Java Junior Backend vizsga feladatok

2023-03-06

A feladatokat nagyon figyelmesen olvassa el. Amennyiben valami nem világos a feladat leírásában nyugodtan kérdezzen rá (email: nagy.gabor70@gmail.com). Az alap feladatok, esetén előre megírt metódusoknak kell kitölteni a törzsét. Segéd metódusok bevezetése lehetséges amennyiben szükségét látja (egy metódus csak egy dolgot csinálhat viszont azt nagyon jól). *A megadott metódusok szignatúráját nem szabad megváltoztatni, csak a metódus törzsét kell kitölteni.* A Fundamental osztály elején talál egy testMySolutions nevű metódust. Itt tudja tesztelni az Ön által írt kódot. Ennek a metódusnak a kódja, neve, paraméterei szabadon átírható majd a projekt main metódusából meghívható (main csomag Main osztályban találja). Természetesen a main metódusból közvetlen is meghívhatja a tesztelendő metódusát 😊. Amennyiben elégedett az eredménnyel akkor futtassa a teszteseteket (Ctrl+F6). Ezek mindegyikének helyes eredménnyel kell lefutnia, hogy az összes feladatot késznek tekintse. A tesztesetek kódjához nem szabad nyúlni. Az előre megadott fájlok tartalmát ne módosítsa mert a tesztesetek nem fognak működni.

Beadási határidő: 2023-03-12 18 óra. Az ez után feltöltött feladatok vagy módosítások nem kerülnek értékelésre! A beküldés feltételei a GIT readme fájlban találhatók.

Alapok (fundamentals.Fundamental osztály metódusai)

A vizsgarész teljesítéséhez legalább 25 pontot kell elérnie. Ebbe beleszámítanak a plusz pontok is. Hajrá!

1. Metódusa egy csak az angol ABC betű karaktereit tartalmazó szöveget kap. Ebben a szövegben cserélje le az összes kis betűt nagy betűre és az összes nagy betűt kis betűre. A szöveg mindig helyes, de lehet üres („”) is. (*invertCase(String):String)*
   10 pont

2. A paraméterben kapott szöveg karaktereit rendezze ABC sorrendbe. Feltételezheti, hogy a paraméter nem tartalmaz ékezetes karaktereket és csak kisbetűk vannak benne. Nem használható semmilyen „beépített” rendező metódus. Mindegy, hogy melyik módszerrel végzi el a rendezést, de azt Önnek kell megírnia. *(orderChars(String): String)*
   10 pont

3. Az előző feladatban előállított szövegben számolja meg, hogy hány darab különböző karakter van. Számíthat rá, hogy a szövegben a karakterek (a-z) ASCII kód szerint emelkedő sorrendben vannak. *(countDifferentChars(String):int)*
   10 pont

4. Metódusa paraméterként egy előjeles egész számot kap és egy számjegyet (szintén egész szám, de csak egy arab számjegyet tartalmazz (0-9)). Térjen vissza igaz értékkel, ha a számjegy benne van a kapott számban. Lehetőleg ne alakítsa a számot szöveggé. Az is elfogadott megoldás, de pontlevonás jár érte. *(containsCertainDigit(int, int):boolean)*

10 - 3 pont

5. Metódusa két String típust kap paraméterként. Az első paraméter egy olyan szöveg mely helyőrzőket tartalmaz (ez az aláhúzás _ karakter). A második paraméter varargs típus azokat a karaktereket tartalmazza melyeket a helyőrzők helyére kell beilleszteni. Az első aláhúzás karakter helyére az első karaktert kell beilleszteni a második paraméterből és így tovább. Amennyiben több beillesztendő karakter van, mint helyőrző akkor a maradékkal nincs dolga. Ellenkező esetben a pár nélküli helyőrzők helyére a csere karakterek utolsó karakterét kell beilleszteni. A csere karakterek biztos tartalmazz legalább egy karaktert és soha nem tartalmazza a helyőrző karaktert. Metódusa az átalakított szöveggel kell, hogy visszatérjen. Törekedjen arra, hogy az algoritmusa csak annyit lépjen ahány aláhúzás karakter van a szövegben. Ez utóbbi feltétel teljesülése +3 pontot ér, ugyanez igaz arra az esetre is, ha nincs elágazás az algoritmusában. Összesen +6 pontot lehet a feladattal gyűjteni.
   (replaceUnderscores(String, String…): String)
   10 + 6 pont
   Példák:

Szöveg	Csere karakterek	Eredmény
___ *(három aláhúzás)*	„a” „b” „c”	abc
val_mi	„a” „x” „y”	valami
a kígyó _lb_völ__az_áldozatát_mielőtt *(hat aláhúzás)*	„e” „ű” „i” „ „ *(az utolsó karakter egy szünet karakter, csak négy csere)*	a kígyó elbűvöli az áldozatát mielőtt

OOP, adatbázis, állományok, gyűjtemények

Most, hogy túl vagyunk a nehezén jöhet a könnyebbik rész. Ez már csak móka és kacagás. A programnak egy adatbázist kell kezelnie. Az adatbázist is Önnek kell elkészítenie a megadott feltételek alapján. Feltöltéskor az adatbázist létrehozó sql scriptet helyezze el a program könyvtára alatt található Database könyvtárba. Ez lehet több script is és lehetőleg tartalmazzon helyes teszt adatokat is. A java kódot és a felhasználói felületet az OOP alapelvek betartásával kell megírni. Ahol lehet ott az absztrakció legmagasabb szintjét kell alkalmazni. Itt is igaz, hogy az osztályok csak egy dolgot csinálhatnak viszont azt nagyon jól. Amennyiben nem tartja be az alapelveket akkor hiába „működik” a program az a rész nem fog teljes vagy inkább túl sok pontot eredményezni. A felhasználói felület kinézetét nem pontozzuk, így ne a csinosítgatással töltse az időt. Abban az esetben, ha a felület használhatatlan vagy rosszul működik az díjazásra (citrom díj) kerül. Az adatbáziskezelést megvalósíthatja a jdbc („kézi” vagy „hagyományos”) módszerrel, de nyugodtan használhatja a JPA keretrendszert is. Vigyázat ez utóbbi használata esetén a Netbeans IDE nem elfogadható kódot ír (hosszú metódusok, többször leírt ugyanolyan kód, stb). Szerintem jobban jár, ha kézzel írja meg az adatkezelő osztályokat és az entitás osztályokat, főleg ha a táblák közt felfedez azonos tulajdonságokat. A program tartalmazz egy oop csomagot lehetőleg ezen csomag alá helyezze el a saját csomagjait. Ebben a csomagban talál egy MainForm osztályt ezt tudja használni a program fő ablakának. Persze nem kötelező készíthet saját fő ablakot is. A program futtatásához állítsa át a program properties -> Run -> Main Class alatt a fő osztályt. Hajrá!

Adatbázis:

A mysql adatbázis szerver segítségével készítsen adatbázist „storage” néven. Ebben az adatbázisban hozzon létre három táblát az alábbiak alapján:

perishable_product (romlandó termék) tábla:

article_number (cikkszám): varchar(10) PK formátuma: két nagy betű mindig: PP majd nyolc számjegy

name (név): nvarchar(150) not null: szabad szöveg

brand (márka) nvarchar(50) nullable: szabad szöveg pl.: (Turcsi túrógyár, Búúú tej, stb)

family (típus, család) nvarchar(50) not null: szabad szöveg pl.: élelmiszer, vegyi árú

netto_price (nettó ár) int not null: a termék egység ára nem lehet negatív

tax_id (ÁFA idegen kulcs) int not null: egész szám idegen kulcs a state_sales_tax táblából

quantity (mennyiség) int not null: a raktárban levő mennyiség nem lehet negatív érték

amount_units (mennyisége egység) nvarchar(10) not null: szabad szöveg pl.: kg, darab, deka

critical_quantity (kritikus mennyiség) int not null: azt jelzi mikor kell újra rendelni az adott termékből

expiration_date (lejárati idő) date not null

production_date (gyártás dátuma) date not null

durable_product (tartós fogyasztási cikk) tábla:

article_number (cikkszám): varchar(10) PK formátuma: két nagy betű mindig: DP majd nyolc számjegy

name (név): nvarchar(150) not null: szabad szöveg

brand (márka) nvarchar(50) nullable: szabad szöveg pl.: (Calcy Calculators, ChinePearl washing machines factory)

family (típus, család) nvarchar(50) not null: szabad szöveg pl.: élelmiszer, vegyi árú, háztartási cikk

netto_price (nettó ár) int not null: a termék egység ára nem lehet negatív

tax_id (ÁFA idegen kulcs) int not null: egész szám idegen kulcs a state_sales_tax táblából

quantity (mennyiség) int not null: a raktárban levő mennyiség nem lehet negatív érték

amount_units (mennyisége egység) nvarchar(10) not null: szabad szöveg pl.: kg, darab, deka

critical_quantity (kritikus mennyiség) int not null: azt jelzi mikor kell újra rendelni az adott termékből min. 0 ekkor a termék elfogyhat

waranty_period (jótállási idő) not null int: a jótállás időtartama hónapban

gross_weight (bruttó súly) decimal(4,1) not null: a csomag súlya kg nagyobb mint 0

state_sales_tax (ÁFA) tábla:

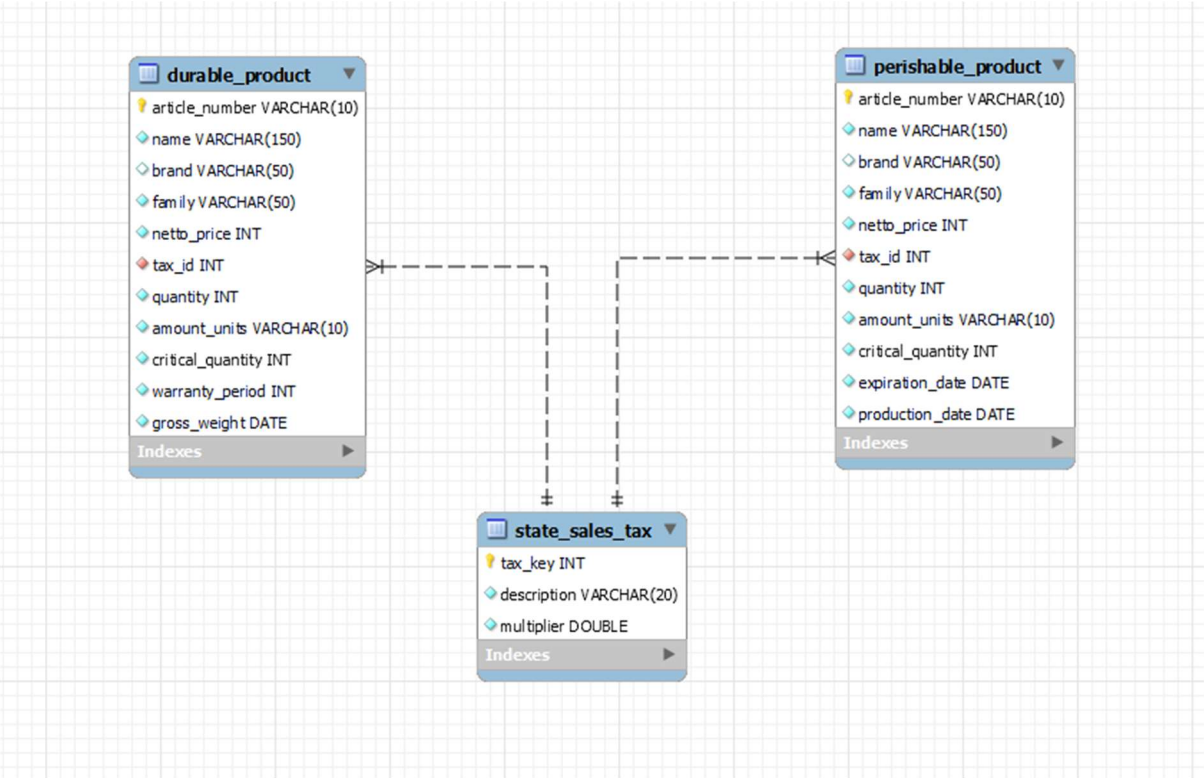
tax_key int PK: Az áfa kulcs 27% esetén 27 az értéke lehet 0 és 100 között

description (leírás) varchar(20) not null: pl.: 27%, 0%, …

multiplier (szorzó) double not null: pl.: 1.27, 1.05, ….

(20 pont)

A lenti ábra megmutatja az adatbázis szerkezetét.



Plusz pont (10 pont):

Készítsen tárolt eljárást valamelyik vagy kettőt mindkét termék táblára. Az eljárás egy terméket tud eltárolni a megfelelő táblába olyan módon, hogy:

- a termék minden adatát megkapja paraméterként
- ebbe beletartozik az áfa kulcs is, de csak a kulcs
- abban az esetben, ha nem létezik a megadott áfa az insert eljárással eltárolja a megfelelő táblába (a kulcs segítségével minden előállítható. Ugye feltűnt?)
- a termék tárolásra kerül az új vagy ha létezett a régi áfa kulccsal együtt.
- eljárása igaz értékkel (select 1 as success) tér vissza a sikeres tárolás esetén

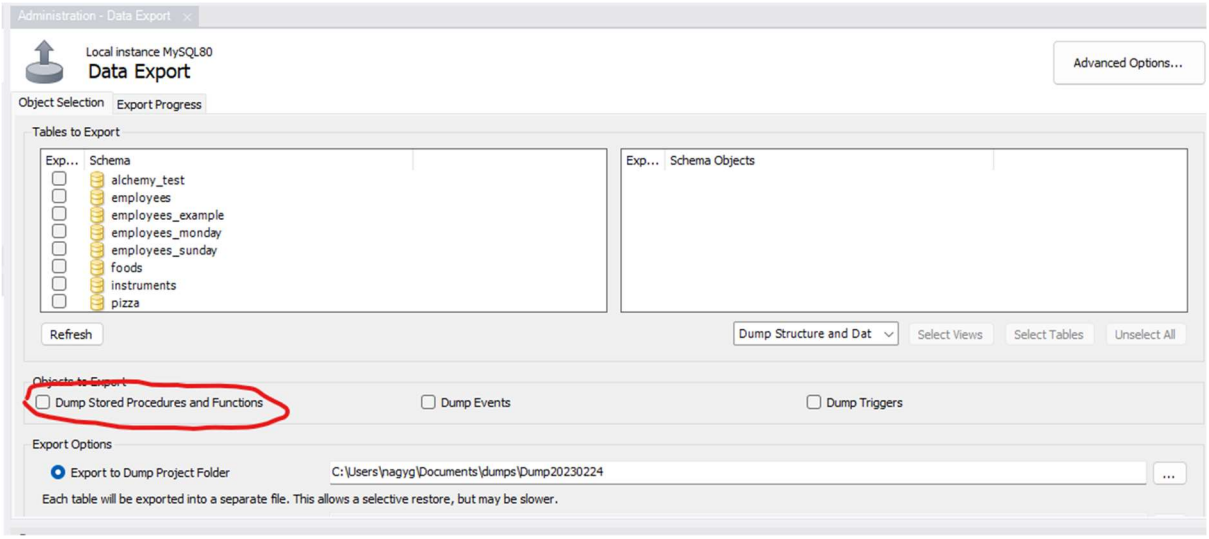
The Lord of the Pluses (I beg your pardon Sir Tolkien!) (5 pont):

Bármely jó ötlet, metódusok tárolt eljárások megírása maximum 5 pont díjazással jutalmazható.

Pl.: a cikkszám formátum ellenőrzése function-ban a tárolt eljárásban, vagy az á….

Fenti feladat megoldásával összesen: 20 + 10 + 5 = 25 azaz huszonöt pontot lehet gyűjteni.

A „dump” készítése esetén ne feledje bepipálni a megfelelő check box-okat, hogy a tárolt eljárások és a funkciók is bekerüljenek a scriptbe!



Persze ez még kevés a sikeres vizsgához. Úgyhogy haladjunk tovább a rögös, de legalább szép tájon vezető ösvényünkön. Lehet, hogy találkozunk a „MI NORRISUNKAL” ahogy bit pocsolyákat ugrál át 😊.

A PROGRAM:

Mivel az adatbázis szerkezete meghatározza a program struktúráját ezért arra nem vesztegetünk bit tintát. Egy a lényeg, hogy a program a tanult és unalomig ismételt OOP szabályok szerint legyen megírva.

Lent a funkcionális és a nem funkcionális követelményeket fogom megadni. Ezeknek működnie kell. A programnak futtatható állapotban kell lennie nem lehet benne piros aláhúzás (az automatikusan 0 pont). Megjegyzések lehetnek a program szövegében, de ezek nem fognak pontot érni (hacsak nem a frankó jövő heti lottó számokat tartalmazza).

Figyelmeztetés a mellékhatásokra: Az OOP szabályok be nem tartása súlyos pont vesztességgel járhat. Amennyiben ilyet tapasztal forduljon bit doktorához, oop gyógyszerészéhez.

Nem funkcionális követelmények

NFR00001: Olyan cikkszám nem jelenhet meg a programban vagy az adatbázisban mely nem felel meg a táblák leírásában megadottnak.

NFR00002: Ahol meghatározták a minimum maximum értéke ott a tartományba nem tartozó érték nem jelenhet meg.

NFR00003: A keletkezett kivételeket a megfelelő program rétegben kezelni kell. Egy sql kivétel nem szivároghat fel a nézet rétegbe.

NFR00004: A rétegeknek lazán csatoltnak kell lennie (interface, factory egyéb design pattern) használata erősen javasolt.

NFR00005: Az adatbázisból beolvasott adatokat valamilyen Java vagy saját maga által írt (pl.: observable list felhasználható) gyűjteménybe tartsa.

NFR00006: Programja nem formatálhatja a felhasználó gépének a merevlemezét és politikailag is korrektnek kell lennie 😊 .

Funkcionális követelmények:

FR00001: Programja induláskor valamilyen módon (felugró ablak, gumikalapács, hangjelzés) figyelmeztesse a raktárost, ha van kritikus szint alatti termék a raktárban. Ezekről lehessen egy listát kérni (jtable).

FR00002: A romlandó termékek tudják nap mértékegységben „megmondani”, hogy hány nap múlva járnak le.

FR00003: Minden termék tudja kiszámolni a bruttó árát

FR00004: Lehessen terméket és áfa kulcsot felvinni az adatbázisba. Az áfa felvitele történhet automatikusa is a termékkel együtt (tárolt eljárás vagy a java kód keretében).

FR00005: Termékekre lehessen keresni cikkszám vagy név töredékek alapján. Egy-egy keresés csak az egyik termék táblában kell, hogy dolgozzon.

FR00006: Legyen egy felület, ahol statisztikát lehet látni valamelyik termék típusról. Ebben legyen benne az áfa kulcs szerint csoportosított termékeknek az összes nettó ára, bruttó ára, átlag nettó ára és az összes adott áfa kulcsba tartozó termék darabszáma.

FR00006: A termékeket egy-egy táblában (jtable) mutassa meg a felhasználónak. A tábla fejlécére kattintva a megfelelő oszlop szerint kerüljön rendezésre a tábla.

FR00007: A termékek (tartós, romlandó) ne keveredjenek, ne kerüljenek egy táblába a program szintjén. Mindig legyen egyértelmű a termék fajtája pl.: label -> romlandók vagy bármilyen más jelzés.

FR00008: Az adatbázisban létező termékeket lehessen bevételezni, kivételezni (persze csak a raktárban levő mennyiségig). Ezeknek a változásoknak az adatbázisba be kell kerülni.

FR00009: A programja készítsen a program gyökér könyvtárába egy szöveges .log kiterjesztésű fájl. Ebben a fájlban minden a termékekkel kapcsolatos tranzakciónak (új termék, kivét, bevét, update) meg kell jelennie egy új sorban a következő formában:

dátum és idő, cikkszám, tranzakció típusa

dátum és idő: az az időpont melyben a felhasználó elkövette a tranzakciót (lehet timestamp is) System.currentTimeMillis()

cikkszám: a termék cikkszáma melyen a tranzakció történt.

tranzakció típusa: új termék, módosítás, kivétel, bevétel értékek jelenhetnek meg. Kivét, bevét esetén legyen ott a darabszám is.

FR00010: A felhasználó egy gomb vagy menü választásával lásson egy file save dialógust. Ebben tudjon megadni egy könyvtárat és egy fájlnevet. A megadott könyvtárban jöjjön létre egy másolat az aktuális log fájlról txt kiterjesztéssel. Ez után a program log fájlját ürítse ki és a további bejegyzések ebbe az üres fájlba kerüljenek.

KÖNNYÍTÉS:

AMIT A FENTI SPECIFIKÁCIÓK NEM HATÁROZNAK MEG AZT NEM KELL MEGVALÓSÍTANI.

Amennyiben ellentmondást vagy valamilyen nem érthető dolgot talál a feladat leírásában kérdezzen bátran a nagy.gabor70@gmail.com címen.

A sikeres vizsgához nem kell minden funkciónak működnie, de igyekezzen a lehető legtöbbet kihozni belőle. Abban a szinte lehetetlen esetben, ha elakadna. Töltse fel a kódját és kérdezzen email-ben. Igyekszem segíteni a vizsga integritásának a betartásával. Magyarul kódot nem írok csak iránymutatást tudok adni. Ne feledje a Google a barátja.

Sok sikert és soha ne adja fel a csüggedését.

Nagy Gábor