Datenstrukturen und Algorithmen: Übungsblatt #10

Abgabe am 5. Juli 2018

Niklas Hempel (349492), Jan Knichel (377779), Paul Orschau (381085)

29.06.2018

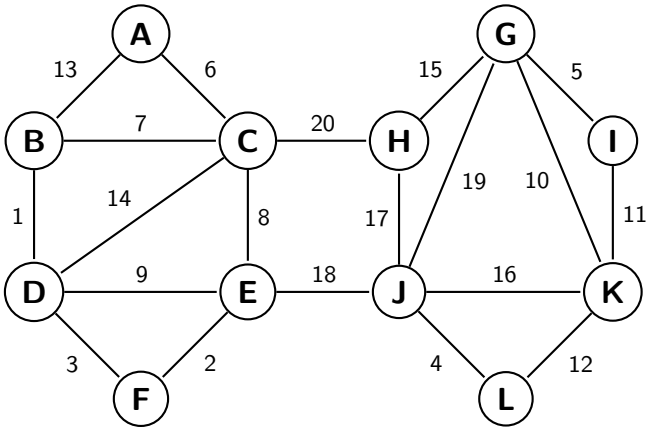
 Niklas Hempel (349492), Jan Knichel (377779), Paul Orschau (381085)

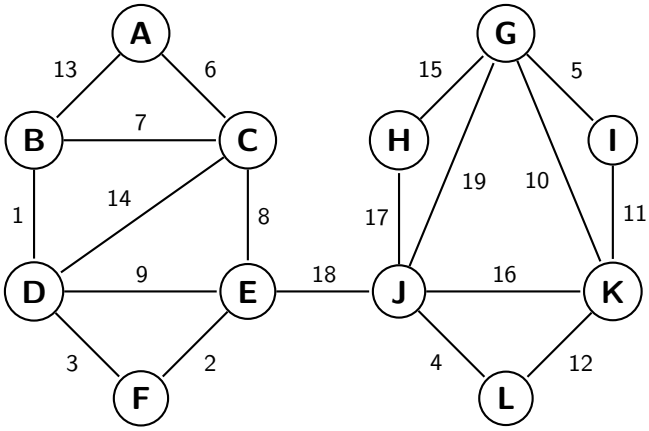
Aufgabe 1

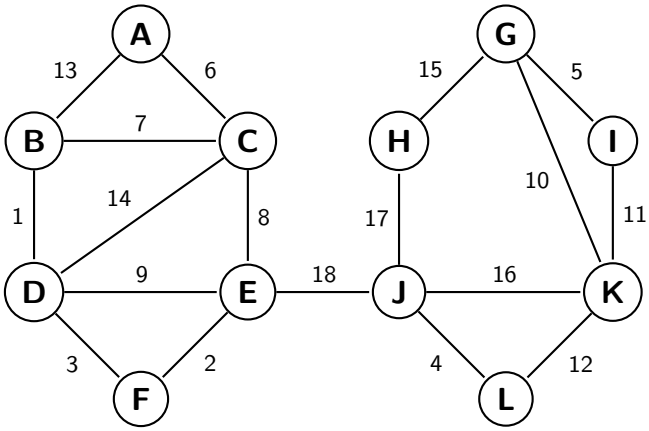
Teil a)

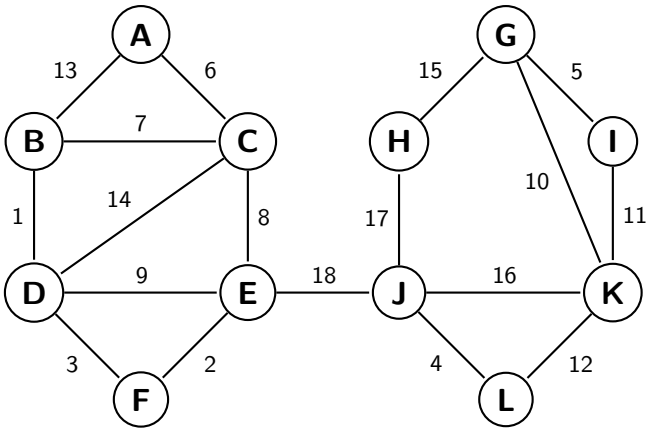
Der Algorithmus berechnet einen minimalen zusammenhängenden Graphen. Die größten Kanten, die nicht wichtig sind, um einen zusammenhängenden Graphen zu erhalten, werden gelöscht.
Dies geht daraus hervor, dass in jeder Schleifeniteration die Kante mit dem größten Gewicht gelöscht wird, falls der Graph danach noch zusammenhängend ist.

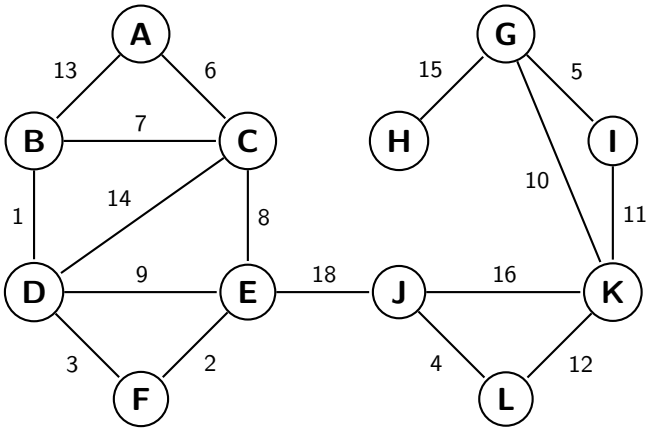
Teil b)

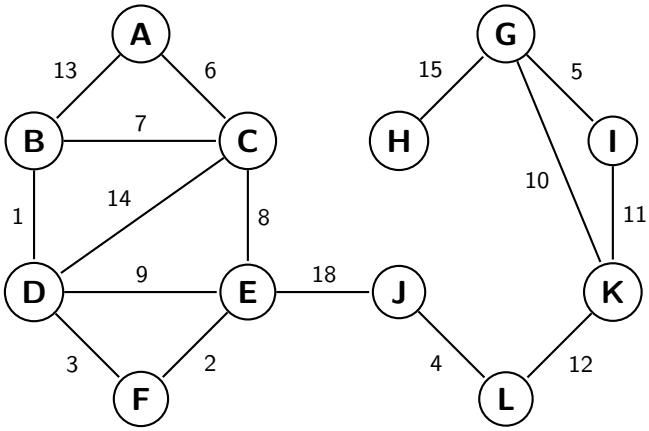


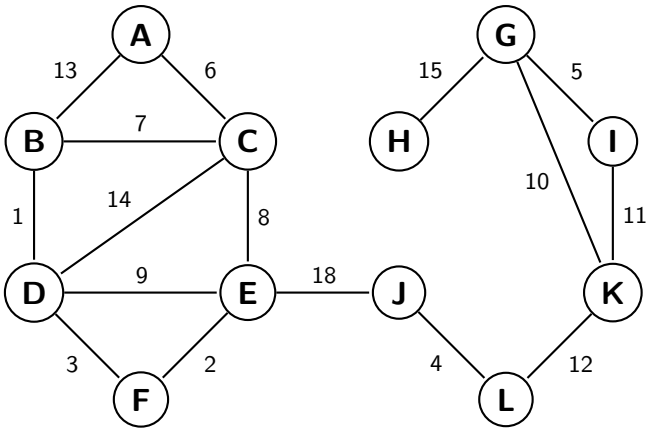


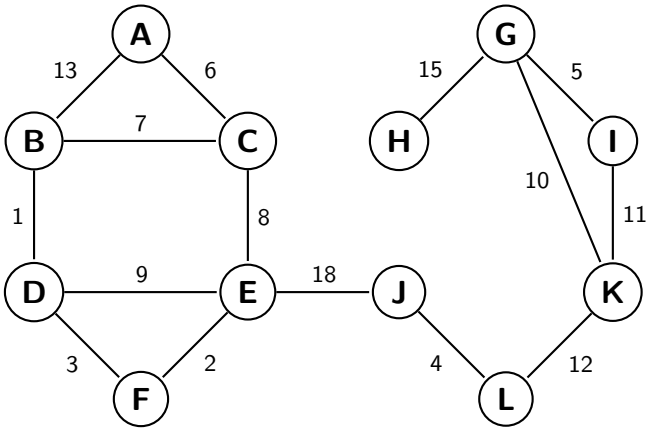

A
G
13
6
15
5
B
7
C
H
I
10
14
1
8
17
11
D
9
E
18
J
16
K
3
2
4
12
F
L

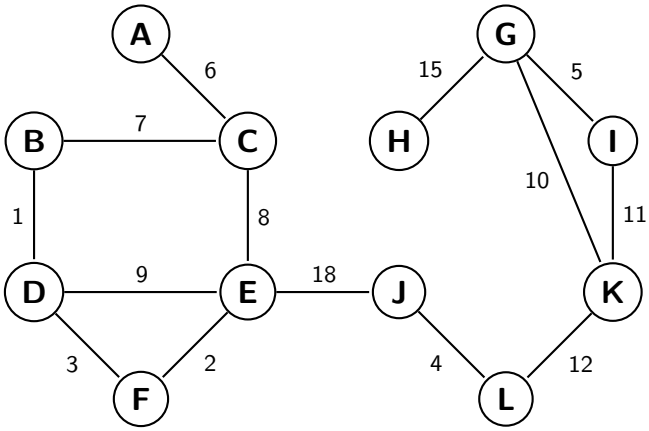

A
G
13
6
15
5
B
7
C
H
I
10
14
1
8
17
11
D
9
E
18
J
16
K
3
2
4
12
F
L

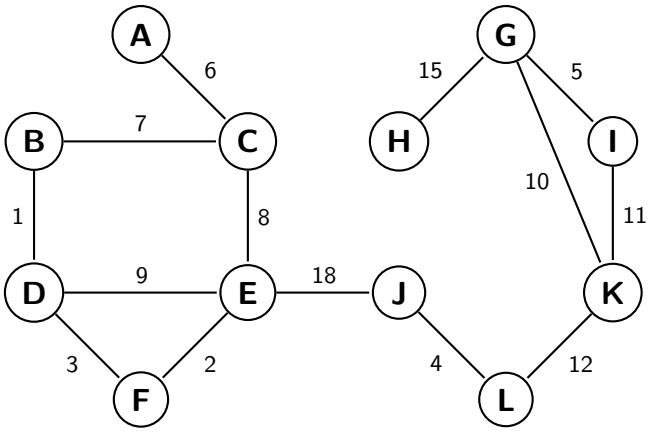

A
G
13
6
15
5
B
7
C
H
I
10
14
1
8
11
D
9
E
18
J
16
K
3
2
4
12
F
L

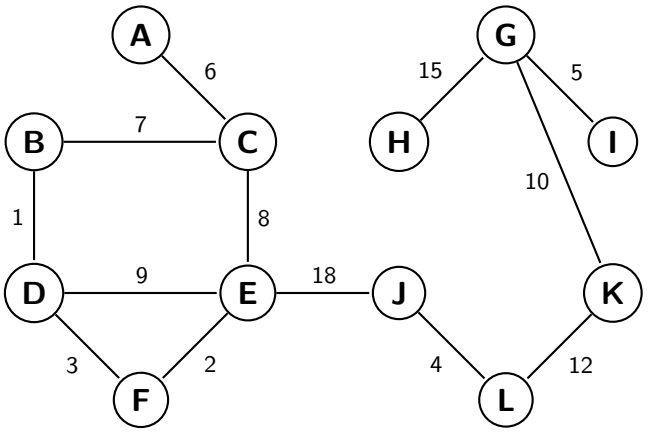

A
G
13
6
15
5
B
7
C
H
I
14
10
1
8
11
D
9
E
18
J
K
3
2
4
12
F
L

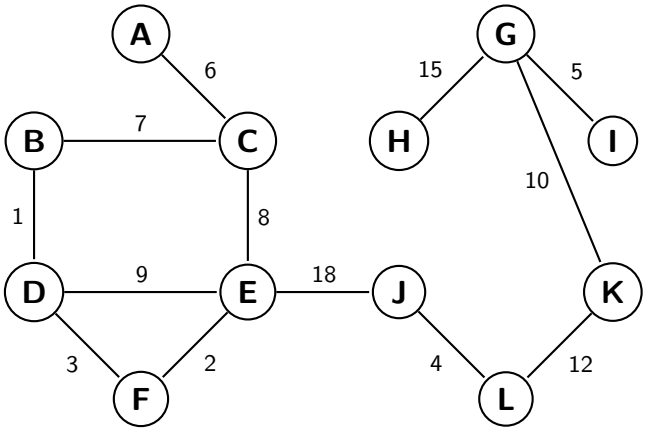

A
G
13
6
15
5
B
7
C
H
I
14
10
1
8
11
D
9
E
18
J
K
3
2
4
12
F
L

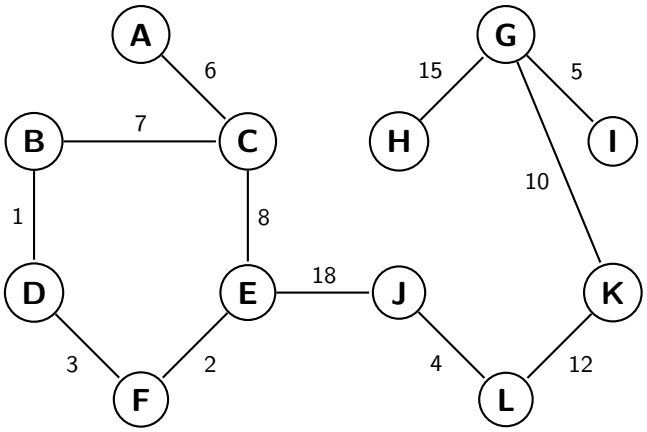

A
G
13
6
15
5
B
7
C
H
I
14
10
1
8
11
D
9
E
18
J
K
3
2
4
12
F
L


A
G
6
15
5
7
B
C
H
I
10
1
8
11
D
9
E
18
J
K
3
2
4
12
F
L

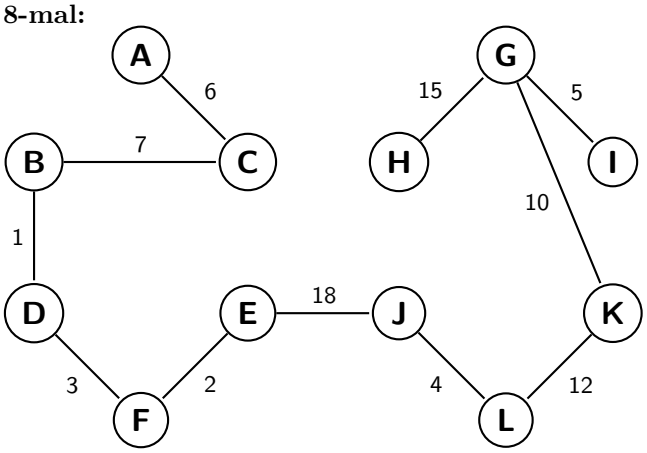

A
G
6
15
5
7
B
C
H
I
10
1
8
11
D
9
E
18
J
K
3
2
4
12
F
L


A
G
6
15
5
7
B
C
H
I
10
1
8
D
9
E
18
J
K
3
2
4
12
F
L


A
G
6
15
5
7
B
C
H
I
10
1
8
D
9
E
18
J
K
3
2
4
12
F
L


A
G
6
15
5
7
B
C
H
I
10
1
8
D
E
18
J
K
3
2
4
12
F
L

8-mal:


A
G
6
15
5
7
B
C
H
I
10
1
D
E
18
J
K
3
2
4
12
F
L

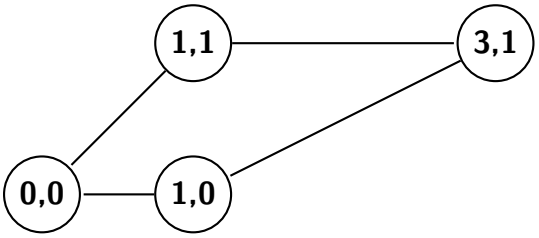
Aufgabe 2

"Der minimale Spannbaum eines Graphen, dessen Kantengewichte alle paarweise verschieden sind ist eindeutig."

Beweis: Durch Widerspruch
Angenommen, der minimale Spannbaum eines solchen Graphen wäre nicht eindeutig. Dann gäbe es mindestens zwei minimale Spannbäume, A und B. Da A und B unterschiedlich sind, muss es Kanten geben, die in einem der beiden Bäume, aber nicht im anderen enthalten sind. Da alle Kantengewichte paarweise verschieden sind, können wir unter all diesen Kanten diejenige mit dem niedrigsten Kantengewicht eindeutig auswählen und a nennen.
O.B.d.A. sei a in A und nicht in B, ansonsten benenne die Graphen um.
Da B auch ein Spannbaum ist, muss $B \cup a$ einen Kreis C enthalten. A (als Spannbaum) hat keine Kreise, daher muss C eine Kante b enthalten, die nicht in A enthalten ist. Weil wir festgelegt haben, dass a in A liegt, müssen b und a zwei unterschiedliche Kanten sein. b kann also nur in B liegen.
Da a die Kante mit dem kleinsten Kantengewicht ist, die nur in einem der beiden Graphen enthalten ist, muss $W(a) < W(c)$ gelten, denn alle Kantengewichte sind paarweise verschieden. Ersetzt man in B die Kante b durch a, ist der resultierende Spannbaum B' kleiner als B. Dies widerspricht der Annahme, dass B ein minimaler Spannbaum war, und damit ist die ursprüngliche These bewiesen. □

Aufgabe 3

1. $h(v) = \|K(v) - K(t)\|_1$
Gegenbeispiel:



Es gibt zwei Möglichkeiten, von $s(0,0)$ zu $t(3,1)$ zu gelangen. Einmal über $v'(1,1)$ oder über $v''(1,0)$.
1) über v': $\sqrt{1^2+1^2}+2 = \sqrt{2}+2 \approx 3,41$
2) über v'': $\sqrt{2^2+1^2}+1 = \sqrt{5}+1 \approx 3,24$
`ExtractMin` wält nun das Minimum der Funktion $h(v) = \|K(v) - K(t)\|_1 + dist(v)$ aus.
Für v' ist dies:

$$\|K(v') - K(t)\|_1 + dist(v') = \left\| \begin{pmatrix} 1-3 \\ 1-1 \end{pmatrix} \right\|_1 + \sqrt{2} = 2 + 0 + \sqrt{2} = 2 + \sqrt{2}$$

Für v'' ist dies:

$$\|K(v'') - K(t)\|_1 + dist(v'') = \left\| \begin{pmatrix} 1-3 \\ 0-1 \end{pmatrix} \right\|_1 + 1 = 2 + 1 + 1 = 4$$

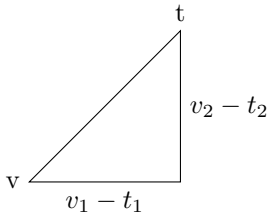
Da $4 > 2 + \sqrt{2}$ würde hier der Weg über v' gewählt werde, der Weg über v'' ist aber, wie oben berechnet, kürzer.

Aufgabe 3 wird auf nächster Seite fortgesetzt...

2. $h(v) = \|K(v) - K(t)\|_\infty$

Der klassische Dijkstra-Algorithmus berechnet den kürzesten Abstand von Knoten s zu anderen beliebigen Knoten.

Hier fließt nun statt der min. Distanz $dist(v)$ zum Startknoten s auch der minimale Abstand (als Maximumnorm) zum gesuchten Endknoten ein. Die Maximumnorm gibt den Abstand zwischen v und dem Zielknoten t in vertikaler oder horizontaler Ebene zurück, je nachdem, welcher größer ist. Dies entspricht dem minimalen möglichen Abstand zwischen diesen Knoten (v und t liegen auf einer Höhe/Breite), bzw. der größten Kathete des Dreiecks:



Da nun das Minimum der Funktion $\|K(v) - K(t)\|_\infty + dist(v)$ gesucht wird, findet man genau den Knoten v, für den die Distanz von v zu s addiert mit dem kleinsten möglichen Luftlinienabstand (= die grösste Kathetenlänge) ($v_1 - t_1$ oder $v_2 - t_2$) minimal wird.

Wird die grösste Kathetenlänge minimiert, wird auch die Hypotenuse minimiert, die der euklidschen Norm, bzw. dem Luftlinienabstand entspricht, der hier als unteres Maß für das Gewicht definiert ist.

Es gilt:

$$\begin{aligned}
&min_{v \in Q} \quad \|K(v) - K(t)\|_\infty + dist(v) \\
&= min_{v \in Q} \quad max((v_1 - t_1), (v_2 - t_2)) + dist(v) \\
&\overset{(*)}{\leq} min_{v \in Q} \quad \sqrt{(v_1 - t_1)^2 + (v_2 - t_2)^2} + dist(v) \\
&= min_{v \in Q} \quad \|K(v) - K(t)\|_2 + dist(v) \\
&\leq min_{v \in Q} \quad W(v,t) + dist(v) \\
&= dist(t)
\end{aligned} \tag{1}$$

Vom Startknoten ausgehend wird also immer der bestmögliche Nachfolger gefunden, bis man den Zielknoten t erreicht, den kürzesten Weg gefunden hat und der Algorithmus terminiert.

mit $(*)$:

$$\|x\|_\infty = \sqrt{(\|x\|_\infty)^2} = \sqrt{|x_{max}|^2} \leq \sqrt{|x_1|^2 + ... + |x_n|^2} = \|x\|_2$$

Aufgabe 4

Das Problem lässt sich durch einen ungerichteten Graphen darstellen, bei dem jede Kante einer **Verschiebung** im Puzzle entsprich und jeder Knoten einen **eindeutigen Zustand des Puzzles** beschreibt. Der abgebildete Zustand eines 3x3 Puzzles wäre ein Knoten in diesem Graph, vom dem genau 3 Kanten ausgehen, jeweils für das Verschieben des Puzzleteils 2, 7 oder 4. Jeder Knoten im Graph kann höchstens 4 adjazente Knoten haben, da zu jedem Zeitpunkt nur höchstens 4 nächste Verschiebungen möglich sind. Der Graph ist ungerichtet, da sich jeder Zug immer rückgängig machen lässt.

Um das Problem zu lösen, muss man den aktuellen Zustand des Puzzles als Knoten im zugehörigen Graphen finden (s), und dann den gewünschten Zielzustand bzw. -knoten ausfindig machen (t). Hat man beide Knoten, so lässt sich das Problem mit einem Single-Source Shortest Path Algorithmus wie z.B. Dijktra lösen, die Lösung ist der kürzeste Pfad zwischen den Knoten/Zuständen, denn er verwendet möglichst wenige Züge/Verschiebungen/Kanten.

Jedes der $n^2 - 1$ Puzzleteile sowie die "Lücke" brauchen eine eindeutige Position auf dem Schiebebrett. Jede Anordnung ist erlaubt. Für das erste Teil gibt es n^2 mögliche Positionen, für das nächste nur noch $n^2 - 1$ Positionen, usw. bis irgendwann das letzte Stück nur noch 2 mögliche Positionen hat, die Lücke ist danach eindeutig festgelegt. Die Anzahl von möglichen Zuständen errechnet sich also durch:

$$n^2 * (n^2 - 1) * (n^2 - 3) * \cdots * 2 * 1 = n^2! \tag{2}$$

Aufgabe 5

	A	D	E	B	F	G	H
A	0	0	0	0	0	0	0
B	6	6	6	6	6	6	6
C	∞	∞	∞	17	17	17	17
D	3	3	3	3	3	3	3
E	3	3	3	3	3	3	3
F	∞	6	6	6	6	6	6
G	∞	12	7	7	7	7	7
H	∞	11	11	11	11	11	11

Aufgabe 6

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	$\underline{0}$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
2	0	4	3	∞	∞	9	∞	∞	∞	$\underline{1}$	∞	∞	∞
3	0	4	$\underline{3}$	∞	∞	9	∞	∞	∞	1	∞	∞	∞
4	0	4	3	4	$\underline{3}$	9	∞	∞	∞	1	∞	∞	∞
5	0	$\underline{4}$	3	4	3	5	∞	∞	∞	1	∞	∞	∞
6	0	4	3	4	3	5	∞	$\underline{3}$	3	1	∞	∞	∞
7	0	4	3	4	3	5	∞	3	$\underline{3}$	1	∞	∞	∞
8	0	4	3	$\underline{4}$	3	5	∞	3	3	1	∞	∞	∞
9	0	4	3	4	3	5	$\underline{3}$	3	3	1	∞	∞	∞
10	0	4	3	4	3	$\underline{5}$	3	3	3	1	∞	∞	∞
11	0	4	3	4	3	5	3	3	3	1	$\underline{4}$	∞	4
12	0	4	3	4	3	5	3	3	3	1	4	4	$\underline{3}$
13	0	4	3	4	3	5	3	3	3	1	4	$\underline{4}$	3