**Ülesanne 3 lisa**

Osaliselt sorditud loendi sortimine läheb insertion sordiga kiiremini kui täielikult sortimata loendi sortimine, kuna mõned võrdlused ning ümbertõstmised jäävad ära.

**Ülesanne 4**

**a) Millistel eelmainitud sortimisalgoritmidel on halvimal juhul ajakompleksus O(n^2)?**

Kõigil eelmainitud sortimisalgoritmidel on halvimal juhul ajakompleksus O(n^2).

**b) Milline sortimisalgoritm oleks kõige sobivam sortimaks loendit täisarvudega, mis jäävad vahemikku 1 kuni 100 ja miks?**

Kõige sobivam võiks olla insertion sort. Kui on vähe arve ning need arvud on väiksed, siis on insertion sort väga kiire. Ka radix sort on küll väga kiire, kuid see on hulga keerulisem, kui insertion sort.

**Ülesanne 5**

**a) Defineeri, mida tähendab, et sortimisalgoritm on "stabiilne." Anna näide stabiilsest sortimisalgoritmist antud loendist.**

Stabiilne sortimisalgoritm tähendab seda, et samad väärtused (näiteks kahel inimesel täpselt sama nimi) jäävad samale kohale ka peale sortimist.

Praeguses kodutöös kasutasin kolme sortimismeetodit: Insertion sort, bubble sort ja selection sort. Insertion sort ja bubble sort on stabiilsed. Selection sort on ebastabiilne.

Näide stabiilsest sortimisalgoritmist

10 20 20 30 10 - sortimata loend

10 10 20 20 30 - sorditud loend

Kõik eelnevad liikmed jäid omale kohale (punane 20 enne rohelist 20 ning sinine 10 jäi endiselt esimeseks).

Kui kasutada ebastabiilset sortimist, näiteks selection sorti, võib tulemus olla hoopis selline:

10 10 20 20 30 - sorditud loend samast sortimata loendist.

Tulemus ei pruugi ilmtingimata olla selline, kuna neid kombinatsioone on erinevaid.

**Boonus (3p): Kuidas näeks loend [8, 3, 5, 4, 7, 6, 2] välja pärast esimest läbimist antud algoritmide puhul?**

**a) Bubble Sort:** Võrdleb üksteise kõrval olevaid elemente i1 <> i2 ning kui i2 on suurem kui 1i, vahetatakse nende kohad ära.

**[3, 8, 5, 4, 7, 6, 2]**

**b) Selection Sort:** Otsib massiivist kõige väiksema elemendi ning tõstab selle kõige ette. Antud massiivis on kõige väiksem element 2

**[2, 8, 3, 5, 4, 7, 6]**

**c) Insertion Sort:** Võtab järjest rohkem elemente juurde ning võrdleb iga järgmist elementi kõigi eelnevate elementidega. Kuna läbitakse ainult üks samm, siis erinevus Bubble sortiga välja ei tule. Selles massiivis tuleks bubble sorti ja insertion sorti erinevus välja alles peale kolmandat läbimist.

**[3, 8, 5, 4, 7, 6, 2]**