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.

Nápojový automat

Motivácia: Automaty pozná hádam každý. Nápojové automaty sú v škole na mnohých miestach, konečnostavové sú zas vo vyšších ročníkoch. V tomto zadaní sa pozrieme na tie prvé. Našou úlohou je vytvoriť systém, ktorý pomôže analyzovať predaje v moderných nápojových automatoch.

Pohľad analytika: Pri implementácii aplikácie budeme potrebovať:

- triedu `Predaj`, ktorá reprezentuje jeden predaj nápoja
- triedu `ZoznamPredajov`, ktorá bude uchovávať zoznam predajov nápojov.

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

- **`datum`** (dátum, kedy bol predaj nápoja uskutočnený vo formáte DD.MM.RRRR, napr. "11.01.2019" – v dátume sú vždy úvodné nuly doplnené tak, aby deň aj mesiac boli dvojciferné),
- **`cas`** (čas, kedy bol predaj nápoja uskutočnený 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),
- **`nazov`** (názov predaného nápoja, napr. "Citrónový čaj", alebo "Káva so smotanou"),
- **cukor** (množstvo cukru: 0 – bez cukru, 1 – menej cukru, 2 – normálny cukor, 3 – extra cukor)
- **cena** (celková cena predaného nápoja v EUR, napr. *0.60*)
- **platba** (čiarkou oddelený zoznam hodnôt vhodených mincí, napr. "0.20, 0.30, 0.20", ak bola platba uskutočnená v hotovosti vhodením mincí, alebo reťazec tvaru "KARTA 1234 5678 1234 5678", ak bol nákup zaplatený platobnou kartou – 4 štvorčíslia tvoria číslo karty)
- **`extra`** (čiarkami oddelený zoznam extra volieb – napr. „BEZ KELIMKU, MENEJ VODY“)

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

Ak bol nákup zaplatený hotovosťou, je takýto nákup anonymný. Ak bol nákup zaplatený kartou, môžeme číslo karty použiť ako identifikátor platiteľa (kupujúcej osoby) a tak analyzovať jeho/jej zvyklosti.

Ďalej vytvorte aj triedu `sk.upjs.finalTerm.ZoznamPredajov`, ktorá bude uchovávať zoznam predajov v automate.

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

- `public Predaj(String datum, String cas, String nazov, int cukor, double cena, String platba)` – použije sa na vytvorenie záznamu o predaji nápoja bez extra volieb.
- `public Predaj(String datum, String cas, String nazov, int cukor, double cena, String platba, String extra)` – použije sa na vytvorenie záznamu o predaji nápoja s extra voľbami
- `public void pridaj(Predaj predaj)` – inštančná metóda v triede `ZoznamPredajov`, ktorá pridá záznam o predaji do zoznamu predajov.

Práca so súbormi (povinné):

V triede `Predaj`:

- `public static Predaj zoStringu(String popis)` – statická metóda, ktorá vráti referenciu na novovytvorený objekt triedy `Predaj`. Parameter je `String` v tvare "`datum \t cas \t nazov \t cukor \t cena \t platba`", resp. "`datum \t cas \t nazov \t cukor \t cena \t platba \t extra`", ak predaj nápoja obsahuje extra voľby (3 body);
  Poznámka: Znak `\t` je neviditeľný znak tabulátora. `Scanner`-u môžete povedať, že oddeľovač má byť tabulátor zavolaním jeho metódy `useDelimiter("\t").` Medzera pre a za `\t` sú len kvôli zlepšeniu čitateľnosti zadania, v reťazci reálne nie sú.
- `public String toString()` – vráti reťazec vhodne reprezentujúci údaje o predaji nápoja (1 bod).

V triede `ZoznamPredajov`:

- `public static ZoznamPredajov zoSuboru(String nazovSuboru)` – statická metóda, ktorá z uvedeného súboru prečíta zoznam predajov nápojov, pričom v každom riadku bude popis jedného predaja nápoja (4 body).
- `public void uloz(String nazovSuboru)` – uloží všetky predaje nápojov v zozname do súboru v tvare, ktorý vie spracovať metóda `zoSuboru(String nazovSuboru)` (3 body).
- `public String toString()` – vráti reťazec vhodne reprezentujúci všetky predaje v zozname predajov (1 bod).

Inštančné metódy triedy `Predaj`:

- `public boolean jePlatbaKartou()` – vráti, či nákup bol zaplatený kartou (1 bod).
- `public double[] vratMince()` – vráti hodnoty vhodených mincí alebo `null`, ak bol predaj zaplatený kartou (2 body).
- `public String vratCisloKarty()` – vráti číslo karty bez medzier alebo `null`, ak bol predaj zaplatený v hotovosti (1 bod).

Inštančné metódy triedy `ZoznamPredajov`:

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

- `public double vratTrzbu()` – vráti celkovú tržbu (1 bod).
- `public double vratPodielKariet()` – vráti koľko percent predajov bolo zaplatených platobnou kartou (1 bod).
- `public ZoznamPredajov vytvorZoznamNaDen(String datum)` – vráti referenciu na novovytvorený zoznam predajov, ktorý bude obsahovať len predaje realizované v zadaný dátum. Vzájomné poradie záznamov musí byť zachované (2 body).
- `public List<String> vratCislaKariet()` - vráti čísla kariet, ktorými bolo platené; každé číslo karty sa vo vrátenom zozname môže vyskytovať najviac raz (2 body).
- `public int pocetUsetrenychKelimkov()` – vráti počet predajov, ktoré majú v extra voľbách "BEZ KELIMKU" (2 body).

- `public double vratSumuVhodenychMinci()` – vráti celkovú sumu, ktorá bola vhodená v minciach (2 body).
- `public Map<String, Integer> vratPoctyPredajov()` – vráti mapovanie, ktoré ku každému nápoju priradí počet predaných kusov tohto nápoja (3 body).
- `public int[] vytvorCasovyHistogram(Set<String> napoje)` – vráti 24 prvkové pole, ktoré na indexe `h` bude obsahovať počty predajov, ktoré sa uskutočnili v `h`-tej hodine. Zároveň uvažujeme iba predaje nápojov, ktorých názvy sú v parametrom zadanej množine názvov nápojov. Ak je parameter `napoje` `null`, uvažujeme všetky nápoje (5 bodov).
- `public ZoznamPredajov vytvorZoznamVObdobi(String odDatumu, String poDatum)` - vráti referenciu na novovytvorený zoznam predajov, ktorý bude obsahovať len predaje realizované medzi dátumami `odDatumu` a `poDatum` (vrátane). Vzájomné poradie záznamov musí byť zachované (6 bodov).
- `public List<String> najdiMaloSladke()` – vráti názvy tých nápojov, u ktorých je treba zvážiť dosladenie, t.j. počet predajov s voľbou cukru „extra“ (3) prevyšuje v súčte počet predajov nápojov s voľbou menej cukru (1) a normálny cukor (2). Voľbu bez cukru (0) v tejto analýze neuvažujeme. Vo vrátenom zozname nech je každý názov nápoja nanajvýš raz (6 bodov).
- `public Set<String> klientiSJasnouVolbou()` – vráti čísla kariet tých klientov, ktorí si stále vyberajú rovnaký nápoj (6 bodov + 5 bodov za implementáciu, ktorá zoznamom predajov preiteruje len raz).
- `public void simulujPredaje(Map<Double, Integer> zasobnikMinci)` – vytvorte simuláciu stavu zásobníka mincí za predpokladu, že predaje sú v zozname predajov usporiadané chronologicky. Na začiatku sú v zásobníku mince, ktorých počet je určený mapovaním `zasobnikMinci` - ku hodnote mince je priradený ich počet. Teda hodnota `zasobnikMinci.get(0.2)` určuje, koľko 20-centových mincí je v zásobníku. Ak hodnota nie je v mapovaní, je to rovnaké, akoby počet mincí tejto hodnoty bol 0. Pri každej platbe mincami sa jednotlivé vhodené mince najprv uložia do zásobníka mincí. Ak je treba vydať mince, na vydanie výdavku *V (V>0)* v minciach sa použije algoritmus:
  1. nájdi takú mincu v zásobníku, že jej hodnota *H* je najväčšia možná a $H \leq V$ – ak takej niet, vyhoď výnimku indikujúcu, že nie je dostatok mincí v zásobníku
  2. vydaj mincu s hodnotou *H*
  3. zníž *V* o hodnotu *H* vydanej mince
  4. ak $V > 0$, choď na krok 1, inak ukonči vydávanie

  Na konci simulácie všetkých predajov, parametrom referencované mapovanie `zasobnikMinci` nech obsahuje nový stav zásobníka mincí (12 bodov).

Výnimky (5 bodov)

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 `NekorektnyDatum`, ktorá vyhodí výnimku, ak parametrom zadaný reťazec nie je korektný dátumový reťazec.

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

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