Kodutöö 2 küsimused

Ülesanne 1 Bubble sort

Lisan kirjutatud programmi väljundi.

Algne sisend:

[64, 34, 25, 12, 22, 11, 90]

Vahetus 1

Vahetati element 64 ja element 34

[34, 64, 25, 12, 22, 11, 90]

Vahetus 2

Vahetati element 64 ja element 25

[34, 25, 64, 12, 22, 11, 90]

Vahetus 3

Vahetati element 64 ja element 12

[34, 25, 12, 64, 22, 11, 90]

Vahetus 4

Vahetati element 64 ja element 22

[34, 25, 12, 22, 64, 11, 90]

Vahetus 5

Vahetati element 64 ja element 11

[34, 25, 12, 22, 11, 64, 90]

Vahetus 6

Vahetati element 34 ja element 25

[25, 34, 12, 22, 11, 64, 90]

Vahetus 7

Vahetati element 34 ja element 12

[25, 12, 34, 22, 11, 64, 90]

Vahetus 8

Vahetati element 34 ja element 22

[25, 12, 22, 34, 11, 64, 90]

Vahetus 9

Vahetati element 34 ja element 11

[25, 12, 22, 11, 34, 64, 90]

Vahetus 10

Vahetati element 25 ja element 12

[12, 25, 22, 11, 34, 64, 90]

Vahetus 11

Vahetati element 25 ja element 22

[12, 22, 25, 11, 34, 64, 90]

Vahetus 12

Vahetati element 25 ja element 11

[12, 22, 11, 25, 34, 64, 90]

Vahetus 13

Vahetati element 22 ja element 11

[12, 11, 22, 25, 34, 64, 90]

Vahetus 14

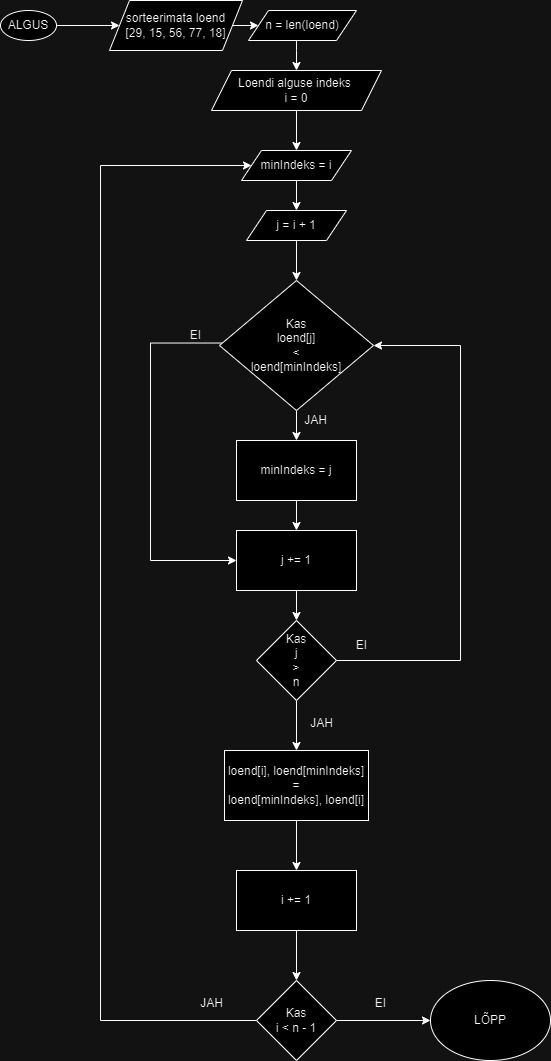
Vahetati element 12 ja element 11

[11, 12, 22, 25, 34, 64, 90]

Sorteeritud loend on:

11 12 22 25 34 64 90

Ülesanne 2 Selection sort



Ülesande 2 programmi väljund. Esimene vahetatav element on iteratsiooni väikseim element.

Iteratsioon 1

Vahetati elemendid 15 ja 29

[15, 29, 56, 77, 18]

Iteratsioon 2

Vahetati elemendid 18 ja 29

[15, 18, 56, 77, 29]

Iteratsioon 3

Vahetati elemendid 29 ja 56

[15, 18, 29, 77, 56]

Iteratsioon 4

Vahetati elemendid 56 ja 77

[15, 18, 29, 56, 77]

Sorteeritud loend

[15, 18, 29, 56, 77]

Ülesanne 3 Insertion sort

Antud on osaliselt sorteeritud loend: [12, 11, 13, 5, 6, 7].

Miks Insertion Sort võib olla tõhusam sellele loendile võrreldes täielikult sortimata loendiga?

Halvimal juhul on insertion sorti ajakomplekslus O(n2), näiteks kui loend on sorteeritud tagurpidi.

Ajakompleksus läheb paremaks insertion sortil osaliselt sorteeritud loendil, kuna igal uuel iteratsioonil ta eirab sorteeritud tabeli osa ja tegeleb ainult sorteerimist vajava osaga.

Pilt, millel on kujutatud tekst, kuvatõmmis, Font, number

Kirjeldus on genereeritud automaatselt

Programm läheb while loopi kui j on suurem kui „0“ **JA** sorteeritav muutuja key on väiksem kui sorteeritud osa viimane element j. Kui muutujat key pole vaja sorteeritud loendisse panna läheb see kohale j + 1 ja while loopi sisse ei minda, mis tõttu ka ajakompleksus väheneb.

Ülesanne 4 Ajakompleksuse analüüs

1. Kõikide kolme sortimisalgoritmi halvim ajakompleksus on O(n2). Üks näide on, et loend on kahanevad järjekorras kuid soovime kasvavat järjekorda. Igal iteratsioonil tuleb teha N(N-1)2 võrdlust ja N(N-1)/2 vahetust.
2. Arvan, et kõige sobivam sortimisalgoritm oleks insertion sort, kuna sortimise käigus algoritm hakkab ka kiiremini töötama võrreldes selection ja bubble sortiga.

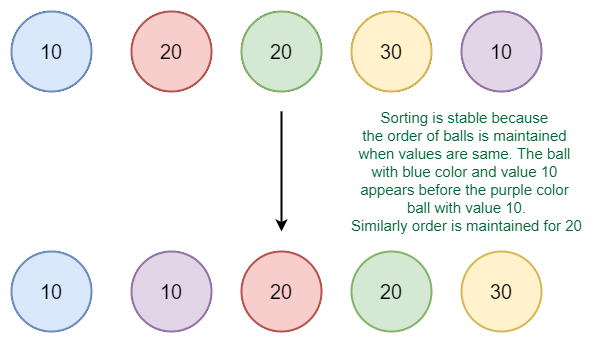
Leidsin ka lehekülje kus saab näha erinevate algoritmide tööd ning erinevate sisenditega mängides lõpetas insertion sort alati enne selection ja bubble sorti.

<https://www.toptal.com/developers/sorting-algorithms>

Ülesanne 5 Stabiilsus ja adaptiivsus sortimisel

Sortimisalgortitm on stabiilne kui algoritm jätab kaks sama võtmeväärtusega muutujat samasse järjekorda ka peale sorteerimist.

Vaikimisi on bubble sort ja insertion sort stabiilsed. Selection sort ei ole stabiilne kuna algoritm ei muuda kõrvuti asetsevaid elemente.



<https://www.geeksforgeeks.org/stable-and-unstable-sorting-algorithms/>

Sortimisalgoritm on adaptiivne kui algoritmi ajakompleksus muutub vastavalt sellele kuidas loendi elemendid järjestatud on.

Insertion sorti ja bubble sorti loetakse adaptiivseteks algoritmidest kuna algoritmid ei muuda sorteeritud elemente, mis tõttu väheneb ka iteratsioonide arv.

Selection sort ei ole adaptiivne kuna algne loendi järjekord ei mõjuta võrdluste arvu.

Boonus

Sisend loend: [8, 3, 5, 4, 7, 6, 2]

Bubble sorti esimene iteratsioon:

Vahetus 1

Vahetati element 8 ja element 3

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

Selection sorti esimene iteratsioon:

Iteratsioon 1

Vahetati elemendid 2 ja 8

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

Insertion sorti esimene iteratsioon:

Iteratsioon 1

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