1. Ülesandeid

I Ülesanded kogust [2], ptk 13 ja 14

13.3, 13.4, 13.8, 13.9, 13.14-26, 13.33, 14.17-21.

II Lisaks

1.1. *(s)* Koostada meetod kõikide teede leidmiseks antud graafi kahe tipu vahel.

1.2. *(s)* Realiseerida Dijkstra algoritm ([1], lk 100) graafis lühimate teede otsimiseks. Eelistusjärjekord (front) Q esitada tippude kahendkuhjana, milles kirje võtmeks tipu väli "d".

1.3. *(s)* Eesti asulate-teede plaani kujutab graaf failis *g_linnad_teed.txt* (kärbitud plaan failina *g_linnad_teed_mini.txt*), milles asulat esindab graafi tipp ning asulate $t1$ ja $t2$ vahelist (maan)teed kaar; kaare märgendiks on tee pikkus km.

Olgu antud asulate-teede plaan ja lõigulimiit x. Koostada programm, mis loob Floyd-Warshalli meetodil asulate kaugusmaatriksi a; maatriksi elemendiks a_{ij} olgu sellise lühima marsruudi kogupikkus i-ndast asulast j-ndasse asulasse (või $+\infty$, kui marsruuti ei leidu), milles ükski lõik asulast asulasse ei ületa lõigulimiiti x.

Testida programmi juhtudel

1) *g_linnad_teed_mini.txt* ja $x = 70$;
2) *g_linnad_teed.txt* ja $x = 55$.

Tulemuste (osaliseks) kontrollimiseks rakendada graafis lühimate teede leidmise Bellman-Fordi meetodit.

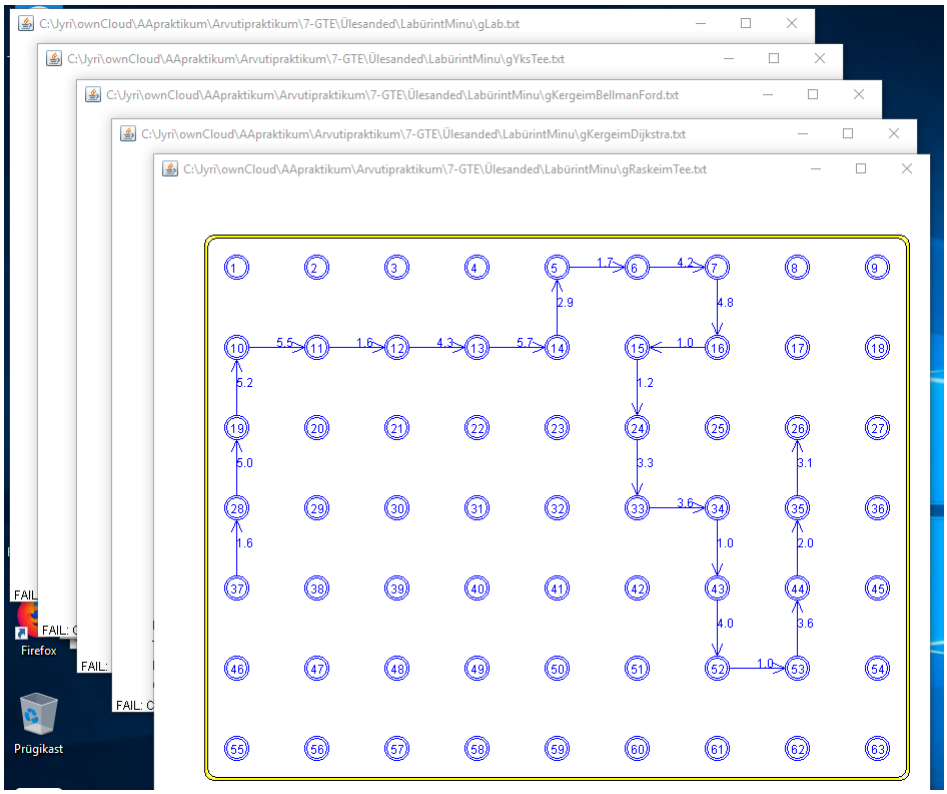
1.4. *(s)* Koostada labürindi uurimise programm.

1) Luua ja lisada uus omatehtud näitelabürint klassi *Labürindid.java*;
2) Kirjutada ja testida selle labürindi uurimise programm, milles
   - esitatakse see labürint graafina g
   - leitakse üks (suva)tee graafis g sisenemiskohast väljumiskohta;
   - Bellman-Fordi meetodil leitakse kergeim (lühim) tee graafis g sisenemiskohast väljumiskohta;
   - Dijkstra meetodil leitakse kergeim (lühim) tee graafis g sisenemiskohast väljumiskohta;
   - leitakse kõik teed graafis g sisenemiskohast väljumiskohta ja nende seast raskeima koguraskus (teepikkus).

Ekraanile kuvatakse viis graafi: graaf g, graaf ühe suvateega, graaf Bellman-Fordi leitud kergeima teega, graaf Dijkstra leitud kergeima teega ja graaf raskeima teega (vt joonis 1). Võetakse ka tee(de) leimiseks kulunud aeg igal konkreetsel juhul.

Soovitatav väljundvorm konsoolil (nädisena):

```
================ NÄITELABÜRINT nr 2 ================

Labürint (7 x 9 maatriks):
     0: 1: 2: 3: 4: 5: 6: 7: 8:
 0:  1  0  1  1  0  0  0  0  1
 1:  0  0  0  0  0  0  0  1  0
 2:  0  1  0  1  0  0  1  0  0
 3:  0  0  0  0  1  0  0  0  1
 4:  0  1  0  0  1  0  0  0  1
 5:  1  0  1  0  1  1  0  0  0
 6:  0  0  0  0  0  0  1  0  0
sisenemiskoha indeksid: 4 0
väljumiskoha  indeksid: 2 7

Labürint graafina failis: gLab.txt
    lähtetipu number: 37
    sihttipu number:  26
Servadel juhuraskused

------------------------- ÜKS TEE
Leitud: [37, 28, 19, 10, 11, 12, 13, 14, 5, 6, 7, 16, 15, ...]
Aeg (millisek): 3.643485
Kuvatud: gYksTee.txt
Tee raskus: 58

------------------------- KERGEIM TEE, Bellman-Ford
Leitud: [37, 28, 29, 30, 21, 12, 13, 14, 15, 24, 33, 34, ...]
Aeg (millisek): 52.005891
Kuvatud: gKergeimBellmanFord.txt
Tee raskus: 44

------------------------- KERGEIM TEE, Dijkstra
Leitud: [37, 28, 29, 30, 21, 12, 13, 14, 15, 24, 33, 34, ...]
Aeg (millisek): 12.411617
Kuvatud: gKergeimDijkstra.txt
Tee raskus: 44

------------------------- KÕIK TEED
Kõikvõimalike teede arv: 48
Aeg (millisek): 16.063072
Kuvatud: gRaskeimTee.txt
Tee raskus: 66
=====================================================
```



Joonis 1: Graafide ekraanikuvad ülesandest 1.4. Esiplaanil raskeim tee.

Suunised

1. Ülesandeid

1.1. Hõlpsasti saadav klassis *TeedGraafis.java* kirjeldatud ühe tee leidmise meetodist *otsida_suvatee*. Viimase lõppu lisada tipu *t* märgise "kinni" eemaldamine. **1.2.** Vt *DijkstraMall.java* ja *KahendkuhiTMall.java*. **1.3.** Vt *FloydWarshallMall.java*. **1.4.** Klassid *Labürint.java*, *Labürindid.java*, *TeeInfo.java*.

Viited

[1] J. Kiho. *Algoritmid ja andmestruktuurid.* Kolmas, parandatud ja täiendatud trükk. TÜ, 2003, 147 lk. http://dspace.ut.ee/bitstream/handle/10062/16872/9985567676.pdf?sequence=1 (16.04.2017)

[2] A. Peder, J. Kiho, H. Nestra. *Algoritmid ja andmestruktuurid. Ülesannete kogu.* TÜ, 2017, 131 lk.