

Praktikum 7

Kahendpuud

- **Abivahendid kahendpuu loomiseks, kuvamiseks ja töötlemiseks**
- **Harjutusülesanded**
 - KP_1. Täieliku kahendpuu valmistamine
 - KP_2. Kahendpuu kõrgus
 - KP_3. Tasakaalu kontrollimine
 - KP_4. Kahendpuu ja puu läbimine
 - KP_5. Aritmeetilise avaldise puu
 - KP_6. Tipu järglaste koguarv
 - KP_7. Tippudele kõrguse- ja tasemeväljad
 - KP_8. Tasemete generaator
 - KP_9. Kompaktse kahendpuu valmistamine
 - KP_10. Kahendpuu peegeldus

Abivahendid kahendpuu loomiseks, kuvamiseks ja töötlemiseks

Pakett *ee.ut.kiho.aa.graaf* (`import ee.ut.kiho.aa.graaf.*;`)

sisaldab klasside

Graaf

GraafiJoonistaja

Kaar

Kahendpuu

Puu

Tipp

kirjeldusi.

Kahendpuud käsitletakse kui graafi/puu erijuhtu. Seetõttu on rakendatavad graafi- ja puutöötlemise meetodid klassidest *Graaf/Puu*. Kahendpuu/puu juureks on graafi esimene tipp (tipp indeksiga 0).

Vasak-parem alluvussuhe on määratud tippude x-koordinaatidega:

kui tipp t on tipu u alluv (st on olemas kaar $u \rightarrow t$), siis

t on tipu u vasak alluv, kui $t.x < u.x$,

t on tipu u parem alluv, kui $t.x > u.x$.

n-tipulise juhu-kahendpuu *kp* valmistamine:

```
Kahendpuu kp = new Kahendpuu(Puu.juhupuu(n, 2));
```

Harjutusülesanded

KP_1. Täieliku kahendpuu valmistamine [Kiho 2003, lk 28]

Kirjutada ja testida rekursiivne meetod järgmise ülesande lahendamiseks.

Antud: arv m – loodava kahendpuu tasemete arv.

Tulemus: täielik m -tasemeline kahendpuu.

Java

Ühetipulise (ainult juur) kahendpuu kp tegemine:

```
Graaf g = Graaf();           -- tühi graaf
g.lisada(new Tipp());        -- lisada üks tühi tipp
Kahendpuu kp = new Kahendpuu(g);
```

Kahendpuus kp alluvateta tipule t mõlema alluva lisamine:

```
Tipp uus = new Tipp();       -- tühi tipp
kp.lisadaAlluv(t, uus, true); -- true: lisamine vasakuks alluvaks
uus = new Tipp();
kp.lisadaAlluv(t, uus, false) -- false: lisamine paremaks alluvaks
```

Ehk lühemalt:

```
kp.lisadaAlluv(t, new Tipp(), true);
kp.lisadaAlluv(t, new Tipp(), false);
```

Kahendpuu (ka graafi ja puu) kuvamine:

```
kp.väljastadaTabelina("fail.txt"); -- kahendpuu faili (tabelkujul)
new GraafiJoonistaja("fail.txt", true); -- tippudes märgendid
new GraafiJoonistaja("fail.txt"); -- tippudes numbrid
```

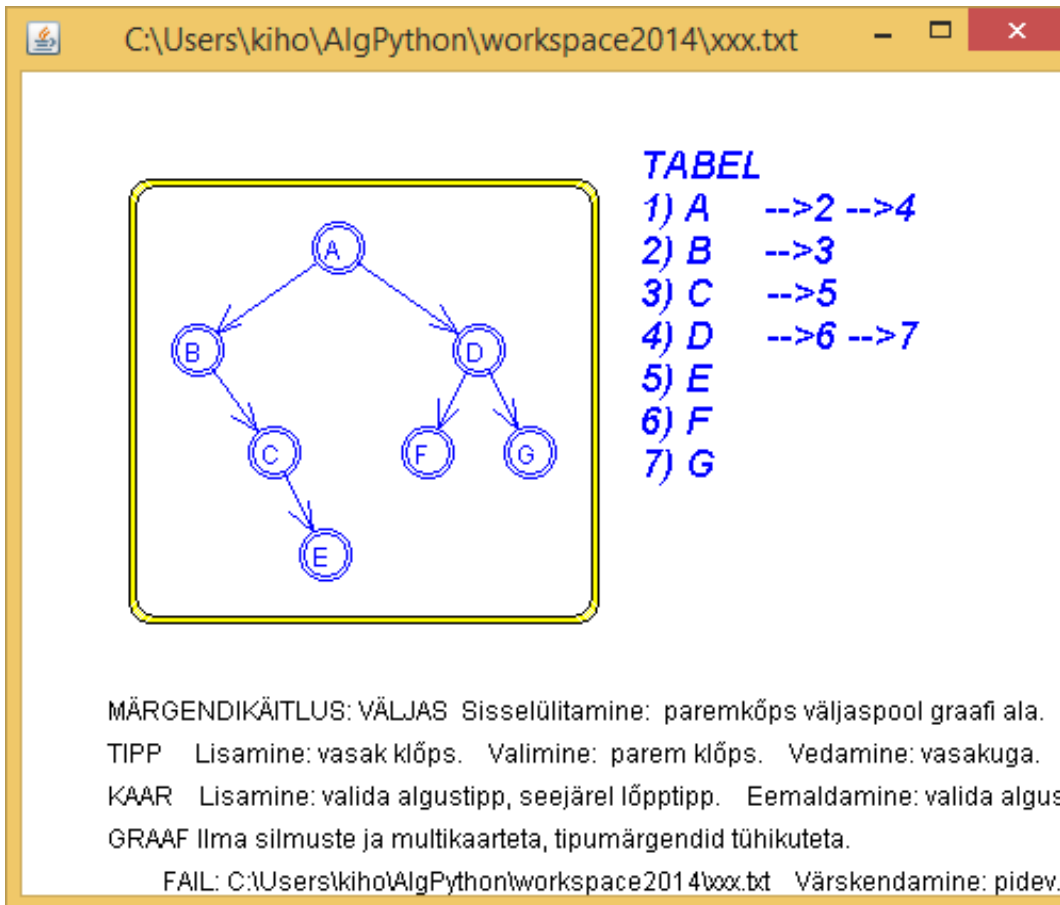
KP_2. Kahendpuu kõrgus

Kirjutada ja testida rekursiivne meetod järgmise ülesande lahendamiseks.

Antud: kahendpuu ja selle tipp.

Tulemus: antud tipust algava (alam)kahendpuu kõrgus.

Testi tulemuse näide



The screenshot shows a text editor window with the following content:

TABEL

1) A	-->2 -->4
2) B	-->3
3) C	-->5
4) D	-->6 -->7
5) E	
6) F	
7) G	

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi ala.
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasakuga.
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida algus
GRAAF ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\kiho\AlgPython\workspace2014\xxx.txt Värskendamine: pidev.

Alampuu
juur kõrgus

A	4
B	3
C	2
D	2
E	1
F	1
G	1

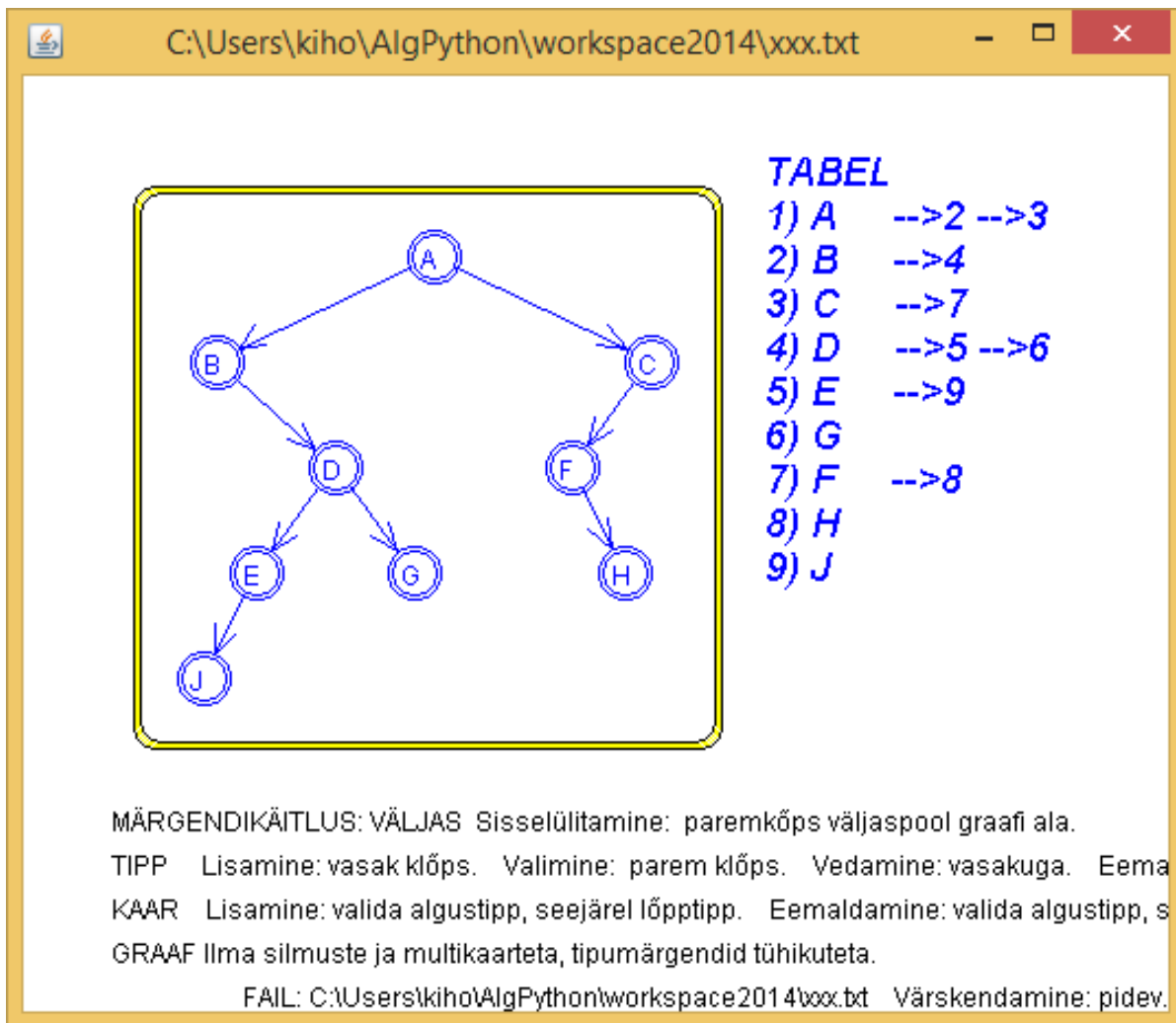
KP_3. Tasakaalu kontrollimine

Kirjutada ja testida meetod järgmise ülesande lahendamiseks.

Antud: kahendpuu.

Tulemus: antud kahendpuu nende tippude järjend (listina), kus tasakaal on rikutud, st tipu alluvate kõrgus erineb rohkem kui 1 võrra.

Testi tulemuse näide



The screenshot shows a window titled "C:\Users\kiho\AlgPython\workspace2014\xxx.txt". Inside, there is a binary tree diagram with nodes A through J. Node A is the root. Node B is the left child of A, and node C is the right child. Node D is the left child of B, and node F is the left child of C. Node E is the left child of D, and node G is the right child of D. Node J is the left child of E, and node H is the right child of F. To the right of the tree is a table:

TABEL		
1) A	-->2	-->3
2) B	-->4	
3) C	-->7	
4) D	-->5	-->6
5) E	-->9	
6) G		
7) F	-->8	
8) H		
9) J		

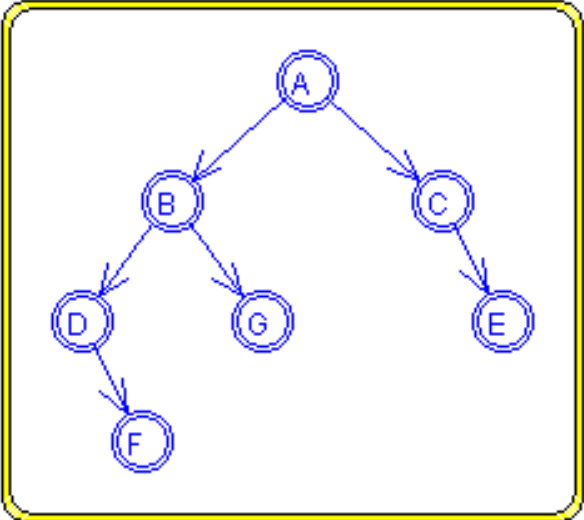
Below the table, there is a list of instructions in Estonian:

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi ala.
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasakuga. Eema
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida algustipp, s
GRAAF ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\kiho\AlgPython\workspace2014\xxx.txt Värskendamine: pidev.

Tasakaal rikutud
tipus B
tipus C

Testi tulemuse näide

C:\Users\kiho\AlgPython\workspace2014\xxx.txt



```
graph TD; A((A)) --> B((B)); A --> C((C)); B --> D((D)); B --> G((G)); C --> E((E)); D --> F((F));
```

TABEL

- 1) A -->2 -->3
- 2) B -->4 -->7
- 3) C -->5
- 4) D -->6
- 5) E
- 6) F
- 7) G

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi a
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasaku
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida a
GRAAF ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\kiho\AlgPython\workspace2014\xxx.txt Värskendamine: pidev.

On tasakaalus.

KP_4. Kahendpuu ja puu läbimine

KP_4-1. Kahendpuu läbimine [Kiho 2003, lk 29]

Kirjutada ja testida rekursiivsed meetodid järgmiste ülesannete lahendamiseks

Antud: kahendpuu ja selle tipp t .

Tulemus: tipust t algava kahendpuu tipud eesjärjestuses (listina).

Antud: kahendpuu ja selle tipp t .

Tulemus: tipust t algava kahendpuu tipud keskjärjestuses (listina).

Antud: kahendpuu ja selle tipp t .

Tulemus: tipust t algava kahendpuu tipud lõppjärjestuses (listina).

Java

Tagastada (listina) tipust t algava kp alamkahendpuu tipud eesjärjestuses:

```
ArrayList<Tipp> eesjrst(Kahendpuu kp, Tipp t){
    ArrayList<Tipp> tulem = new ArrayList<Tipp>();
    if (t == null) return tulem;
    tulem.add(t);
    tulem.addAll(eesjrst(kp, kp.vasakAlluv(t)));
    tulem.addAll(eesjrst(kp, kp.paremAlluv(t)));
    return tulem;
}
```


KP_4-2. Puu läbimine

Kirjutada ja testida rekursiivsed funktsioonid järgmiste ülesannete lahendamiseks.

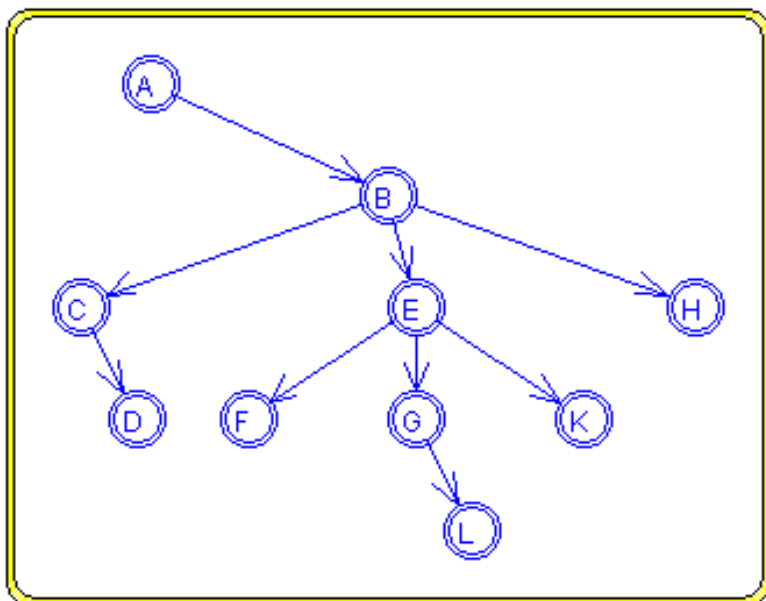
Antud: puu ja selle tipp t .

Tulemus: tipust t algava (alam)puu tipud eesjärjestuses (listina).

Antud: puu ja selle tipp t .

Tulemus: tipust t algava (alam)puu tipud lõppjärjestuses (listina).

C:\Users\kiho\AlgPython\workspace2014\xxx.txt



TABEL

- 1) A -->2
- 2) B -->3 -->5 -->8
- 3) C -->4
- 4) D
- 5) E -->6 -->7 -->9
- 6) F
- 7) G -->10
- 8) H
- 9) K
- 10) L

Testi tulemuse näide

Puu eesjrst:
ABCDEFGHIGLKH

Puu lõppjrst:
DCFLGKEHBA

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi ala.

TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasakuga. Eemaldamine: pari

KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida algustipp, seejärel lõpti

GRAAF ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.

FAIL: C:\Users\kiho\AlgPython\workspace2014\xxx.txt Värskendamine: pidev.

KP_5. Aritmeetilise avaldise puu

Koostada programm, mis valmistab (juhusliku) aritmeetilise avaldise kahendpuu ja arvutab vastava avaldise väärtuse. (Vt ka järgmine slaid.)

Eraldi meetoditena kirjeldada järgmised alamülesanded:

KP_5-1. Aritmeetilise avaldise puu valmistamine

Kirjutada ja testida funktsioon järgmise ülesande lahendamiseks.

Antud: kahendpuu.

Tulemus: antudkahendpuu muudetud 0-2 kahendpuuks (igale ühe alluvaga tipule lisatud teine, puudunud alluv), seejärel rippuvate e. lehttippude märgenditeks seatud juhuarvud, ülejäänutele – binaarsete tehete märgid, juhuvalik (+ - * /) seast.

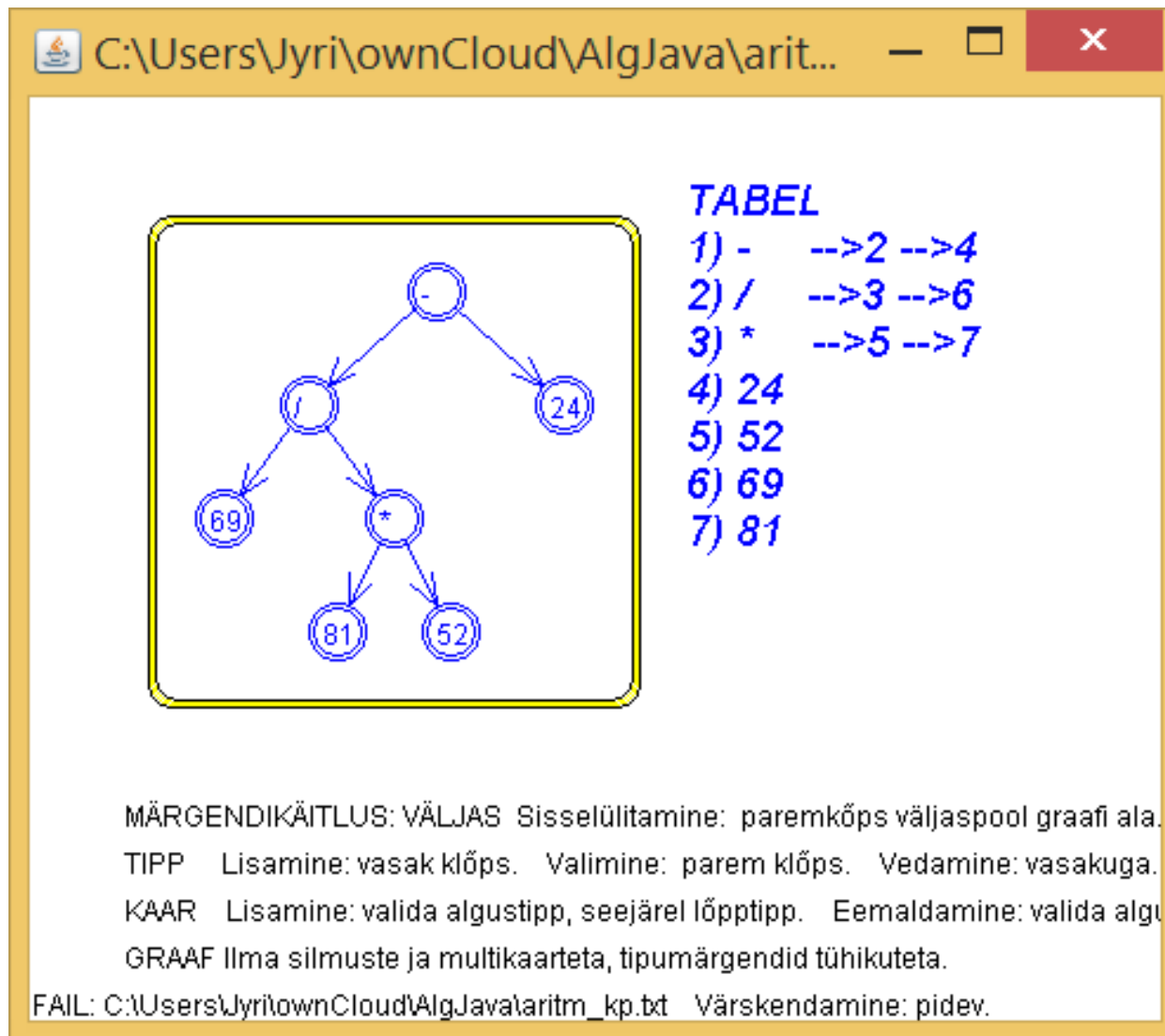
KP_5-2. Aritmeetilise avaldise puust väärtuse arvutamine

Kirjutada ja testida rekursiivne funktsioon järgmise ülesande lahendamiseks.

Antud: aritmeetiline kahendpuu ja selle tipp t .

Tulemus: tipust t algava aritmeetilisest (alam)puust arvutatud väärtus.

Aritmeetilise avaldise kahendpuu (KOP_5 testi tulemuse) näide.



The screenshot shows a Java application window titled "C:\Users\Jyri\ownCloud\AlgJava\arit...". Inside the window, there is a diagram of an arithmetic expression tree and a table of operations.

The tree structure is as follows:

- Root node: $-$
- Left child of root: $/$
- Right child of root: 24
- Left child of $/$: 69
- Right child of $/$: $*$
- Left child of $*$: 81
- Right child of $*$: 52

The table of operations is:

TABEL		
1) $-$	-->2	-->4
2) $/$	-->3	-->6
3) $*$	-->5	-->7
4)	24	
5)	52	
6)	69	
7)	81	

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi ala.
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasakuga.
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida algu
GRAAF ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\Jyri\ownCloud\AlgJava\aritm_kp.txt Värskendamine: pidev.

Avaldise väärtus: -23.98.

KP_6. Tipu järglaste koguarv

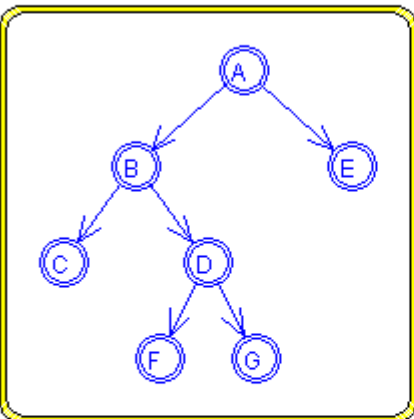
Kirjutada ja testida rekursiivne meetod järgmise ülesande lahendamiseks.

Antud: kahendpuu ja selle tipp.

Tulemus: antud tipu kõigi (vahetute ja kaugemate) järglaste koguarv antud kahendpuus.

Testi tulemuse näide

C:\AlgPyhton\workspace2014\box.txt



TABEL

- 1) A -->2 -->5
- 2) B -->3 -->4
- 3) C
- 4) D -->6 -->7
- 5) E
- 6) F
- 7) G

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi ala.
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasakuga. Eemaldamine
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida algustipp, seejärel l
GRAAF ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.

FAIL: C:\AlgPyhton\workspace2014\box.txt Värskendamine: pidev.

Tipp	Järglasi
A	6
B	4
C	0
D	2
E	0
F	0
G	0

KP_7. Tippudele kõrguse- ja tasemeväljad

KP_7-1. Kirjutada ja testida rekursiivne meetod järgmise ülesande lahendamiseks.

Antud: kahendpuu kp , selle tipp t .

Tulemus: tipust t algavas kp alampuus on iga tipu u väljale "h" omistatud tipust u algava alampuu kõrgus; tagastatakse tipust t algava alampuu kõrgus.

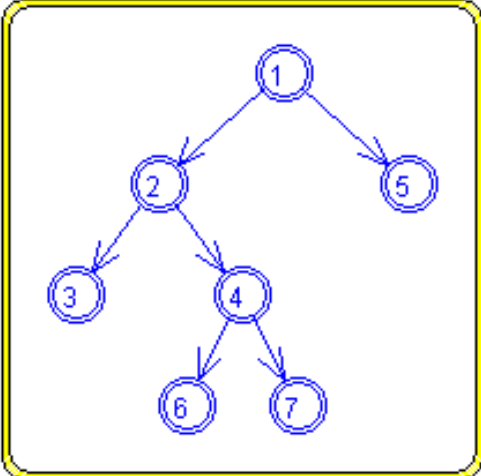
KP_7-2. Kirjutada ja testida rekursiivne meetod järgmise ülesande lahendamiseks.

Antud: kahendpuu kp , selle tipp t ja tipu t tasemenumber m .

Tulemus: tipust t algavas kp alampuus on iga tipu väljale "tase" omistatud tasemenumber; antud tipu tasemenumbriks on m , selle alluvatel $m+1$ jne.

KP_7 testi tulemuse näide

C:\Users\Jyri\ownCloud\AlgJava\kpxxx.t... — □ ×



TABEL

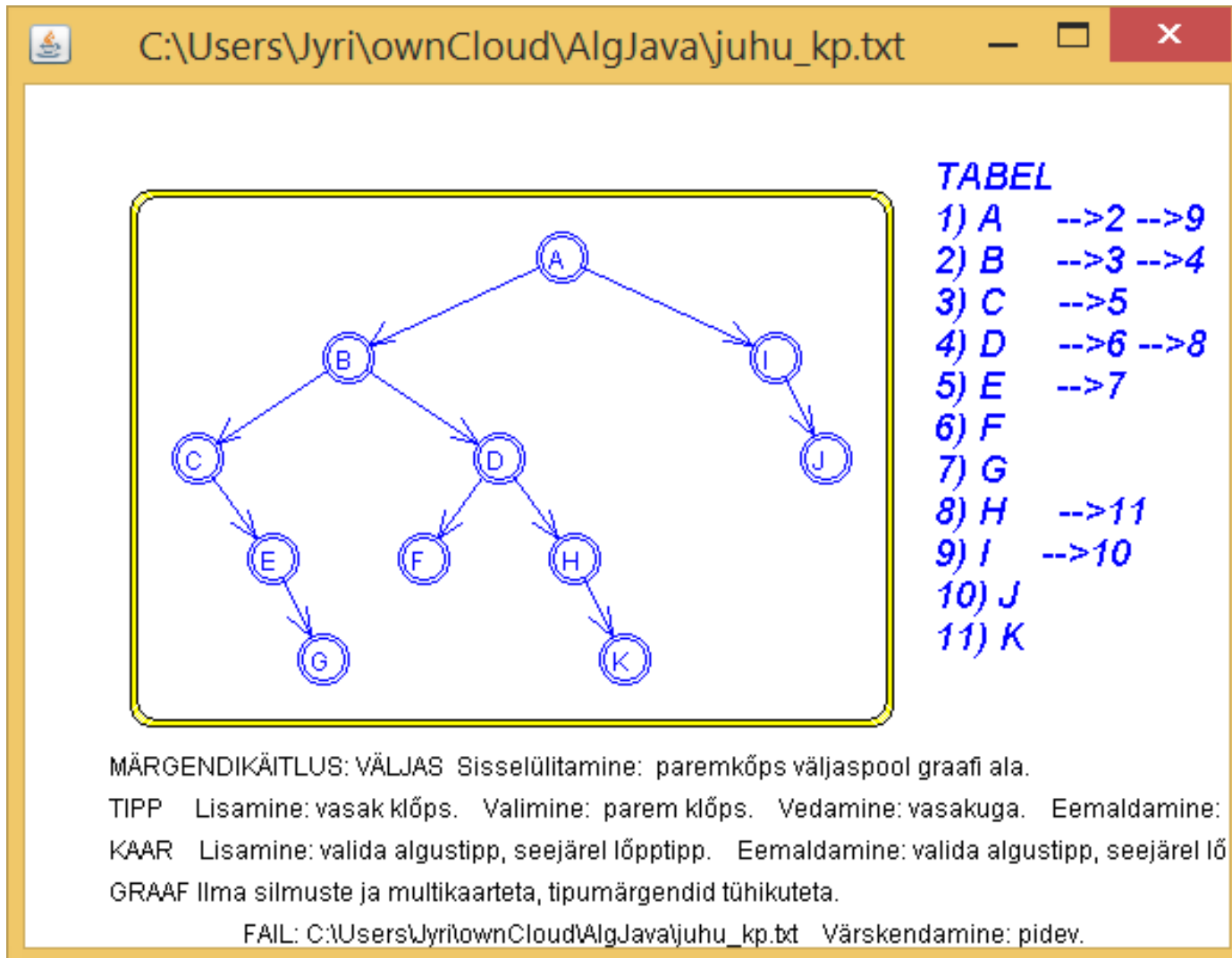
- 1) tase=1;h=4;A-->2 -->5
- 2) tase=2;h=3;B-->3 -->4
- 3) tase=3;h=1;C
- 4) tase=3;h=2;D-->6 -->7
- 5) tase=2;h=1;E
- 6) tase=4;h=1;F
- 7) tase=4;h=1;G

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremköps väljaspool graafi ala.
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasakuga. E
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida algustip
GRAAF Ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\Jyri\ownCloud\AlgJava\kpxxx.bt Värskendamine: pidev.

KP_8. Tasemete generaator

Kirjutada ja testida tasemete generaator genereerimaks järjest antud kahendpuu tasemeid (tippude loeteludena).

Testi tulemuse näide



The screenshot shows a Java application window titled "C:\Users\Jyri\ownCloud\AlgJava\juhu_kp.txt". The main content area displays a binary tree diagram with nodes labeled A through K. Node A is the root, with children B and I. B has children C and D. C has child E, which has child G. D has children F and H. H has child K. I has child J. The nodes are arranged in a level-by-level order from top to bottom.

TABEL

1) A	-->2	-->9
2) B	-->3	-->4
3) C	-->5	
4) D	-->6	-->8
5) E	-->7	
6) F		
7) G		
8) H	-->11	
9) I	-->10	
10) J		
11) K		

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi ala.
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasakuga. Eemaldamine:
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida algustipp, seejärel lõ
GRAAF ilma silmuse ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\Jyri\ownCloud\AlgJava\juhu_kp.txt Värskendamine: pidev.

Tipumärgendid
tasemeti:

A
B I
C D J
E F H
G K

KP_9. Kompaktse kahendpuu valmistamine [Kiho 2003, lk 28]

Kirjutada ja testida meetod järgmise ülesande lahendamiseks.

Antud: arv n – loodava kahendpuu tippude arv.

Tulemus: kompaktne n -tipuline kahendpuu.

Meetodi
üldskeem:

Valmistada (KP_1) täielik kahendpuu, milles on $\log_2(n + 1)$ taset.
Leida (KP_8) viimase taseme tipud ja lisada neile alluvateks
veel vajalik arv uusi tippe.

KP_10. Kahendpuu peegeldus

Kirjutada ja testida meetod järgmise ülesande lahendamiseks.

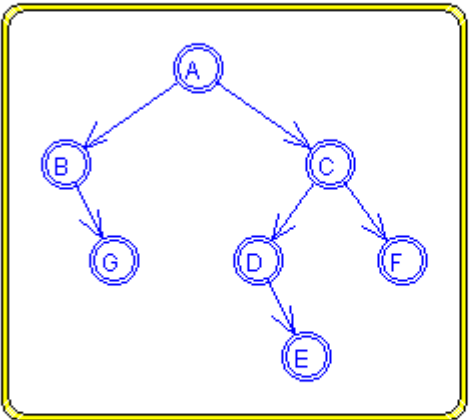
Antud: kahendpuu.

Tulemus: antud kahendpuu peegeldus üle vertikaaltelje.

Idee: tippude x-koordinaadid asendada vastandaruudega.

Testi tulemuse näide

C:\Users\kiho\AlgPython\workspace2014\xxx.txt



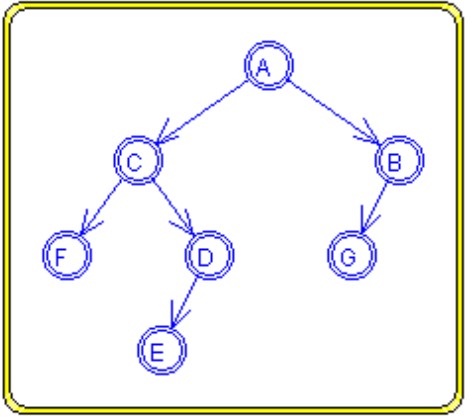
```
graph TD; A((A)) --> B((B)); A --> C((C)); B --> G((G)); C --> D((D)); C --> F((F)); D --> E((E));
```

TABEL

- 1) A -->2 -->3
- 2) B -->7
- 3) C -->4 -->6
- 4) D -->5
- 5) E
- 6) F
- 7) G

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasaku
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida
GRAAF Ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\kiho\AlgPython\workspace2014\xxx.txt Värskendamine: pidev.

C:\Users\kiho\AlgPython\workspace2014\yyy.txt



```
graph TD; A((A)) --> C((C)); A --> B((B)); C --> F((F)); C --> D((D)); D --> E((E)); B --> G((G));
```

TABEL

- 1) A -->2 -->3
- 2) B -->7
- 3) C -->4 -->6
- 4) D -->5
- 5) E
- 6) F
- 7) G

MÄRGENDIKÄITLUS: VÄLJAS Sisselülitamine: paremkõps väljaspool graafi
TIPP Lisamine: vasak klõps. Valimine: parem klõps. Vedamine: vasaku
KAAR Lisamine: valida algustipp, seejärel lõpptipp. Eemaldamine: valida
GRAAF Ilma silmuste ja multikaarteta, tipumärgendid tühikuteta.
FAIL: C:\Users\kiho\AlgPython\workspace2014\yyy.txt Värskendamine: pidev.