Záverečný test
Zadanie

Ústav informatiky
Prírodovedecká fakulta
UPJŠ v Košiciach

Dvakrát meraj (rozmýšľaj), raz rež (programuj)

Pravidlá a informácie:

- 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,
- **funkčnosť každej metódy musí byť preukázaná spustením na vami vytvorenom testovacom vstupe, nespustiteľné metódy neumožňujú zisk príslušných bodov,**
- všetky inštančné premenné musia byť neverejné.

Rozprávkovo

Motivácia: V kráľovstve rozprávok žije nespočetné množstvo rôznych obyvateľov - od malých trpaslíkov až po obrovských drakov. Každý z nich má nejakú jednu čarovnú schopnosť (napr. „neviditeľnosť“, „čítanie myšlienok“, ...) a možno jednoznačne určiť, či patrí na stranu dobra alebo zla. Na čele kráľovstva stojí kráľovná Zuzana. Ako dobrá a múdra kráľovná v duchu hesla *panem et circenses* často organizuje pre svojich poddaných rozličné podujatia – „social eventy“ (napríklad zlet dobrých čarodejníc, kongres trpaslíkov a pod.). No zorganizovať taký event, to nie je len tak. Totiž niektoré postavičky sa na jednom mieste z pochopiteľných dôvodov zísť nemôžu. Ako informatička sa rozhodla implementovať systém, ktorý by jej pomáhal pri zostavovaní zoznamov hostí tak, aby sa predišlo nepríjemným situáciám počas týchto „social eventov“.

Pohľad analytika: Pri implementácii budeme potrebovať:

- triedu `Obyvatel`, ktorá bude uchovávať údaje o jednom obyvateľovi kráľovstva,
- triedu `ZoznamObyvatelov`, ktorá bude uchovávať nejaký zoznam obyvateľov kráľovstva (napr. všetkých obyvateľov kráľovstva).

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

- **meno** (meno obyvateľa)
- **typ** (napr. trpaslík, čarodejnica, vlkolak, atď...)
- **carovnaSchopnost** (pomenovanie čarovnej schopnosti)
- **charakter** (či osoba patrí na stranu dobra alebo zla, napr. „dobro“ a „zlo“)
- **datumNarodenia** (dátum narodenia v obvyklom formáte, napr. „17.12.2014“)
- **telefonneCislo** (ak je známe, tak telefónne číslo na obyvateľa – nemusí byť zadané, niektorí obyvatelia sa z rôznych dôvodov bránia telefónom)

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

Ďalej vytvorte aj triedu `sk.upjs.finalTerm.ZoznamObyvatelov`, ktorá bude uchovávať nejaký zoznam obyvateľov kráľovstva (napr. zoznam všetkých známych obyvateľov kráľovstva).

Konštruktory a pridávanie osôb (3 body dokopy – povinné):

- `public Obyvatel(String meno, String typ, String carovnaSchopnost, boolean jeDobry, String datumNarodenia)` – použije sa na vytvorenie záznamu o obyvateľovi, ktorý nemá telefón.
- `public Obyvatel(String meno, String typ, String carovnaSchopnost, boolean jeDobry, String datumNarodenia, String telefonneCislo)` – použije sa na vytvorenie záznamu o obyvateľovi, ktorý vlastní telefón.
- `public void pridaj(Obyvatel obyvatel)` – inštančná metóda v triede `ZoznamObyvatelov`, ktorá pridá záznam o obyvateľovi do zoznamu obyvateľov.

Práca so súbormi (povinné):

V triede `Obyvatel`:

- `public static Obyvatel zoStringu(String popis)` – statická metóda, ktorá vráti referenciu na novovytvorený objekt triedy `Obyvatel`. Parameter je `String` v tvare "meno `\t` typ `\t` carovnaSchopnost `\t` charakter `\t` datumNarodenia", resp. "meno `\t` typ `\t` carovnaSchopnost `\t` charakter `\t` datumNarodenia `\t` telefonneCislo", ak obyvateľ má telefón (3 body);
  Poznámka: Znak `\t` je neviditeľný znak tabulátora. `Scanner`-u môžete povedať, že oddeľovač má byť tabulátor zavolaním jeho metódy `useDelimiter("\t").`
- `public String toString()` – vráti reťazec vhodne reprezentujúci údaje o obyvateľovi (1 bod).

V triede `ZoznamObyvatelov`:

- `public static ZoznamObyvatelov zoSuboru(File f)` – statická metóda, ktorá z uvedeného súboru prečíta zoznam obyvateľov, pričom v každom riadku bude popis jedného obyvateľa (4 body).
- `public void uloz(File subor)` – uloží všetky záznamy o všetkých obyvateľoch v zozname do súboru v tvare, ktorý vie spracovať metóda `zoSuboru(File f)` (3 body).
- `public String toString()` – vráti reťazec vhodne reprezentujúci všetkých obyvateľov v zozname (1 bod).

Inštančné metódy triedy `ZoznamObyvatelov`:

- `public double podielZla()` – vráti koľko percent obyvateľov je na strane zla (1 bod).
- `public List<String> dobriAZastihnuteli()` – vráti mená obyvateľov s telefónom na strane dobra (1 bod).
- `public boolean suCarovneDvojicky()` – vráti, či sa v zozname nachádzajú takí dvaja obyvatelia, ktorí majú rovnaký typ, charakter aj čarovné schopnosti (2 body).
- `public Map<String, Integer> stavObyvatelstva()` – vráti mapovanie, ktoré každému typu obyvateľa v zozname priradí celkový počet obyvateľov tohto typu v zozname (4 body).
- `public int najstarsiZ(String typ)` – vráti rok narodenia najstaršieho obyvateľa zadaného typu (5 bodov).
- `public List<String> typyVZozname()` - vráti zoznam pomenovaní všetkých typov obyvateľov v zozname, pričom každé pomenovanie sa nachádza v zozname len raz (5 bodov)
- `public int najviacOslavencov()` - vráti poradové číslo mesiaca (1-12), v ktorom oslavuje najviac obyvateľov narodeniny (6 bodov).
- `public Set<Integer> dobreCasy()` - vráti množinu rokov, v ktorých sa narodilo viac dobrých ako zlých obyvateľov (6 bodov).
- `public List<String> nezastihnutelneTypy()` - vráti zoznam pomenovaní (bez opakovania) takých typov, že neexistuje obyvateľ daného typu, ktorý by mal telefón. (5 bodov).
- `public String najlepsiTyp()` - vráti názov takého typu obyvateľov v zozname, medzi ktorými je najväčšie percento dobrých; napr. ak medzi drakmi je len 20% dobrých, zatiaľ čo 90% trpaslíkov je dobrých, vrátili by sme „trpaslík“. (7 bodov).
- `public List<Obyvatel> zastupcoviaLudu()` – vráti referenciu na novovytvorený zoznam obyvateľov tvorený po jednom zástupcovi z každého typu. Za zástupcu daného typu vyberáme najstaršieho obyvateľa tohto typu (5 bodov).

- `public String najrozmanitejsiTyp()` - vráti názov takého typu obyvateľov, že obyvatelia tohto typu spoločne disponujú najväčším počtom rôznych čarovných schopností (9 bodov).
- `public boolean bezkonfliktnyZoznam(File konflikty)` – vráti, či obyvatelia v zozname sú navzájom bezkonfliktní na základe popisu dvojíc konfliktných typov v textovom súbore. Čo sú to dvojice konfliktných typov? Napr. ak je trpaslík a obor na jednom evente, ľahko sa môže stať, že obor zašliapne trpaslíka. Nuž a to je zárodok konfliktu. Preto typ trpaslík a typ obor tvoria dvojicu navzájom konfliktných typov. Preto, ak sa v zozname obyvateľov vyskytne aj obyvateľ typu trpaslík aj obyvateľ typu obor, je tento zoznam konfliktný. Takýchto dvojíc konfliktných typov je v kráľovstve viacero.
  Formát textového súboru s popisom dvojíc konfliktných typov si zvoľte podľa uváženia. Napríklad jeden riadok môže obsahovať tabulátorom oddelenú jednu dvojicu pomenovaní typov obyvateľov, medzi ktorými môže vzniknúť konflikt, napr. trpaslík `\t` obor (10 bodov).

Triedenie a komparátor (dokopy 8 bodov):

Niektoré podujatia sú vhodné len pre určitý vek a preto je potrebné zoznam hostí zostaviť veľmi starostlivo. Vytvorte preto metódu, ktorá zoradí obyvateľov v zozname podľa veku.

Vytvorte triedu `VekovyKomparator` implementujúcu `java.util.Comparator<Obyvatel>` s metódou (6 bodov):

- `public int compare(Obyvatel o1, Obyvatel o2)` – porovná obyvateľov podľa veku.

V triede `ZoznamObyvatelov` implementujte inštančnú metódu (2 body):

- `public void zoradPodlaVeku()`– usporiada obyvateľov v zozname podľa veku počnúc najmladším obyvateľom.

Výnimky (3 body)

Vytvorte nekontrolovanú výnimku `NeznamyTypException` a vhodne ju použite aspoň v jednej metóde.