**Ülesanne Bussisõit (maksimaalselt 10 punkti)**

* **(3 punkti) Buss**iga sõidavad **Reisija**d, igat reisijat kirjeldab **nimi**, **vanus** ning **elukoht** (nt. Tallinn või Rapla). Klassis **Reisija** koostada meetod **toString()**, mis tagastab konkreetse reisija andmed stringina.

Koostada programm, millega saab modelleerida busse ja reisijaid. Luua klassis **Buss** meetod **reisijaSiseneb(Reisija\*)**, millega reisijad bussi sisenevad. Buss peab talletama infot kõikide seal viibivate reisijate kohta. Vaikimisi on bussis 10 istekohta, programm ei tohi lubada, et buss läheks reisijatest "lõhki". Samuti tuleks garanteerida, et iga reisija saab bussis viibida vaid ühes „eksemplaris“. Selle kontrollimine muutub lihtsamaks, kui loote klassis **Reisija** meetodi **bool equals(Reisija\*)**;

Klassis **Buss** koostada meetod **toString()**, mis käib läbi kõik bussis olevad reisijad ja tagastab nende andmed stringina.

* **(3 punkti)** Reisijal võib olla **Sõidukaart**, aga võib ka mitte olla. Osal reisijatest on sõidukaart isikustatud, osadel mitte. Osa reisijaid valideerib bussi sisenedes oma sõidukaardi, osa mitte.

Märkus. Sõidukaardi võib defineerida ka struct-ina.

Sõiduõigust omavad:

* kõik alla 6 aasta vanused isikud;
* kõik vähemalt 65 aasta vanused isikud;
* kõik ülejäänud Tallinna elanikud, kellel on isikustatud sõidukaart ja see on bussi sisenemisel valideeritud;
* kõigil ülejäänud juhtudel – need isikud, kellel on sõidukaardile laetud piisavalt raha ja see on bussi sisenemisel valideeritud;

Klassi Buss lisada meetod **string tuliMUPO()**, mis karistab kõiki sõiduõiguseta reisijaid bussist välja viskamisega. Pärast funktsiooni väljakutset tohivad bussis viibida ainult kehtiva sõiduõigusega reisijad. Funktsioon tagastab sõiduõiguse kontrolli tulemused tekstina.

Realiseerida kirjeldusele vastav programm, luua vajalikud klassid ja meetodid. Programmi põhifunktsioonis **main()** luua mingi arv busse ja reisijaid ning testida programmi kõikide võimalike sõiduõiguse kombinatsioonide korral. Testimisel kasutatavate reisijate andmed peaksid asuma mingis tekstifailis.

* **(4 punkti)** Lisada klassi **Buss** andmeväli **liiniNumber** ja meetod **annaLiiniNumber().** Lisada klassi **Reisija** andmeväli **vajalikuLiiniNumber.** [vastav get-meetod pole hädavajalik]

Luua uus klass **Peatus**, milles on **vector**-tüüpi kollektsioon **ootajad** ja meetod **saabusReisija(Reisija \*keegi)**, mis lisab peatusesse saabunud reisija ootajate kollektsiooni.

Luua klassis **Peatus** meetod **saabusBuss(Buss \*mingi)**, milles kõik peatuses olevad reisijad, kelle vajaliku liini number ühtib saabunud bussi liini numbriga, peavad saama võimaluse siseneda saabunud bussi. Ootavatele reisijatele tuleb teatada bussi saabumisest meetodiga **minePeale(Buss \*mingi).** Need reisijad, kellel õnnestus bussi siseneda, tuleb ootajate kollektsioonist kustutada.

Testprogrammis **main()** luua vähemalt üks peatus ja panna sinna vajalik hulk reisijaid ootama saabuvaid busse. Busse peaks olema vähemalt kaks (erinevate liininumbritega). Printida konsoolile ammendav informatsioon peatuses toimuvast. Et see info printimine oleks mugavam, võiks kõigis klassides olla meetod **toString();**

Ülesande lahenduse (2 esimest alamülesannet) prototüüp asub projektis Bussisõit. Ülesande lahendamisel võiks eeskujuks võtta harjutustundide projektid Riigikogu ja Erase.

**Üldised nõuded kodutöödele**.

* Programm peab töötama!
* Programm teeb täpselt seda ja ainult seda, mis on nõutud ülesandes.
* Programmi ülesehitus peab baseeruma OOP 3 vaalal: [kapseldamine (ka realisatsiooni varjamine!)](http://en.wikipedia.org/wiki/Encapsulation_%28object-oriented_programming%29), tuletamine, polümorfism.
* Programmi lähtetekst **peab** olema „[trepitud](http://foorum.hinnavaatlus.ee/viewtopic.php?t=294507)“ («[лесенкой](http://www.amse.ru/courses/algorithms6/03/Silina/03.php)») – treppimise *stiili* võite valida ise. Treppimata lähtekoodiga programm ei kuulu hindamisele! Ka klasside, nende väljade ja meetodite **nimede** juures tuleks järgida mingit ühte [stiili](https://en.wikibooks.org/wiki/C%2B%2B_Programming/Weblinks#C.2B.2B_Coding_Conventions).
* Kogu programm (eriti aga kõigi klasside kõik väljad ja meetodid) peab olema varustatud asjakohaste kommentaaridega.
* [Copy/Paste stiilis programmeerimine](https://en.wikipedia.org/wiki/Copy_and_paste_programming) on karistatav trahvipunktidega! Alternatiiviks on korduvalt kasutatava koodi paigutamine alamprogrammi või meetodisse.
* Kui töö kaitsmisel tudeng ei oska seletada oma programmi ülesehitust ja selle toimimise loogikat, siis ka funktsionaalselt õige lahenduse korral saab tudeng hindeks mitte rohkem kui 1/3 ülesande maksimumpunktidest.

TÄHTAEG: **3. november**