26.03 Kontrolltöö 1 aines *Objektorienteeritud programmeerimine*

Käivitage ekraanisalvesti, vt Moodle Kontrolltöö reeglid. Kontrolltöö ajal on Moodle's BBB liin, kus saate küsimusi esitada.

Kõik klassid peavad asuma vaikepaketis (st failide alguses ei ole `package` direktiivi) ja failide kodeering on UTF-8.

Kohvik pakub klientidele erinevaid jooke (nii pudelis kui ka vaadist). Joogid on esitatud failis järgmisel kujul:

```
mahl,1.00,250,2
vaadikali,0.35,500
limonaad,1.60,330,3
vesi,1.60,330,1
siider,0.50,350
```

Iga rida algab joogi nimetusega. Nimetusele järgneb hind ja maht. Kui tegu on pudelijoogiga, siis nimetusele, hinnale ja mahule järgneb tellitud pudelite arv. Kui tegu on pudelijoogiga, siis hinnaks on pudeli hind ja mahuks on ühe pudeli maht milliliitrites. Kui tegu on vaadijoogiga, siis hinnaks on 100 ml hind ja mahuks on tellitud maht milliliitrites. Eraldajaks on koma (tühikut ei ole).

Kontrolltöö seisneb jookide tellimusi käsitleva programmi koostamises. Programm peab vastama alltoodud nõuetele.

Programm peab sisaldama klasse `Jook`, `Pudelijook`, `Vaadijook`, `Tellimus` ning peaklassi. Peaklassis loetakse sisse joogid ja koostatakse tellimused. Peaklassis testitakse ka erinevate isendimeetodite tööd. Kõikide klasside kõik isendiväljad peavad olema privaatsed.

1. (4 p) Abstraktses klassis `Jook` peavad olema privaatsed isendiväljad joogi nimetuse (`String`), hinna (`double`) ja mahu (`int`) jaoks.
    1. Klassis peab olema kolme parameetriga konstruktor isendiväljade määramiseks.
    2. Klassis peavad olema `get`-meetodid isendiväljade jaoks (kui neid on vaja).
    3. Klassis peab olema abstraktne parameetriteta `double`-tüüpi meetod `liitriHind`.
    4. Klassis peab olema abstraktne parameetriteta `double`-tüüpi meetod `tellimuseHind`.
    5. Klassis peab olema ka meetod `toString` joogi info tekstina esitamiseks, näidates lisaks joogi nimetusele, hinnale ja mahule ka liitri ja tellimuse hinda.
    6. Klass `Jook` peab realiseerima liidese `Comparable<Jook>`, kusjuures `compareTo` meetod realiseeritakse nii, et võrdlemine toimub liitri hinna alusel.

2. (3 p) Klass `Pudelijook` on klassi `Jook` alamklass. Ülemklassis olemasolevaid isendivälju siin uuesti mitte kirjeldada. Lisaks peab olema privaatne isendiväli tellitud pudelite arvu (`int`) jaoks.
    1. Klassis peab olema nelja parameetriga konstruktor, mille abil saab määrata joogi nimetuse, hinna, mahu ja tellitud pudelite arvu.
    2. Klassis peab olema meetod `liitriHind`, mis arvutab liitri hinna kasutades valemit *1000*pudeli_hind/pudeli_maht*.

3. Klassis peab olema meetod `tellimuseHind`, mis arvutab tellimuse hinna kasutades valemit *pudeli_hind*pudelite_arv*.
4. Klassis peab olema ka meetod `toString` pudelijoogi info tekstina esitamiseks, mille ülekatmisel on rakendatud ülemklassi meetodit `toString` lisades ka tellitud pudelite arvu.

3. (2 p) Klass `Vaadijook` on klassi `Jook` alamklass. Ülemklassis olemasolevaid isendivälju siin uuesti mitte kirjeldada.

1. Klassis peab olema kolme parameetriga konstruktor, mille abil saab määrata joogi nimetuse, hinna ja mahu.
2. Klassis peab olema meetod `liitriHind`, mis arvutab liitri hinna kasutades valemit *10*hind*.
3. Klassis peab olema meetod `tellimuseHind`, mis arvutab tellimuse hinna kasutades valemit *hind*maht/100*.
4. Klassis peab olema ka meetod `toString` pudelijoogi info tekstina esitamiseks, mille ülekatmisel on rakendatud ülemklassi meetodit `toString` lisades ka teksti ”Vaadijook”.

4. (3 p) Klassis `Tellimus` peab olema privaatne isendiväli tellitud jookide nimekirja (`List<Jook>`) jaoks.

1. Klassis peab olema parameetriteta konstruktor.
2. Äsjaloodud tellimuses ei olegi ühtegi jooki. Jookide lisamiseks peab olema `void`-tüüpi meetod `telliJook`, mis jätab argumendiks antud `Jook`-tüüpi isendi meelde.
3. Klassis peab olema `double`-tüüpi parameetriteta meetod `tellimuseMaksumus`, mis tagastab tellimuse kõikide jookide kogumaksumuse.
4. Klassis peab olema `void`-tüüpi parameetriteta meetod `tellimuseJoogid`, kus tellimusse salvestatud joogid väljastatakse ekraanile nii, et iga jook on eraldi real.
5. Klassis peab olema ka meetod `toString` tellimuse maksumuse ja erinevate jookide arvu tekstina esitamiseks (tellitud pudelite arvuga ei ole vaja arvestada) .

5. (4 p) Peaklass peab olema nimega `Peaklass`. Klassis peab olema staatiline avalik meetod `loeJoogid` tagastustüübiga `List<Jook>`, mis võtab argumendiks failinime ja tagastab selles failis olevad jookide andmed. Meetod võib visata erindi (st meetodi signatuuris võib olla `throws Exception`). Jookide faili formaat on ülalpool toodud. Jookide arv failis ei ole teada (programm peaks töötama suvalise arvu jookidega). Kui failist lugemist ei õnnestu programmeerida, siis kirjutage selles meetodis vastav list programmi sisse (vähendab tulemust 2 punkti võrra).

Peameetodis tehakse järgmised tegevused.

1. Rakendatakse vastavat staatilist meetodit, et lugeda failist jookide andmed.
2. Sorteeritakse joogid vastavalt meetodis `compareTo` kirjeldatud järjekorrale (kasutada meetodit `Collections.sort`).
3. Jookide info väljastatakse ekraanile.
4. Luuakse 5 tellimust.
5. Kõikidest tellimustest tehakse `Tellimus`[]-tüüpi massiiv. (Massiivi võib ka enne tellimuste tegemist luua ja järjest täita.)

6. Iga tellimuse jaoks valitakse juhuslikult 1 kuni 4 jooki (võib eeldada, et failis on vähemalt 4 jooki). Selleks tuleb kasutada `Collections.shuffle` meetodit. Antud meetod võtab argumendiks listi ning järjestab selle suvalises järjekorras. Jookide list järjestada iga tellimuse jaoks uuesti ümber, jookide arv *n* genereerida juhuslikult lõigust [1, 4] (täisarvuna) ning lisada tellimusse *n* esimest jooki.
7. Tellimuste info ja iga tellimuse jookide nimekiri väljastatakse ekraanile.

Programmi väljund peab olema arusaadav ja loetav.

Andmete fail on Moodles. Salvestage see oma arvutisse. Fail on kodeeringus UTF-8.

Palun esitada viimane töötav versioon! Palun esitada kahel viisil:

1. Moodle'sse (Kontrolltöö nr 1) ja
2. e-postiga praktikumijuhendajale, Subject: OOP KT1