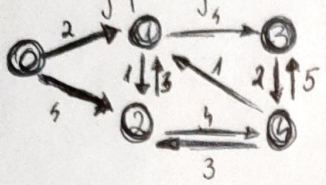
practical work no. 3

1. write a program that, given a graph with positive costs and two vertices, finds a lowest cost walk between the given vertices, using the Dijkstra algorithm.

- we're using vertex 1 as the starting vertex and vertex 2 as the ending vertex

random_graph_dijkstra_1.txt



vertex 1 = 0
vertex 2 = 2

	x	y	dist: dictionary	q: priority queue	prev: dictionary
initialization			0 | 1ex | 1ex | 1ex | 1ex	(0,0)	-1 | -1 | -1 | -1 | -1
iteration 1	0				
iteration 1.1		1	0 | 2 | 1ex | 1ex | 1ex	(2,1)	-1 | 0 | -1 | -1 | -1
iteration 1.2.		2	0 | 2 | 4 | 1ex | 1ex	(2,1) | (4,2)	-1 | 0 | 0 | -1 | -1
iteration 2	1			(4,2)	
relaxation ← iteration 2.1		2	0 | 2 | 3 | 1ex | 1ex	(3,2)	-1 | 0 | 1 | -1 | -1
iteration 2.2		3	0 | 2 | 3 | 6 | 1ex	(3,2) | (6,3)	-1 | 0 | 1 | 1 | -1
iteration 3	2			(6,3)	

x == vertex 2 ⇒ found = True ⇒ we got a walk

the minimum cost walk from vertex 1 = 0 to vertex 2 = 2 is built backwards from prev:

vertex 2 = 2, prev[2] = 1, prev[1] = 0 = vertex 1 ⇒ 0 → 1 → 2

cost = dist[2] = 3

vertex 1 = 4
vertex 2 = 0

	x	y	dist: dictionary	q: priority queue	prev: dictionary
initialization			1ex | 1ex | 1ex | 1ex | 0	(0,4)	-1 | -1 | -1 | -1 | -1
iteration 1	4				
iteration 1.1		1	1ex | 1ex | 1ex | 1ex | 0	(1,1)	-1 | 4 | -1 | -1 | -1
iteration 1.2		2	1ex | 7 | 3 | 1ex | 0	(1,1) | (3,2)	-1 | 4 | 4 | -1 | -1
iteration 1.3		3	1ex | 1 | 3 | 5 | 0	(1,1) | (3,2) | (5,3)	-1 | 4 | 4 | 4 | -1
iteration 2	1			(3,2) | (5,3)	
relaxation ← iteration 2.1		2	1ex | 1 | 2 | 5 | 0	(2,2) | (5,3)	-1 | 4 | 1 | 4 | -1
iteration 2.2		3	1ex | 1 | 2 | 5 | 0	(2,2) | (5,3)	-1 | 4 | 1 | 4 | -1
iteration 3	2			(5,3)	
iteration 3.1		1	1ex | 1 | 2 | 5 | 0	(5,3)	-1 | 4 | 1 | 4 | -1
iteration 3.2		4	1ex | 1 | 2 | 5 | 0	(5,3)	-1 | 4 | 1 | 4 | -1
iteration 4	3				
iteration 4.1		4	1ex | 1 | 2 | 5 | 0		-1 | 4 | 1 | 4 | -1

q is empty ⇒ found = False ⇒ we got no walk

graph1K.txt

1 → 100

1 → 5 → 487 → 175 → 714 → 799 → 222 → 561 → 100 | 111

100 → 1

100 → 259 → 229 → 641 → 538 → 854 → 1 | 186

graph10K.txt

1 → 7317 → 460 → 6010 → 5295 → 4560 → 5513 → 8467 → 3517 → 99 → 9159 → 6840 → 5177 → 7133 → 288 → 100 | 344

100 → 4442 → 3980 → 1974 → 407 → 4489 → 5162 → 2008 → 3631 → 2305 → 8336 → 1 | 238

graph100K.txt

1 → 99842 → 59480 → 5210 → 19068 → 66428 → 33692 → 97073 → 23675 → 73057 → 100 | 304

100 → 85636 → 77467 → 52472 → 38155 → 40962 → 34650 → 29215 → 35260 → 1 | 361