az adatok megjelenítése előtt írja ki a feladat sorszámát a következő módon: pl.: „3. feladat:”

1. Importálja be a forrásként csatolt „**5_irszamok.sql**” szöveges fájlt a MySQL adatbázisba. A továbbiakban az importálás során létrejött „**irszamok**” adatbázissal kell dolgoznia.

2. A feladat megoldásához nyissa meg a „**Budapest**” nevű projektet. Itt dolgozzon! A benne lévő „Main.java” forrás fájlban írja meg a főprogramot.

3. A következő utasításokat a „Start” függvényben végezze el! Csatlakoztassa a programhoz az „**irszamok**” („**url**”) adatbázis, továbbá adja meg az adatbázis csatlakoztatáshoz tartozó nevet („**user**”) és jelszót („**password**”), azaz a zárójelben lévő mezőknek adja meg a megfelelő értékeket a függvény törzsében.

4. A csatolt „**rekord**” nevű osztálynak készítse el az osztály üres konstruktorát.

5. A „**lekerdezes**” függvényben az „sqlCommand” nevű szöveges változónak adja meg azt az SQL utasítást amely a „budapest” táblából az összes adatot visszaadja. Az utasítás végére tegyen egy szóközt!

6. Készítsen lekérdezést a „***budapest***” táblából, amely visszaadja az összes olyan rekordot, amelynél a „***varoszresz***” mező nem üres. Ehhez a főprogram „Main” részében található, a „6. feladat:” kiírás utáni „**parameter**” nevű szöveges változónak adja meg azt a kiegészítő SQL feltételt, amely leszűri azokat a rekordokat amelyeknél a „varosresz” mező nem üres. Ezt követően fejezze be a 6-os feladatot. A „varosReszLista” nevű lista *'ker', 'varosresz', 'utcanev'* és *'utotag'* mezőit jelenítse meg soronként tabulátorral elválasztva egymástól. Sikeres megoldás esetén 14 sor jelenik meg.

7. Adja meg azt az SQL utasítást a „**kilencedikKeruletiTerek**” függvényben lévő „**sqlCommand**” szöveges változónak, amely a „***budapest***” táblából visszaadja az összes XI. kerületi tér nevét. . Sikeres utasítás megadása esetén, a képernyőn megjelennek egymás alá kiírva a IX. kerületi terek.

8. Adja meg azt az SQL utasítást a „**torles**” függvényben lévő „**sqlCommand**” szöveges változónak, amely a „***budapest***” táblából törli a paraméterben kapott „**ID**”-vel rendelkező rekordot. Sikeres utasítás megadása esetén, a képernyőn a „Töröltem.” üzenet jelenik meg. Amennyiben nincs ilyen rekord akkor a „Nem történt törlés” üzenet jelenik meg. A törlést az „1”-es ID-vel tesztelje!

9. Adja meg azt a hiányzó SQL parancsot az „**utcanevCsere**” függvényben lévő „**command**” szöveges változónak, amelynek a segítségével a „***budapest***” táblában felülírásra kerülnek azok az utcanevek amelyeket a „***regiNev***” paraméterben adunk meg arra, amit az „***ujNev***” paraméterben adtunk meg. Sikeres utasítás megadása esetén, a képernyőn a „Sikeres csere.” üzenet jelenik meg. Ellenkező esetben a „A csere sikertelen.” üzenet.

10. Adja meg azt a hiányzó SQL parancsot az „**rekordHozzaadasa**” függvényben lévő „**sqlCommand**” szöveges változónak, amelynek a segítségével a „***budapest***” táblába bekerülnek a paraméterben kapott „rekord” típusú objektumban lévő adatok. Sikeres utasítás megadása esetén, a képernyőn az „1 rekord hozzáadása sikeresen megtörtént.” üzenet jelenik meg. Ellenkező esetben a „A rekord hozzáadása sikertelen.” üzenet.

11. Tesztelje a programját a csatolt JUnit tesztekkel!

Adatbázis forrás: internet, Magyar Posta Zrt. honlapjáról letöltve