Kodutöö esitamise tähtaeg: 15. oktoober, 23:59

Magasin ja järjekord

Selles praktikumis vaatleme kaht abstraktset andmetüüpi – magasini ja järjekorda. Abstraktne andmetüüp on oma ideelt väga sarnane liidesega – selles kirjeldatakse, mida on võimalik teha, kuid ei puudutata seda, kuidas midagi teha.

Magasin ja järjekord on mõlemad andmetüübid, mis võimaldavad elemente lisada ja eemaldada. Erinevus seisneb selles, et magasinist saab eemaldada ainult sinna kõige viimasena lisatud magasinis veel allesoleva elemendi (*Last in, first out* – LIFO), järjekorrast aga ainult elemendi, mis lisati varem kui kõik teised järjekorras olevad elemendid (*First in, first out* – FIFO).

Ülesanne 1

1. Implementeerida magasin vastavalt liidesele *Stack*.
2. Implementeerida järjekord vastavalt liidesel *Queue*.

Eelmises praktikumis tegime tutvust kahendpuudega. See praktikum vaatleme kaht peamist viisi, kuidas kahendpuid (ja ka puid üldisemalt) **läbitakse** (*traverse*). Puu läbimise all mõeldakse puu kõikide tippude mingil viisil töötlemist.

Esimene viis puu läbimiseks on teha seda **sügavuti** (*depth-first*). Sellisel juhul töödeldakse iga tipu korral kõigepealt selle tipu kõik järglased (ehk tipu lapsed või nende järglased) ning alles pärast seda teised tipud.

Teine viis puu läbimiseks on **laiuti**. Sellisel juhul töödeldakse iga tipu korral kõigepealt tema **õed-vennad** (*siblings*) ehk tipud, millega on tipul sama vanem ning seejärel ülejäänud tipud.

Ülesanne 2
Kasutada eelnevas ülesandes loodud magasini ja järjekorra implementatsioone.

1. Kirjutada meetod, mis läbib kahendpuu süvitsi. Implementeerida liides *DFS*.
2. Kirjutada meetod, mis läbib kahendpuu laiuti. Implementeerida liides *BFS*.

Ülesanne 3
Luua oma versioon brauseri aadressiribast, mille edasi ja tagasi liikumise funktsionaalsus oleks samasugune nagu enimlevinud brauseritel (nt. *Google Chrome*). Implementeerida liides *AddressBar* ning katsetada seda klasside *Browser* ja *BrowserImpl* abil.

`https://github.com/ut-aa/aa2016-lab3`