Záverečný test
Zadanie

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Dvakrát meraj (rozmýšľaj), raz rež (programuj)

Dôležité pravidlá a informácie (viac na stránke predmetu)**:**

- čas na riešenie úloh je **240 minút**,
- nie je dovolená žiadna (elektronická aj neelektronická) komunikácia s kýmkoľvek okrem dozoru
- v prípade akýchkoľvek problémov alebo z dôvodu ohodnotenia riešenia kontaktujte dozor,
- riešenia je možné nechať si ohodnotiť aj priebežne (nie až v závere testu),
- **funkčnosť každej metódy musí byť preukázaná spustením na vami vytvorenom testovacom vstupe, nespustiteľné metódy neumožňujú zisk príslušných bodov,**
- všetky inštančné premenné musia byť neverejné.

Turtle shopping park & Christmas music

Motivácia: Sieť obchodných domov Turtle shopping park (TSP) sa rozhodla vykonať audit spokojnosti zákazníkov a zamestnancov počas Vianočných sviatkov. Ukázalo sa, že množstvo tematickej hudby viacerým vadí. Problém audio vizuálneho smogu sa rozhodli riešiť. Ale ku kvalitným rozhodnutiam treba pristupovať aj na základe dát. Preto najprv treba analyzovať všetky pustené nahrávky v ich obchodných domoch. Okrem hudobnej produkcie môžete v obchodných domoch počuť aj oznamy a reklamu. Preto sa múdre korytnačky rozhodli analyzovať históriu nahrávok vo všeobecnejšej rovine ako iba podiel vianočnej hudby. Tvojou úlohou bude navrhnúť aplikáciu na analýzu histórie nahrávok

Pohľad analytika: Pri implementácii aplikácie budeme potrebovať:

- triedu `AudioBlok`, ktorá reprezentuje jednu pesničku, reklamu alebo oznam, ktorý odznel,
- triedu `AudioZoznam`, ktorá bude uchovávať zoznam audio blokov, ktoré odzneli v obchodnom dome.

Zadanie: V balíku `sk.upjs.finalTerm` vytvorte triedu `AudioBlok` obsahujúcu dátové položky prístupné cez gettre (a podľa uváženia aj modifikovateľné cez settre):

- **`datum`** (dátum, kedy bol audioblok prehraný, vo formáte DD.MM.RRRR, napr. "05.12.2020" – v dátume sú vždy úvodné nuly doplnené tak, aby deň aj mesiac boli dvojciferné),
- **`casZaciatku`** (čas, kedy začal audioblok, vo formáte HH:MM:SS, napr. "13:05:45", v čase sú vždy úvodné nuly doplnené tak, aby sa každá časť časového reťazca skladala z 2 cifier, resp. bola dvojciferná),
- **`casKonca`** (čas, kedy doznel audioblok, v rovnakom formáte ako casZaciatku, navyše pre jednoduchosť predpokladáme, že casKonca je vždy v rovnaký deň ako casZaciatku, teda prístup nie je počas polnoci),
- **`pouzivatel`** (meno používateľa, ktorý spustil audioblok, napr. meno zamestnanca "Jožko Mrkvička" alebo "playlist" ak ide o playlist, ktorý bol pustený automaticky)
- **`nazovNahravky`** (názov súboru v úložisku, napr. "AllIWantForChristmas.mp3")
- **`typ`** ("hudobná nahrávka", "reklama", "oznam" alebo iné podľa toho, o aký typ ide)
- **`autor`** (meno autora alebo autorov skladby, ak ide o hudobnú nahrávku)
- **`styl`** (štýl(štýly) skladby, ak ide o hudobnú nahrávku, napr. rock, pop, folk, jazz metal, klasicka hudba, funk, alebo kombinácia rôznych štýlov)
- **`temaSkladby`** (téma, ak ide o hudobnú nahrávku, napr. "Vianoce", "leto", "láska", alebo iné)

Upozornenie: Zadanie pre triedu `AudioBlok` predpisuje dátové položky prístupné cez gettre. Aké privátne inštančné premenné použijete na uloženie týchto dátových položiek je na vašom rozhodnutí.

Nie všetky audio bloky sú hudobné nahrávky, a preto nemajú vyplnené položky autor žáner a tému skladby.

Ďalej vytvorte aj triedu `sk.upjs.finalTerm.AudioZoznam`, ktorá bude uchovávať zoznam audio blokov.

Konštruktory a pridávanie audio blokov do zoznamu (3 body dokopy – povinné):

- `public AudioBlok(String datum, String casZaciatku, String casKonca, String pouzivatel, String nazovNahravky, String typ, String autor, String styl, String temaSkladby)` – použije sa na vytvorenie záznamu o audio bloku, ak ide o hudobnú nahrávku.
- `public AudioBlok (String datum, String casZaciatku, String casKonca, String pouzivatel, String nazovNahravky, String typ)`– použije sa na vytvorenie záznamu o audio bloku, ktorý nie je hudobnou nahrávkou.
- `public void pridaj(AudioBlok audioBlok)` – inštančná metóda v triede `AudioZoznam`, ktorá pridá záznam o jednom audio bloku do audio zoznamu obchodného domu.

Práca so súbormi (povinné):

V triede `AudioBlok`:

- `public static AudioBlok zoStringu(String popis)` – statická metóda, ktorá vráti referenciu na novovytvorený objekt triedy `AudioBlok`. Parameter je `String` v tvare "`datum \t casZaciatku \t casKonca \t pouzivatel \t nazovNahravky \t typ \t autor \t styl \t temaSkladby`", resp. "`datum \t casZaciatku \t casKonca \t pouzivatel \t nazovNahravky \t typ`", ak neide o hudobnu nahravku(3 body);
  Poznámka: Znak `\t` je neviditeľný znak tabulátora. `Scanner`-u môžete povedať, že oddeľovač má byť tabulátor zavolaním jeho metódy `useDelimiter("\t").` Medzera pred a za `\t` sú len kvôli zlepšeniu čitateľnosti zadania, v reťazci reálne nie sú.
  Poznámka: Forma oddeľovania autorov, štýlov a tém v `string`-och je na vašej voľbe.
- `public String toString()` – vráti reťazec vhodne reprezentujúci údaje o audio bloku (1 bod).

V triede `AudioZoznam`:

- `public static AudioZoznam zoSuboru(String nazovSuboru)` – statická metóda, ktorá z uvedeného súboru prečíta zoznam audio blokov, pričom v každom riadku bude popis jedného audiobloku (4 body).
- `public void uloz(String nazovSuboru)` – uloží všetky audio bloky z audio zoznamu do súboru v tvare, ktorý vie spracovať metóda `zoSuboru(String nazovSuboru)` (3 body).
- `public String toString()` – vráti reťazec vhodne reprezentujúci všetky audio bloky uchované v audio zozname obchodného domu (1 bod).

Nasledujúce úlohy môžete riešiť v ľubovoľnom poradí. Ale riešenie jednej môže zjednodušiť nasledujúce.

Inštančné metódy triedy `AudioBlok`:

- `public int vratCas()` – vráti, koľko sekúnd trval audio blok (1 bod). Hint.: čas začiatku a konca sa prevedie na sekundy od začiatku dňa a vráti sa ich rozdiel.
- `public List<String> vratAutorov()` – vráti list autorov, ak má súbor vyplnenú túto vlastnosť, inak vráti `null` (2 body).

Inštančné metódy triedy `AudioZoznam`:

Ak niektorá z metód nevie vrátiť referenciu na objekt s požadovanými vlastnosťami, metóda nech vráti `null`.

- `public int vratCas()` – vráti koľko celých hodín trvali dokopy audio bloky (1 bod).
- `public int najdlhsiOznam()` – vráti koľko sekúnd trval najdlhší oznam (1 bod).
- `public List<String> vratOblubenychAutorov(String pouzivatel)` – vráti všetkých autorov, od ktorých používateľ pustil aspoň jednu skladbu; každý autor sa v zozname môže vyskytovať najviac raz (2 body).
- `public Set<String> preruseneSkladby(Map<String, Integer> dlzkySkladieb)` – vráti množinu názvov hudobných nahrávok, ktoré nedozneli do konca, parameter `dlzkySkladieb` obsahuje názvy skladieb a ich dĺžky v sekundách (3 body). +3body ak množina neobsahuje skladby, ktoré síce boli prerušené oznamom ale po ozname doznel ich zvyšok.
- `public Set<String> vratSkladbyPodlaTemyAZanru(String tema, String styl)` – vráti množinu názvov skladieb, ktoré obsahujú tému určenú parametrom `tema` a žáner určený parametrom `styl` (2 body). +2 body ak je štýl `null` alebo prázdny reťazec, tak metóda vráti hudobné nahrávky len podľa témy, podobne ak je téma `null` alebo prázdny reťazec, tak berieme do úvahy iba štýl.
- `public double percentoTematickejHudby(String tema)` – vráti koľko percent celkového času hudobnej produkcie je zamerané na tému určenú parametrom (2 body).
- `public Map<String, Integer> vratCasOznamovPouzivatelov()` – vráti mapovanie, ktoré používateľovi priradí počet sekúnd oznamov ktoré ohlásil (3 body)
- `public AudioZoznam vratAudioZoznamZaObdobie(String odDatumu, String poDatum)` – metóda vráti audio zoznam za obdobie určené parametrami (vrátane dní určených parametrami). (3 body).
- `public double ziskZReklamy(double[] cenaPocasHodiny, String odDatumu, String poDatum)` – `cenaPocasHodiny` je 24 prvkové pole, kde sa na i-tom indexe v poli nachádza cena za celú minútu reklamy počas i-tej hodiny. Ak ide o reklamu, ktorá začne a skončí v rôznych hodinách, tak cena reklamy je určená podľa času jej začiatku. Metóda vypočíta a vráti celkový zisk za reklamu v období `odDatumu` po `poDatum` (4 bodov).
- `public Map<String, Integer> najhranejsieSkladby()` – vráti mapu, kde je každej hudobnej nahrávke priradené aké celkové množstvo času v minútach (po zaokrúhlení) ju bolo možné počuť (4 body).
- `public List<String> top5NajhranejsieSkladby()` – na základe predchádzajúcej metódy vytvorte novu metódu, aby vrátila 5 najhranejších skladieb v poradí (3 body). Pozn. Bez korektného riešenia predchádzajúcej úlohy nemôžete riešiť túto.
- `public Map<String, Integer> topStyly()` – vytvorte mapu, kde je štýlu priradené koľko percent času celkovej hudobnej produkcie mu zodpovedá. Ak má skladba viac štýlov, tak sa čas rozdelí rovnomerne medzi všetky štýly, napr. ak máme iba jednu skladbu, ktorá má dva štýly, tak budeme mať 50% pre každý štýl (4 body).
- `public List<AudioBlok> mimoOtvaracejDoby(String otvorenie, String zatvorenie)` – vráti zoznam audio blokov, ktoré zneli aspoň sekundu mimo otváracích hodín, ktoré sú uvedené vo formáte HH:MM, napr. "08:15"(3 body). +3 body, ak ignorujeme 30 minút pred otvorením a 30 minút po zatvorení.
- `public boolean skontrolujBloky(String datumu)` – metóda vráti `false`, ak dva audio bloky počas daného dňa zneli v tom istom čase (2 body). + 5 bodov za riešenie, ktoré vykoná iba jeden prechod audio zoznamom.

Výnimka (4 body)

Vytvorte nekontrolovanú výnimku `NekorektnyDatum` a vhodne ju použite aspoň v jednej metóde. Táto výnimka má byť vyhodená vtedy, keď nejaký reťazec nie je korektným dátumovým reťazcom. Korektný dátumový reťazec je tvaru DD.MM.RRRR, pričom $1 \leq DD \leq 31$, $1 \leq MM \leq 12$, $1000 \leq RRRR \leq 9999$. Zvážte vytvorenie statickej metódy v triede `AudioBlok,` ktorá vyhodí výnimku, ak parametrom zadaný reťazec nie je korektný dátumový reťazec.

- `public static void overRetazec(String datum) throws NekorektnyDatum`

Poznámka k riešeniu: Ak chcete ako oddeľovač (delimiter) tokenov pre `Scanner` nastaviť bodku, použite `useDelimiter("\\.")`