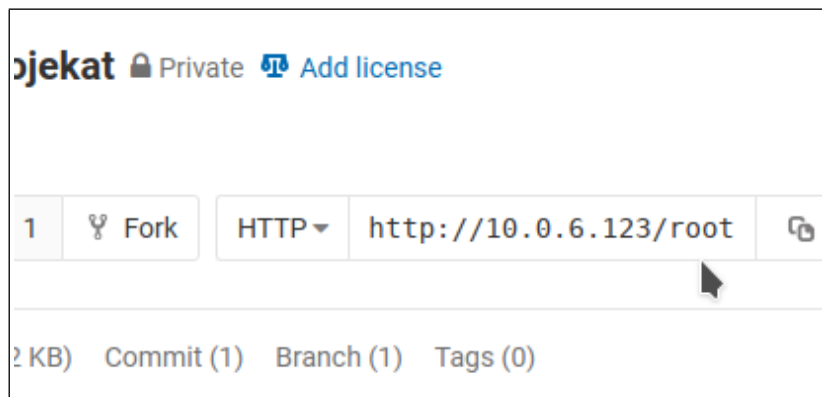


Parcijalni ispit - Upute za rad

1. Upalite okruženje IntelliJ IDEA.
2. Pristupite web baziranom okruženju GitLab na adresi:
<http://10.0.6.123>
3. Kliknite na opciju **Sign In** kako biste kreirali novi korisnički račun.
4. Unesite vaše pravo ime i prezime, vašu pravu ETF email adresu, za korisničko ime uzmite prvi dio email adrese prije znaka @ (npr. mbajraktar1), izaberite neki password koji nije lako pogoditi.
5. Tačno 15 minuta nakon početka ispita dobićete tekst zadatka.
6. Refreshujte GitLab stranicu: ugledaćete repozitorij koji se zove **RPR I parcijalni - prvi termin** (ili **drugi termin**). Otvorite taj repozitorij i kliknite na opciju **Fork**.
7. Sada se vratite na naslovnu stranicu, vidjećete dva repozitorija: jedan je onaj koji ste upravo forkali (Administrator / RPR I parcijalni), a drugi je vaša kopija/fork (Vaše Ime / RPR I parcijalni). Uđite u taj drugi repozitorij.
8. U polju HTTP nalazi se URL vašeg projekta. Možete ga iskoristiti u IDEA da preuzmete projekat sa Git servera kako smo radili na tutorijalima.



9. Radite samostalno projekat i nemojte zaboraviti raditi commit i push! Biće pregledana posljednja verzija koju push-ate.
10. Nemojte gledati ekran od kolega/kolegica pored vas i raditi isto! Zabilježili smo ko je gdje sjedio, poslije ispita biće korišten poznati sistem za provjeru prepisivanja i bićete bodovani sa 0 bodova ako se vaš kod bude neznatno razlikovao od osoba pored vas.

Sarajevo, 24. 11. 2018

Vedran Zuborić

I parcijalni ispit, varijanta A (12:00)

Ukupno bodova: 20 (svaki test nosi 0,5 bodova, što može biti umanjeno zbog hardcodiranja, loše osmišljene strukture, te nepridržavanja pravila pisanja kvalitetnog koda datih na predavanjima i tutorijalima)

Potrebno je razviti aplikaciju koja omogućuje praćenje performansi rada neke fabrike. Fabrika sadrži određeni broj mašina, pri čemu svaka mašina može proizvesti jedan od nekoliko vrsta materijala određenom brzinom. Osnovni podaci o mašini su naziv i serijski broj, a zatim spisak svih materijala koje mašina može proizvesti (ovdje treba uzeti u vidu da nemaju sve mašine isti broj materijala koje mogu proizvesti). Za svaki materijal navodi se koliko sati treba datoj mašini da ga proizvede, što smatramo za *cijenu* tog materijala (jer vrijeme.equals(novac)).

Fabrika koristi mašine strane proizvodnje (kupljene), i mašine koje su u fabrici proizveli za svoje potrebe, koje rade brže. Razliku između domaćih i stranih mašina trebate predstaviti nasljeđivanjem, pri čemu naziv klase nije bitan (ne testira se).

Naziv mašine se sastoji isključivo od velikih i malih slova abecede, sa ograničenjem na dužinu koja mora biti minimalno 2 karaktera. Serijski broj mašine je pozitivan cijeli broj sa najviše 5 cifara ($0 < \text{serijski} < 100000$). Sve mašine su ograničene od strane proizvođača na 8 radnih sati dnevno.

Mašine podržavaju sljedeće operacije:

- **int getSerijski()** - daje serijski broj.
- **void upali()** - metoda za paljenje mašine, baca izuzetak (korisnički definisani) tipa WrongMachineState ako je već upaljena. Pri paljenju mašina dobiva 8 radnih sati za proizvodnju.
- **void ugasi()** - gašenje mašine, baca izuzetak tipa WrongMachineState ako je već ugašena.
- **void resetuj()** - gašenje i paljenje mašine, resetuje broj sati na 8, baca izuzetak tipa WrongMachineState ako je ugašena.
- **int cijena(String materijal)** – vraća broj sati koliko košta proizvodnja datog materijala, baca izuzetak tipa IllegalArgumentException ako se materijal ne može proizvesti.
- **int proizvedi(String materijal)** – proizvodi navedeni materijal, vraća broj sati koliko je proizvodnja trajala, baca izuzetak tipa IllegalArgumentException ako mašina ne može proizvesti navedeni materijal ili ako nema dovoljno sati na raspolaganju da se dati materijal proizvede.
- **int preostaloSati()** - vraća broj sati koliko ova mašina još može raditi, vraća 0 ako je mašina ugašena.
- **void registrujMaterijal(String naziv, int cijena)** - registruje da data mašina može proizvesti navedeni materijal pod određenom cijenom datom u satima.
- **Set<String> dajMaterijaleMogućeZaProizvesti()** - vraća nazive svih materijala koje ova mašina može proizvesti u preostalom vremenu, abecedno sortirano.

- **Map<String, Integer> dajMogucnostProizvodnje()** - vraća novu mapu čiji je ključ tipa string i predstavlja ime materijala, a vrijednost je broj komada koliko se može proizvesti ako se ostatak vremena posveti samo proizvodnji ovog materijala.

Mašina domaće proizvodnje pored navedenih operacija ima još sljedeće operacije:

- **void pocniBrziRad()** - prebacuje mašinu u režim brzog rada, baca izuzetak tipa WrongMachineState ukoliko već jeste u tom režimu. U ovom režimu se sve vremenske cijene smanjuju za 2, s tim da ne mogu ići ispod 1
- **void zaustaviBrziRad()** - vraća mašinu u standardni režim rada, baca izuzetak tipa WrongMachineState ukoliko nije bila u režimu brzog rada

Kod domaćih mašina se, kod svih metoda koje uzimaju u obzir trajanje, uzima modificirana vrijednost u zavisnosti od režima rada u kojem se nalazi.

Materijal predstavljaju dva podatka: naziv (String) materijala i cijena (u satima, od 1 do 5).

Pri registraciji mašina i materijala trebaju se automatski izbaciti duplikati. Kao duplikat smatramo mašinu sa istim serijskim brojem, odnosno mašine sa istim serijskim, neovisno o nazivu i ostalim atributima. Ako unesena **mašina** već postoji, podrazumijeva se da se prethodni unos "pregazi", čime posljednje unesene vrijednosti postaju aktuelne. Međutim, ako uneseni **materijal** već postoji, odbacuje se proslijeđeni podatak i ostaje ranije pohranjeni podatak!

Kada se mašina ispiše (npr. System.out.println(m) gdje je m instanca klase Masina), potrebno je na izlazu da se ispiše:

"Masina <naziv> je upaljena (preostalo <N> sati)" ili "Masina <naziv> je ugasena", zavisno da li je upaljena ili ne. Nakon toga slijedi ispis:

"Ona može proizvesti materijale <spisak>", gdje je <spisak> materijala u obliku "naziv (cijena)", i abecedno su sortirani po nazivu, a razdvojeni su znakom zarez (na kraju se nalazi tačka).

Kada se fabrika ispiše (npr System.out.println(f) gdje je f instanca klase Fabrika), potrebno je da se ispiše spisak svih mašina, uz redni broj ispred, npr.

1. Masina masina1 je ugasena. Ona može proizvesti materijale m1 (1), m5 (5).
2. Masina masina2 je upaljena (preostalo 5 sati). Ona može proizvesti.... (itd.)

Konačno, potrebno je u klasi Fabrika implementirati sljedeće metode:


- **Map<Masina, String> najviseProizvoda()** - vraća mapu gdje je ključ mašina, a vrijednost naziv materijala kojeg ta mašina može najviše proizvesti danas. U mapi se izostavljaju ugašene mašine. Mašine koje nemaju registrovan nijedan materijal treba također izostaviti. Ukoliko ima više materijala koje za neku mašinu ispunjavaju navedene uslove, uzima se onaj materijal koji dolazi prvi po abecedi. (npr ako mašina M može proizvesti i materijal m1 i m2 8 puta danas, uzima se m1).
- **Masina dodajDomacuMasinu(String naziv, int serijski)** - registruje datu mašinu u fabriku i vraća kreiranu instancu mašine. Pri registraciji mašina je ugašena.
- **Masina dodajKupljenuMasinu(String naziv, int serijski)** - registruje datu mašinu u fabriku i vraća kreiranu instancu mašine. Pri registraciji mašina je ugašena.
- **void dodajMaterijal(String nazivMasine, String nazivMaterijala, int cijena)** - registruje dati materijal u mašinu sa proslijeđenim nazivom. Baca izuzetak tipa IllegalArgumentException ukoliko ne postoji. Ukoliko ima više mašina sa istim nazivom,

materijal se dodaje u sve mašine sa navedenim nazivom, neovisno o tome da li je upaljena ili ne.

- **Map<Masina, Integer> cijenaZaMaterijal(String naziv)** – vraća mapu gdje je ključ mašina a vrijednost cijena navedenog materijala, ukoliko se dati materijal proizvodi na toj mašini. Ukoliko mašina nije u stanju da proizvede dati materijal mapa će sadržati -1, a ukoliko je mašina ugašena neće se pojaviti unutar ove mape.
- **Set<Masina> dajMasine()** - vraća set mašina sortiran po upotrebljivosti mašine za proizvodnju. Kriterij za upotrebljivost, tj. da je mašina A korisnija od mašine B, je da ima veći izbor materijala koji se mogu proizvesti sa preostalim satima. Ukoliko ovaj kriterij zaključuje da su dvije mašine jednake, vrši se dalji poredak po nazivu (abecedno, što je i defaultni poredak za String klasu). Sortira se od najmanje vrijednosti do najveće (najmanji broj materijala na početak, odnosno rastući poredak).

Funkcija ima i jedan argument **filter** koji predstavlja pokazivač na funkciju. Ukoliko je vrijednost ovog argumenta **null**, uzimaju se sve mašine u obzir. Ukoliko vrijednost ovog argumenta nije **null**, uzimaju se samo one mašine u obzir za koji poziv prosljeđene funkcije vraća vrijednost **true**.

Sarajevo, 24. 11. 2018

Handwritten signature in blue ink that reads "Vedran Zuborić".

I parcijalni ispit, varijanta B (14:00)

Ukupno bodova: 20 (svaki test nosi 0,5 bodova, što može biti umanjeno zbog hardcodiranja, loše osmišljene strukture, te nepridržavanja pravila pisanja kvalitetnog koda datih na predavanjima i tutorijalima)

Za potrebe praćenja proizvodnje lijekova u apoteci, potrebno je razviti sistem koji omogućuje evidenciju proizvodnje. Glavni alat u proizvodnji lijekova je aparat za doziranje koje vrlo precizno izračunava količinu određenog sastojka koji je potreban za proizvodnju lijeka. Potrebno je prije svega unijeti osnovne podatke (šifra, serijski broj), a zatim spisak svih sastojaka koje dati aparat može dozirati (ovdje treba uzeti u vidu da se svi aparati ne mogu koristiti za doziranje svih sastojaka, nego su prilagođeni za određene sastojke (tečne, praškaste itd.)).

Apoteka koristi digitalne i analogne aparate za doziranje.

Šifra aparata se sastoji isključivo od velikih i malih slova abecede, sa ograničenjem na dužinu koja mora biti minimalno 4 karaktera. Serijski broj aparata je pozitivan cijeli broj sa tačno 6 cifara ($99999 < \text{serijski} < 1000000$). Sve aparati imaju maksimalnu količinu rada koje mogu obaviti u jednom danu koja se označava *kreditima*. Maksimalan broj kredita za bilo koji aparat je 100.

Svi aparati podržavaju sljedeće operacije:

- **int getSerijski()** - daje serijski broj.
- **void upali()** - metoda za paljenje aparata, baca izuzetak (korisnički definisani) tipa WrongDeviceState ako je već upaljen. Pri paljenju aparat dobiva 100 kredita za rad.
- **void ugasi()** - gašenje aparata, baca izuzetak tipa WrongDeviceState ako je već ugašen.
- **void fabrickiReset()** - gašenje i paljenje aparata, resetuje broj kredita na 100, baca izuzetak tipa WrongDeviceState ako je ugašen.
- **int cijena(String sastojak)** – vraća broj kredita koliko košta doziranje datog sastojka, baca izuzetak tipa IllegalArgumentException ako se sastojak ne može dozirati na datom aparatu.
- **int doziraj(String sastojak)** – dozira navedeni sastojak, vraća broj kredita koliko je bilo potrebno za doziranje, baca izuzetak tipa IllegalArgumentException ako aparat ne može dozirati navedeni sastojak ili ako nema dovoljno kredita na raspolaganju da se dozira dati sastojak.
- **int preostaloKredita()** - vraća broj kredita koliko ovaj aparat još može raditi, vraća 0 ako je aparat ugašen.
- **void registrujSastojak(String naziv, int cijena)** - registruje da dati aparat može koristiti za doziranje navedenog sastojka pod određenom cijenom datom u kreditima.
- **Set<String> dajPodrzaneSastojke()** - vraća nazive svih sastojaka koje ovaj aparat može dozirati sa preostalim brojem kredita, abecedno sortirano.

- **Map<String, Integer> dajMogucnostDoziranja()** - vraća novu mapu čiji je ključ tipa string i predstavlja ime sastojka, a vrijednost je količina koliko se može dozirati ako se ostatak vremena posveti samo doziranju ovog sastojka.

Digitalni aparat za doziranje pored navedenih operacija ima još sljedeće operacije:

- **void primijeniNoviCjenovnik()** - postavlja aparat da koristi novi cjenovnik prilikom doziranja, baca izuzetak tipa WrongDeviceState ukoliko već jeste u tom režimu. U ovom režimu se svi sastojci se doziraju za 2 kredita manje, s tim da količina kredita ne može ići ispod 1.
- **void primijeniStariCjenovnik()** - vraća aparat u standardni režim rada, baca izuzetak tipa WrongDeviceState ukoliko nije bio u režimu novog cjenovnika.

Kod digitalnih aparata se, kod svih metoda koje uzimaju u obzir broj kredita, uzima modificirana vrijednost u zavisnosti od režima rada u kojem se nalazi.

Sastojak predstavljaju dva podatka: naziv (String) sastojka i cijena (u kreditima, od 1 do 80).

Pri registraciji aparata i sastojaka trebaju se automatski izbaciti duplikati. Kao duplikat smatramo aparat sa istim serijskim brojem, odnosno aparate sa istim serijskim, neovisno o šifri i ostalim atributima. Ako uneseni **aparat** već postoji, podrazumijeva se da se prethodni unos "pregazi", čime posljednje unesene vrijednosti postaju aktuelne. Međutim, ako uneseni **sastojak** već postoji, odbacuje se prosljeđeni podatak i ostaje ranije pohranjeni podatak!

Kada se aparat ispiše (npr. System.out.println(a) gdje je a instanca klase Aparat), potrebno je na izlazu da se ispiše:

"Aparat <naziv> je upaljen (preostalo <N> kredita)" ili "Aparat <naziv> je ugašen", zavisno da li je upaljena ili ne. Nakon toga slijed ispis:

"On može dozirati sastojke <spisak>", gdje je <spisak> sastojaka u obliku "naziv (cijena)", i abecedno su sortirani po nazivu, a razdvojeni su znakom zarez (na kraju se nalazi tačka).

Kada se apoteka ispiše (npr System.out.println(a) gdje je a instanca klase Apoteka), potrebno je da se ispiše spisak svih aparata, uz redni broj ispred, npr.

1. Aparat a1 je ugašen. On može dozirati sastojke s1 (1), s5 (5).

2. Aparat a2 je upaljen (preostalo 50 kredita). On može dozirati.... (itd.)

Konačno, potrebno je u klasi Apoteka implementirati sljedeće metode:


- **Map<Aparat, String> najviseSastojaka()** - vraća mapu gdje je ključ aparat, a vrijednost naziv sastojka kojeg taj aparat može najviše dozirati danas. U mapi se izostavljaju ugašeni aparati. Aparate koje nemaju registrovan nijedan sastojak treba također izostaviti. Ukoliko ima više sastojaka koji za neki aparat ispunjavaju navedene uslove, uzima se onaj sastojak koji dolazi prvi po abecedi. (npr ako aparat A može dozirati i sastojak s1 i s2 8 puta danas, uzima se s1).
- **Aparat dodajDigitalniAparat(String naziv, int serijski)** - registruje dati aparat u apoteku i vraća kreiranu instancu aparata. Pri registraciji aparat je ugašen.
- **Aparat dodajAnalogniAparat(String naziv, int serijski)** - registruje dati aparat u apoteku i vraća kreiranu instancu aparata. Pri registraciji aparat je ugašen.
- **void dodajSastojak(String sifraAparata, String nazivSastojka, int cijena)** - registruje dati sastojak u aparat sa prosljeđenom šifrom. Baca izuzetak tipa

IllegalArgumentException ukoliko ne postoji. Ukoliko ima više aparata sa istom šifrom, sastojak se dodaje u sve aparate sa navedenom šifrom, neovisno o tome da li je upaljen ili ne.

- **Map<Aparat, Integer> cijenaZaSastojak(String naziv)** – vraća mapu gdje je ključ aparat a vrijednost cijena navedenog sastojka, ukoliko se dati sastojak može dozirati na tom aparatu. Ukoliko aparat nije u stanju da dozira dati sastojak, mapa će sadržati -1, a ukoliko je aparat ugašen neće se pojaviti unutar ove mape.
- **Set<Aparat> dajAparate()** - vraća set aparata sortiran po upotrebljivosti aparata za doziranje. Kriterij za upotrebljivost, tj. da je aparat A korisniji od aparata B, je da ima veći izbor sastojaka koji se mogu dozirati sa preostalim kreditima. Ukoliko ovaj kriterij zaključuje da su dva aparata jednaka, vrši se dalji poredak po šifri aparata (abecedno, što je i defaultni poredak za String klasu). Sortira se od najmanje vrijednosti do najveće (najmanji broj materijala na početak, odnosno rastući poredak).

Funkcija ima i jedan argument **filter** koji predstavlja pokazivač na funkciju. Ukoliko je vrijednost ovog argumenta **null**, uzimaju se svi aparati u obzir. Ukoliko vrijednost ovog argumenta nije **null**, uzimaju se samo oni aparati u obzir za koji poziv prosljeđene funkcije vraća vrijednost **true**.

Sarajevo, 24. 11. 2018

Handwritten signature in blue ink that reads "Vedran Zuborić".