University of Regina
Department of Computer Science

Bao Cao
200363431

CS 340 - Assignment 4

Exercise 9.1

s G D A B H E I F C t

Exercise 9.3

Source code and explanation are submitted as cpp file separately.

Exercise 9.5

a.

v	known	d_v	p_v
A	F	0	0
B	F	∞	0
C	F	∞	0
D	F	∞	0
E	F	∞	0
F	F	∞	0
G	F	∞	0

v	known	d_v	p_v
A	T	0	0
B	F	5	A
C	F	3	A
D	F	∞	0
E	F	∞	0
F	F	∞	0
G	F	∞	0

v	known	d_v	p_v
A	T	0	0
B	F	5	A
C	T	3	A
D	F	10	C
E	F	10	C
F	F	∞	0
G	F	∞	0

v	known	d_v	p_v
A	T	0	0
B	T	5	A
C	T	3	A
D	F	10	C
E	F	8	B
F	F	∞	0
G	F	6	B

University of Regina
Department of Computer Science

Bao Cao
200363431

v	known	d_v	p_v
A	T	0	0
B	T	5	A
C	T	3	A
D	F	10	C
E	F	7	G
F	F	∞	0
G	T	6	B

v	known	d_v	p_v
A	T	0	0
B	T	5	A
C	T	3	A
D	F	9	E
E	T	7	G
F	F	8	E
G	T	6	B

v	known	d_v	p_v
A	T	0	0
B	T	5	A
C	T	3	A
D	F	9	E
E	T	7	G
F	T	8	E
G	T	6	B

v	known	d_v	p_v
A	T	0	0
B	T	5	A
C	T	3	A
D	T	9	E
E	T	7	G
F	T	8	E
G	T	6	B

Shortest paths from A to other vertices:
AB
AC
ABGED
ABGE
ABGEF
ABG

b.

v	known	d_v	p_v
A	F	∞	0
B	F	0	0
C	F	∞	0
D	F	∞	0
E	F	∞	0
F	F	∞	0
G	F	∞	0

v	known	d_v	p_v
A	F	∞	0
B	T	0	0
C	F	1	B
D	F	∞	0
E	F	1	B
F	F	∞	0
G	F	1	B

v	known	d_v	p_v
A	F	∞	0
B	T	0	0
C	T	1	B
D	F	2	C
E	T	1	B
F	F	2	E
G	T	1	B

v	known	d_v	p_v
A	F	3	D
B	T	0	0
C	T	1	B
D	T	2	C
E	T	1	B
F	T	2	E
G	T	1	B

Bao Cao
200363431

v	known	d_v	p_v
A	T	3	D
B	T	0	0
C	T	1	B
D	T	2	C
E	T	1	B
F	T	2	E
G	T	1	B

Shortest paths from B to other vertices:
BCDA
BC
BCD
BE
BEF
BG

Exercise 9.6

The worst-case running time of Dijkstra's algorithm when implemented with d-heaps is $O(|E| log_d |V|)$

Exercise 9.8

The distances of vertices are kept in a priority queue. At any time, the keys in the priority queue are in the set {D + 1, D + 2, … , D + |E|}, where D is the value of the last key removed from the priority queue. Since the sequence of keys removed by the algorithm is increasing, every key is at least D + 1 and at most D + |E|.
The priority queue can be implemented with a circular array A[0…|E|-1] of size |E|. Each vertex with key k in the bucket in cell A[h(k)] where h(k) = k mod(|E|) + 1. Operations on priority queue will be modified as follow:
- delete-min: while A[h(D)] is empty, increment D. Then delete and return a vertex from A[h(D)].
- insert with key k: Add to A[h(k)].
- decrease-key k to k': Move the vertex from A[h(k)] to A[h(k')]

Insert and decrease-key are constant-time operators, so the total time spent in those operations will be O(|V| + |E|). The total time spent in delete-min will be O(|V|) plus the final value of D. The final value of D is the maximum distance from the source to any vertex. The maximum distance is at most |E|(|V| - 1) because each path has at most |V| - 1 edges. Thus, the total time spent by the algorithm is O(|E||V| + |E|).

Exercise 9.9 Source code and explanation are submitted as cpp file separately.

Exercise 9.15

Prim's algorithm

University of Regina
Department of Computer Science

Bao Cao
200363431

v	known	d_v	p_v
A	F	0	0
B	F	∞	0
C	F	∞	0
D	F	∞	0
E	F	∞	0
F	F	∞	0
G	F	∞	0
H	F	∞	0
I	F	∞	0
J	F	∞	0

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	F	10	B
D	F	4	A
E	F	2	B
F	F	3	B
G	F	∞	0
H	F	∞	0
I	F	∞	0
J	F	∞	0

v	known	d_v	p_v
A	T	0	0
B	F	3	A
C	F	∞	0
D	F	4	A
E	F	4	A
F	F	∞	0
G	F	∞	0
H	F	∞	0
I	F	∞	0
J	F	∞	0

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	F	10	B
D	F	4	A
E	T	2	B
F	F	3	B
G	F	∞	0
H	F	2	E
I	F	1	E
J	F	∞	0

University of Regina
Department of Computer Science

Bao Cao
200363431

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	F	10	B
D	F	4	A
E	T	2	B
F	F	3	B
G	F	∞	0
H	F	2	E
I	T	1	E
J	F	7	I

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	F	10	B
D	F	4	A
E	T	2	B
F	F	3	B
G	F	∞	0
H	T	2	E
I	T	1	E
J	F	7	I

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	F	6	F
D	F	4	A
E	T	2	B
F	T	3	B
G	F	2	F
H	T	2	E
I	T	1	E
J	F	7	I

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	F	1	G
D	F	4	A
E	T	2	B
F	T	3	B
G	T	2	F
H	T	2	E
I	T	1	E
J	F	7	I

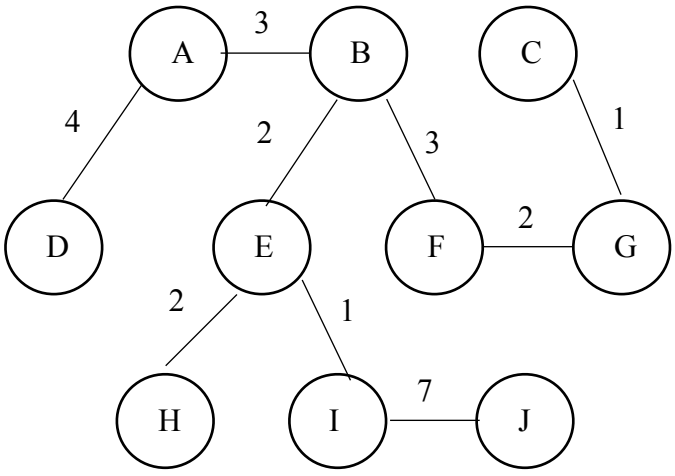
University of Regina
Department of Computer Science

Bao Cao
200363431

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	T	1	G
D	F	4	A
E	T	2	B
F	T	3	B
G	T	2	F
H	T	2	E
I	T	1	E
J	F	7	I

v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	T	1	G
D	T	4	A
E	T	2	B
F	T	3	B
G	T	2	F
H	T	2	E
I	T	1	E
J	F	7	I

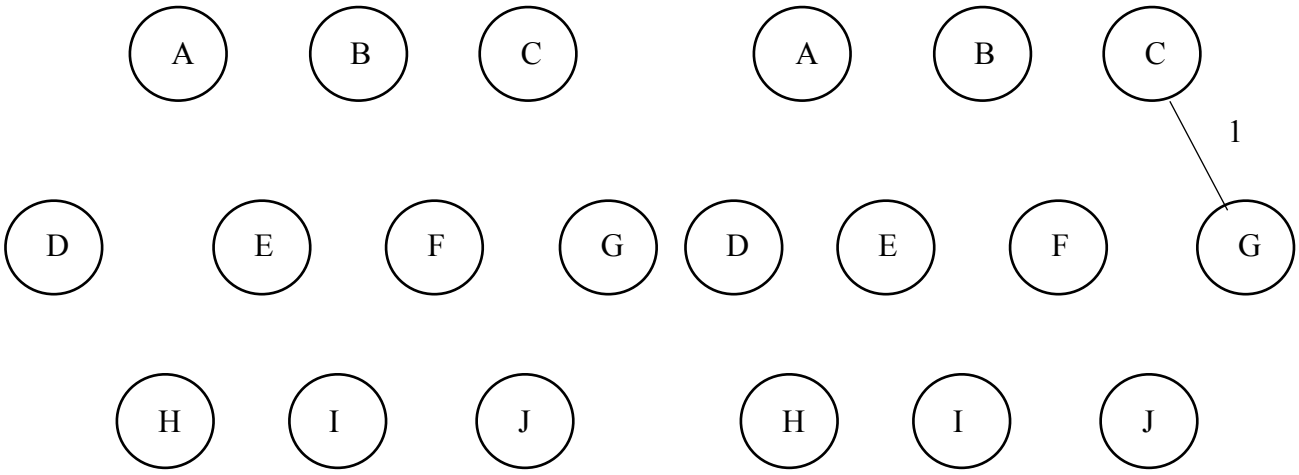
v	known	d_v	p_v
A	T	0	0
B	T	3	A
C	T	1	G
D	T	4	A
E	T	2	B
F	T	3	B
G	T	2	F
H	T	2	E
I	T	1	E
J	T	7	I



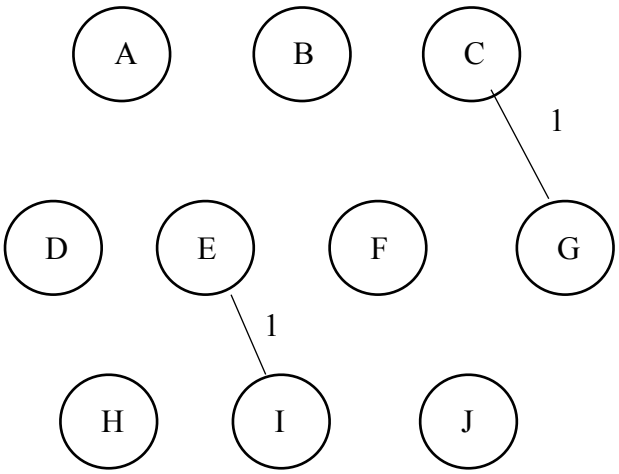
University of Regina
Department of Computer Science

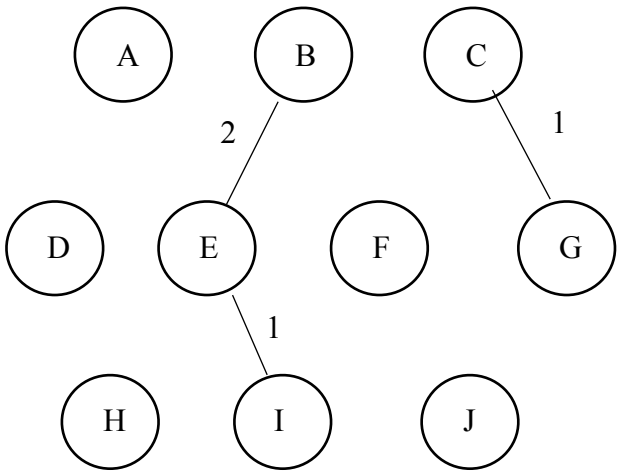
Bao Cao
200363431

Kruskal's algorithm

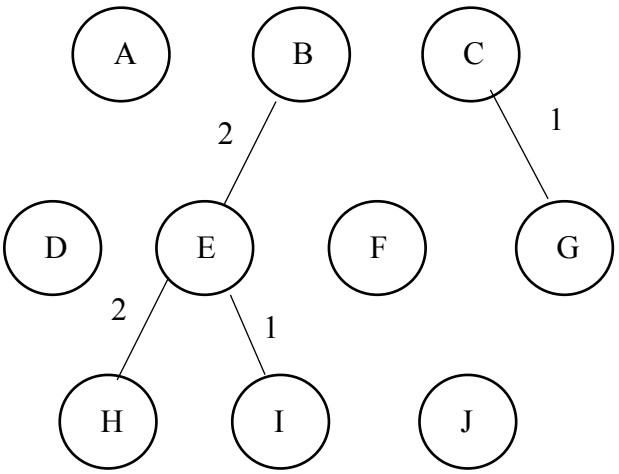


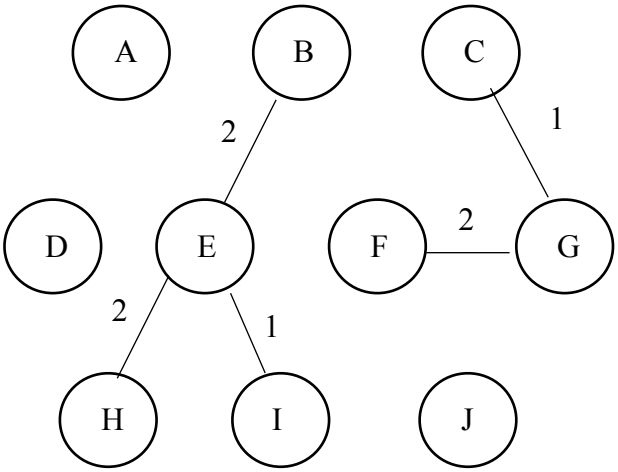
Edge	Weight	Action
(C,G)	1	Accepted
(E, I)	1	Accepted
(B,E)	2	Accepted
(E,H)	2	Accepted
(F,G)	2	Accepted
(A,B)	3	Accepted
(B,F)	3	Accepted
(F,I)	3	Rejected
(A,D)	4	Accepted
(A,E)	4	Rejected
(H,I)	4	Rejected
(D,E)	5	Rejected
(C,F)	6	Rejected
(D,H)	6	Rejected
(I,J)	7	Accepted

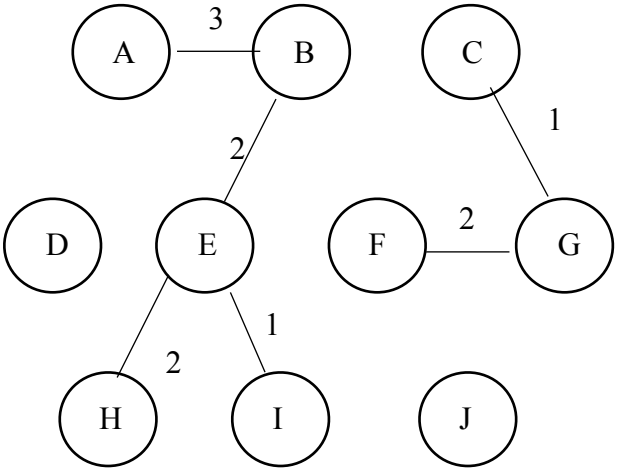


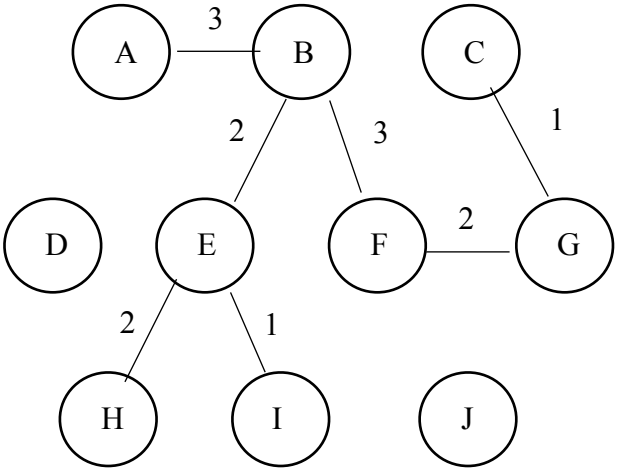


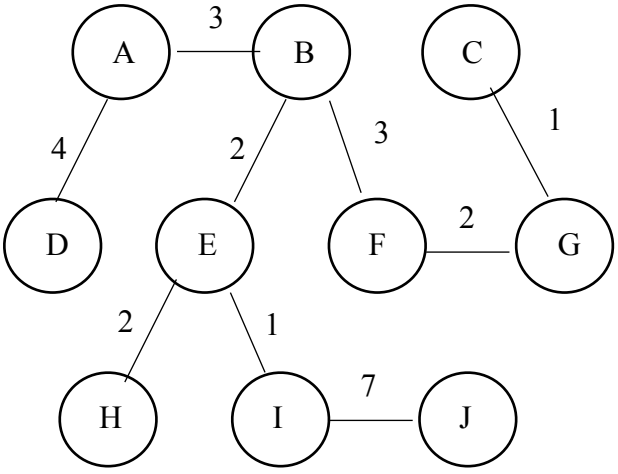
Bao Cao
200363431











This minimum spanning tree is unique. Because the maximum weight edge in any cycle of the graph is unique, there is only one way to remove the cycle, which makes the MST is unique.

Exercise 9.21

Articulation points: C, E, F

University of Regina
Department of Computer Science

Bao Cao
200363431