Ülesande tutvustus

Ülesanne on kirjutada programm arvutiparandustöökojale.

Tööpäeva alguses käivitatakse programm käsurea argumendiga, mis viitab parandamist ootavate arvutite nimekirjale. Programm loeb ootel olevate arvutite nimekirja sisse ja alustab töötsükliga. Töötsükli igal kordusel küsitakse kasutajalt, kas ta tahab parandada (`P`), uut tööd registreerida (`R`) või lõpetada (`L`).

Näide

Kasutamise sessioon võiks näha välja midagi sellist:

```
Rida: Ordi@2017-04-21T14:00:23 Vea selgitus: Väljade arv on vale!

Kas soovid parandada (P), uut tööd registreerida (R) või lõpetada (L) ? P
Arvuti info: Lenovo;kiirtöö@2017-04-21T14:22:42
Sisesta parandamiseks kulunud aeg (täisminutites): 142
Sisesta enda nimi: Jaan
Töö tehtud, arve summa on 59.33 €!

Kas soovid parandada (P), uut tööd registreerida (R) või lõpetada (L) ? R
Sisesta töö kirjeldus: Dell;tavatöö;mnitoriga
Vigane sisestus! Kolmanda välja väärtus ei ole "monitoriga"!
Sisesta töö kirjeldus: Dell;tavatöö;monitoriga
Töö on registreeritud!

Kas soovid parandada (P), uut tööd registreerida (R) või lõpetada (L) ? P
Arvuti info: ML;kiirtöö;monitoriga@2017-04-21T14:08:09
Sisesta parandamiseks kulunud aeg (täisminutites): 42
Sisesta enda nimi: Peeter
Töö tehtud, arve summa on 36.45 €!

Kas soovid parandada (P), uut tööd registreerida (R) või lõpetada (L) ? P
Arvuti info: Dell;tavatöö;monitoriga@2017-04-20T13:08:30.080
Sisesta parandamiseks kulunud aeg (täisminutites): 75
Sisesta enda nimi: Peeter
Töö tehtud, arve summa on 44.88 €!

Kas soovid parandada (P), uut tööd registreerida (R) või lõpetada (L) ? L
Sessiooni kokkuvõte:
Teenitud raha: 140.66€
Parandatud arvuteid:
  Lenovo: 1tk
  Dell: 1tk
  ML: 1tk
Ootele jäi 2 arvuti(t).
```

Peaklass

Programmi peaklass peab olema nimega `Arvutiparandus` ja asuma vaikepaketis.

Klassid `Arvuti` ja `VäliseMonitorigaArvuti`.

Tööde kirjelduste hoidmiseks programmi käigus tuleb koostada vaikepaketti klassid `Arvuti` ja `VäliseMonitorigaArvuti`. Klass `VäliseMonitorigaArvuti` on klassi `Arvuti` alamklass. Ülemklassis olemasolevaid isendivälju ja meetodeid alamklassis uuesti mitte kirjeldada. Mõlema klassi isenditel peavad olema järgnevad meetodid töö kohta käiva info küsimiseks:

- `String getTootja()`
- `boolean onKiirtöö()`
- `java.time.LocalDateTime getRegistreerimiseAeg()`

Veel peab olema meetod `arvutaArveSumma`, mis võtab argumendiks baashinna (`double`-tüüpi), ning tagastab töö lõpphinna (hinna määramise reeglid on allpool).

NB! Nende meetodite nähtavus ei tohiks olla piiratud, st. neid peaks saama välja kutsuda suvalisest klassist.

Tööde nimekirja haldamine

Esialgne tööde nimekiri

Viide esialgsele ootel tööde nimekirjale antakse programmile ette käsureal ja see võib olla kas failinimi või veebiaadress. Võib eeldada, et kui (esimene) käsureaargument algab `http://`-ga või `https://`-ga, siis on tegemist veebiaadressiga, vastasel juhul on tegemist failinimega. Nii veebis kui failis on andmed tekstilisel kujul kodeeringus *UTF-8*.

Mõlemal juhul koosneb tekst ridadest, kus iga rida tähistab ühe töö andmeid. Reas on semikooloniga eraldatuna kirjas arvuti tootja, tellimuse tüüp ("kiirtöö" või "tavatöö"). Kui tegemist on välise monitoriga arvutiga, siis järgneb veel `;monitoriga`. Peale neid andmeid tuleb **@** ja seejärel töö registreerimise aeg.

Näited

- `Lenovo;tavatöö@2017-03-25T12:34:12`
- `Ordi;kiirtöö;monitoriga@2017-04-12T10:12:45`

Iga failis oleva rea põhjal tuleb ootel tööde nimekirja genereerida õige klassiga arvuti (meetodi `loeArvuti` kirjeldus on allpool). Kõik read, millelt pole võimalik töö infot välja lugeda, tuleb vea selgitusega ekraanile väljastada.

Kui veebist või failist lugemisel tekib erind (nt faili ei leitud või veebist lugemisel tekkis tõrge), mis pole faili sisu töötlemisega seotud, siis peab laskma sellel erindil programmi töö katkestada.

Abimeetod `Arvutiparandus.loeArvuti`

Peaklassis peab olema abimeetod `loeArvuti` (võib olla staatiline), mis võtab argumendiks sõne töö kirjeldusega (selles formaadis, nagu ülalpool kirjeldatud) ja tagastab sellele vastava `Arvuti` või `VäliseMonitorigaArvuti`. Kui registreerimise aeg ja sellele eelnev @ puuduvad, siis peab tagastatava objekti `getRegistreerimiseAeg()` tagastama aja, millal sõne töötlemist alustati.

Kui etteantud sõne ei vasta nõutud formaadile, peab `loeArvuti` meetod erindi viskama. Defineeri erindiklass `FormaadiErind`. Visatud erind peab sisaldama vea põhjust seletavat sõnumit. `FormaadiErindi` peab viskama järgnevate veatingimuste puhul (võib eeldada, et sisendis teistsuguseid vigu ei esine):

- semikoolonitega eraldatud väljade arv on väiksem, kui kaks, või suurem, kui kolm
- töö tüübi väljas olev väärtus ei ole "tavatöö" ega "kiirtöö"
- väljade arv on kolm, aga kolmanda välja väärtus ei ole "monitoriga"

NB! Kuna `loeArvuti` puhul on tegemist abimeetodiga, mida läheb vaja vaid peaklassis, siis ei tohiks olla võimalik seda teistest klassidest välja kutsuda.

Vihje: LocalDateTime#parse

Vihje: LocalDateTime#now

Vihje (vaata ainult siis kui jääd hätta)

Uue töö registreerimine

Kui kasutaja valib uue töö registreerimise, siis laseb programm sisestada tal töö kirjelduse (samas formaadis nagu algsete tööde puhul, aga ilma kuupäeva ja sellele eelneva @-ta) ning salvestab sellele vastava objekti (`VäliseMonitorigaArvuti` või `Arvuti`) ootel tööde nimekirja.

Kui sisestatud kirjeldus on vigane, siis tuleb küsimist korrata niikaua, kuni kasutaja sisestab õige formaadiga kirjelduse. Iga vigase sisestuse korral väljastada vea kirjeldus.

Arvuti parandamine

Kui kasutaja valib parandamise, siis tuleb võtta ootel tööde hulgast ükskõik milline kiirtellimus või kui kiirtellimusi (enam) pole, siis ükskõik milline tavatellimus. Programm näitab tellimuse kirjelduse kasutajale ning jääb ootama, kuni kasutaja sisestab töö tegemiseks kulunud aja (täisarvuna, minutites) ja oma nime. Peale seda tuleb määrata tööle hind ja sellega loetakse antud töö tehtuks.

Tööle hinna määramine

Baashind arvutatakse parandamiseks kulunud aja ning parandaja tunnitasu põhjal. Lõpphinna saamiseks lisatakse baashinnale fikseeritud summa töö vastuvõtmise eest (2€ arvuti eest ja täiendav +1€ välise monitori puhul) ning lisaks veel 10€ kiirtellimuse puhul.

Parandajate tunnitasud on kirjas failis *tunnitasud.dat*. See on binaarne fail, moodustatud klassi `java.io.DataOutputStream` abil, kus faili algul on töötajate arv (`int`), ning seejärel iga töötaja kohta tema nimi (`String`) ning tunnitasu (`double`), st. faili sisu on kujul:

- töötajate arv
- esimese töötaja nimi
- esimese töötaja tunnitasu
- ...
- viimase töötaja nimi

- viimase töötaja tunnitasu

Tehtud ja tegemata tööde salvestamine

Programmi lõpus tuleb tehtud tööd salvestada faili *tehtud.dat* kasutades selleks klassi `java.io.DataOutputStream`. Fail peab olema järgnevas formaadis: parandatud arvutite arv (`int`), millele järgneb iga arvuti kohta selle tootja (`String`), registreerimise aeg (`String`, kasuta `LocalDateTime` meetodit `toString`) ja arve summa (`double`). NB! Kui sellise nimega fail juba eksisteerib, siis kirjuta selle sisu üle.

Tegemata tööd tuleb salvestada tekstifaili nimega *ootel.txt* (kodeeringus *UTF-8*, samas formaadis nagu kasutati esialgsete ootel tööde puhul).

Tööde kokkuvõtte kuvamine

Tööde kokkuvõtteks tuleb programmi lõpus ekraanile kuvada antud sessioonis teenitud raha ning tootjate kaupa, mitu arvutit parandati. Lisaks tuleb näidata, mitu arvutit jäi ootele.

Vihje (vaata ainult siis kui jääd hätta)

Andmefailid

Soovitame testimiseks vajalikud andmefailid ise koostada. Mõtle seejuures, millised kirjed peaks seal olema, et võimalikud vead programmis avalikuks tuleksid!

Kui jääd andmefailide koostamisega hätta, siis võta failid siit: arvutiparanduse_naidisfailid.zip.

Veebist lugemise testimiseks võid kasutada seda linki: https://courses.cs.ut.ee/2017/OOP/spring/uploads/Main/ootel_arvutid.txt

Pehmed nõuded

- Tekstifailide lugemiseks ja kirjutamiseks peab juba põhjalikult läbi harjutatud `Scanner` ja `PrintWriter` asemel kasutama värskelt õpitud voogude klasse, mis on mõeldud tekstiga töötamiseks!
- Kood peaks olema võimalikult kergesti loetav!
  - Võimalusel eralda alamprobleemid omaette meetoditesse!
- Kui märkad end koodi *copy-paste*mas, siis mõtle, kas saaksid mõne abimeetodi sissetoomisega seda vältida!
- Kontrolli, kas oled otstarbekalt kasutanud pärilust, polümorfismi ja kõige sobivamaid andmestruktuure!
- Kõik väljad peavad olema privaatsed ja mittestaatilised!

Mittekompileeruva programmi eest punkte ei saa.