Alexis Martinez

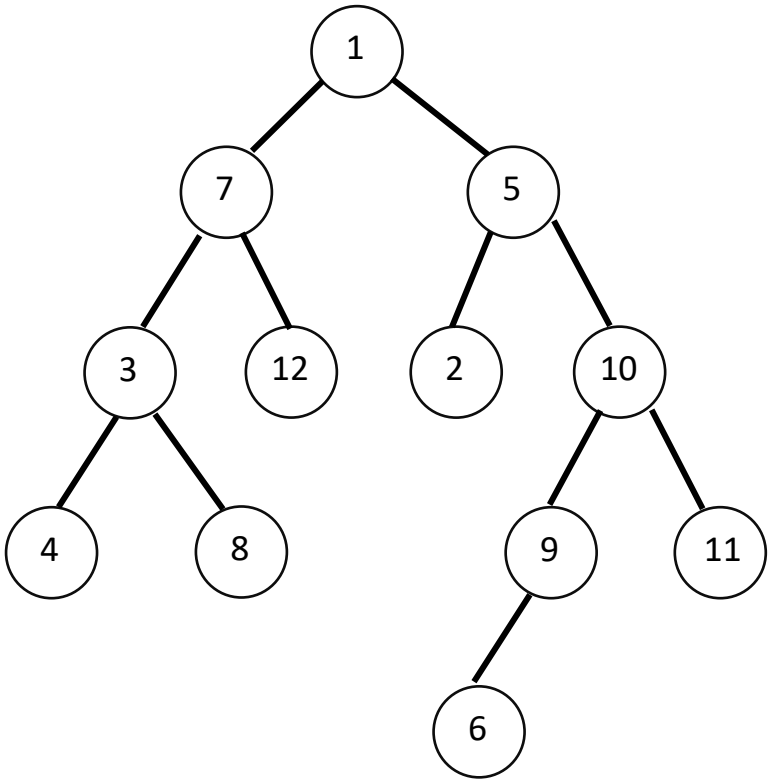
Homework #8

CSE 2321

12/07/2021

1.

(a)



Queue: <1>

i = 1

Nodes	1	2	3	4	5	6	7	8	9	10	11	12
MARK	1	0	0	0	0	0	0	0	0	0	0	0

Queue: <7, 5>

i = 1

Nodes	1	2	3	4	5	6	7	8	9	10	11	12
MARK	1	0	0	0	1	0	1	0	0	0	0	0

Queue: <5| 3, 12>

i = 7

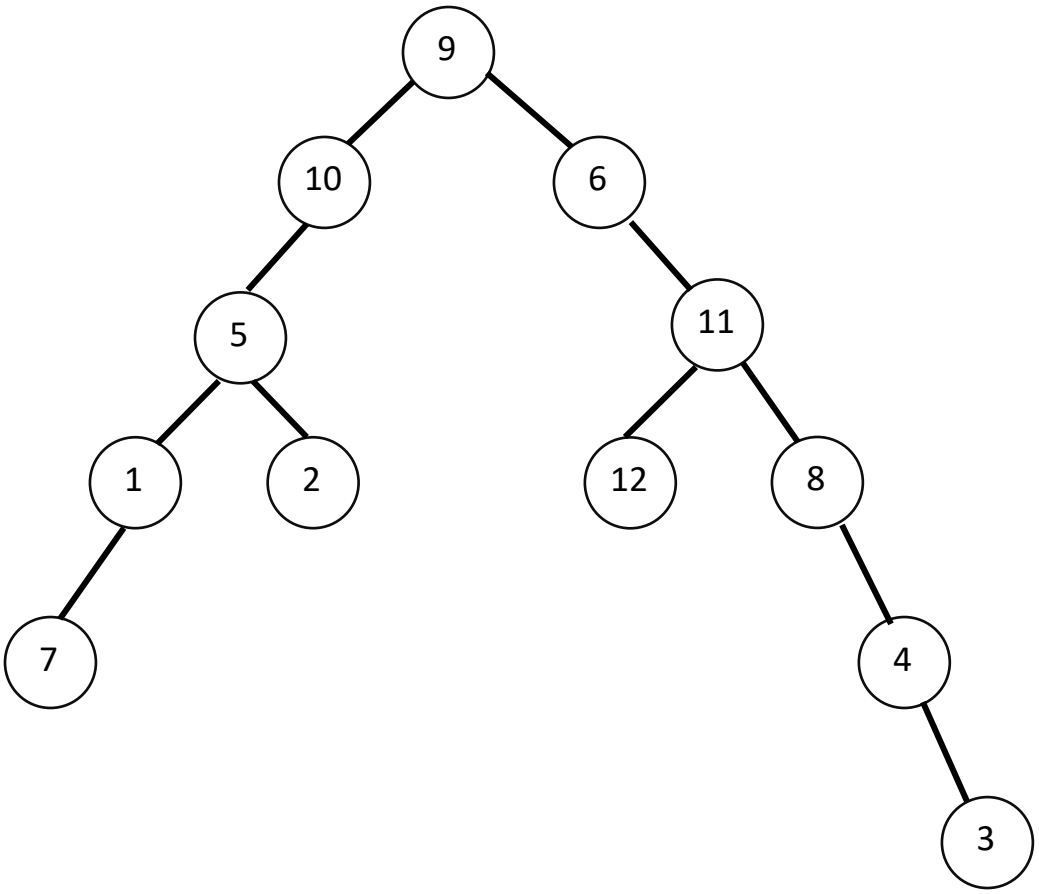
Nodes	1	2	3	4	5	6	7	8	9	10	11	12
MARK	1	0	1	0	1	0	1	0	0	0	0	1

…Until Queue is empty and all nodes have been visited

(b)

Nodes	1	2	3	4	5	6	7	8	9	10	11	12
PATH LENGTH	0	2	2	3	1	4	1	3	3	2	3	2

(c)



Queue: <9>

i = 9

Nodes	1	2	3	4	5	6	7	8	9	10	11	12
MARK	0	0	0	0	0	0	0	0	1	0	0	0

Queue: <10, 6>

i = 9

Nodes	1	2	3	4	5	6	7	8	9	10	11	12
MARK	0	0	0	0	0	1	0	0	1	1	0	0

Queue: <6|5>

i = 10

Nodes	1	2	3	4	5	6	7	8	9	10	11	12
MARK	0	0	0	0	1	1	0	0	1	1	0	0

…Until Queue is empty and all nodes have been visited

(d)

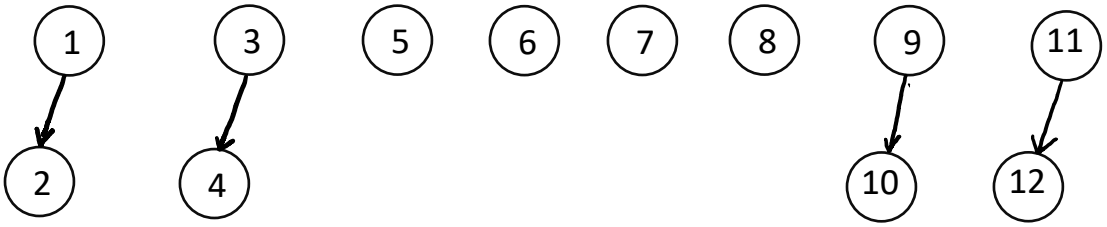
Nodes	1	2	3	4	5	6	7	8	9	10	11	12
PATH LENGTH	3	3	5	4	2	1	4	3	0	1	2	3

2.

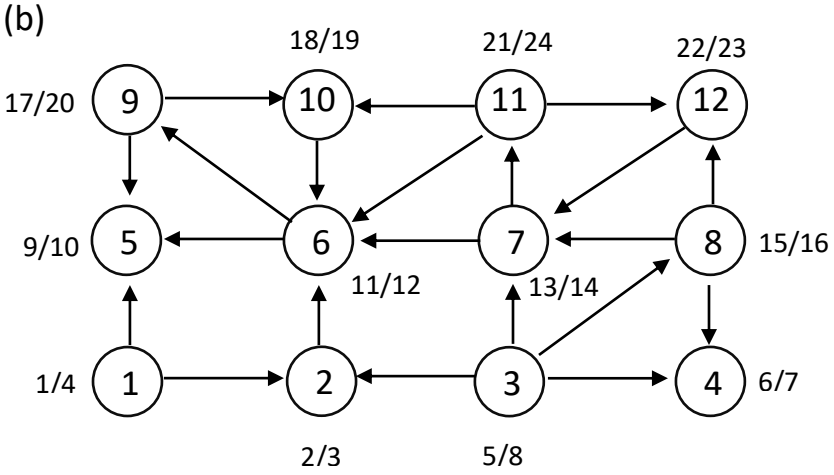
```
for each vertex vi of G do G.V[i].mark ← not-visited;
q.Init() ; /* q is a queue of vertices */
G.V[k].mark ← visited;
G.V[k].pathlen ← 0;
G.V[k].num_paths ← 0;
```

```
q.Enqueue(k);
while (q.Length() > 0) do
        i ← q.Dequeue();
        foreach edge (i, j) incident on vj do
                If (G.V[j].mark = not-visited) then
                        q.Enqueue(j);
                        G.V[j].mark ← visited;
                        G.V[j].pathlen ← G.V[i].pathlen + 1;

                else {

                        G.V[j].num_paths ← G.V[i].num_paths + 1;

                }
```

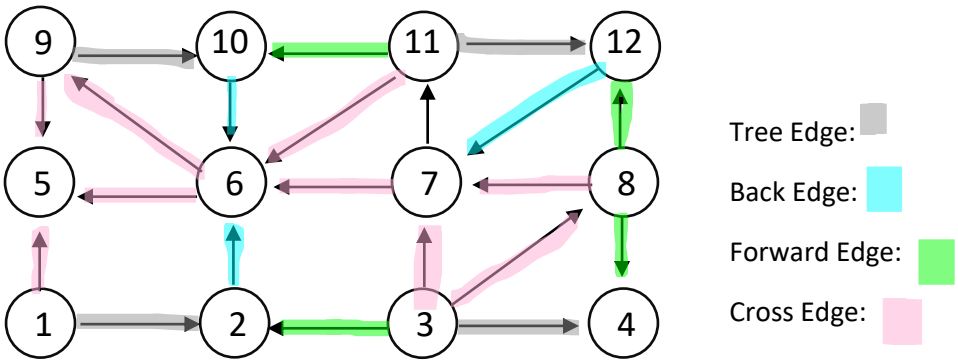
3. (a)



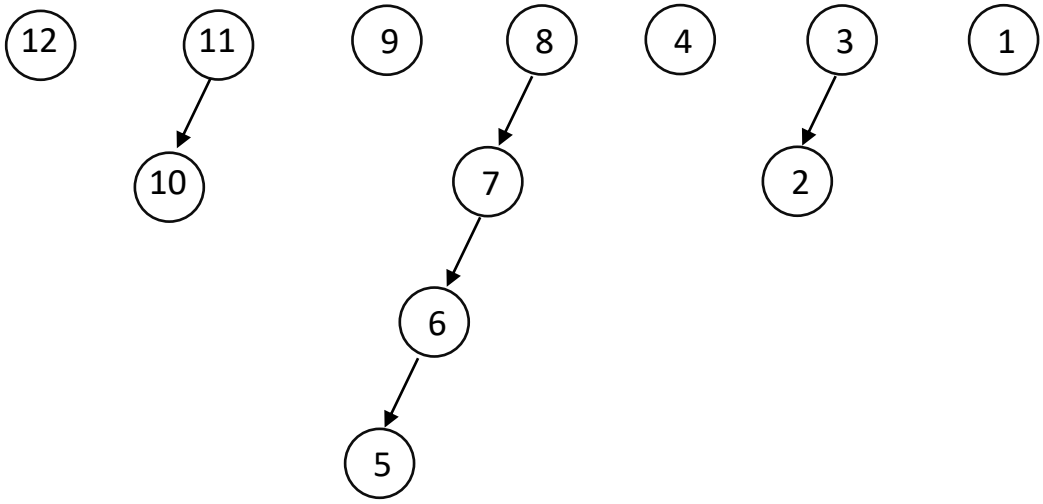
(b)



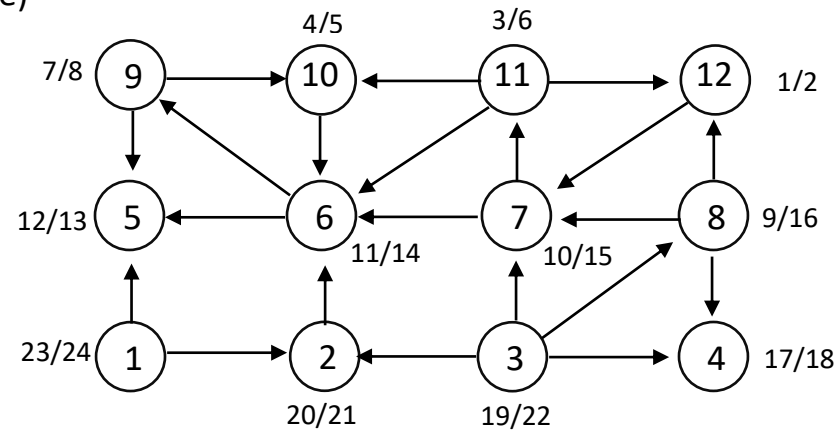
(c)


9
10
11
12
5
6
7
8
1
2
3
4
Tree Edge:
Back Edge:
Forward Edge:
Cross Edge:

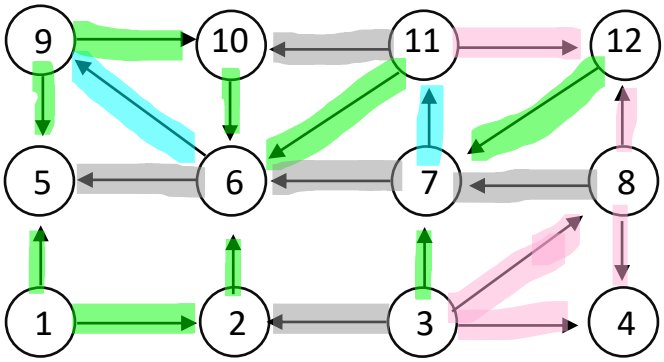
(d)


12
11
9
8
4
3
1
10
7
2
6
5

(e)


4/5
3/6
7/8
9
10
11
12
1/2
12/13
5
6
7
8
9/16
11/14
10/15
23/24
1
2
3
4
17/18
20/21
19/22

(f)

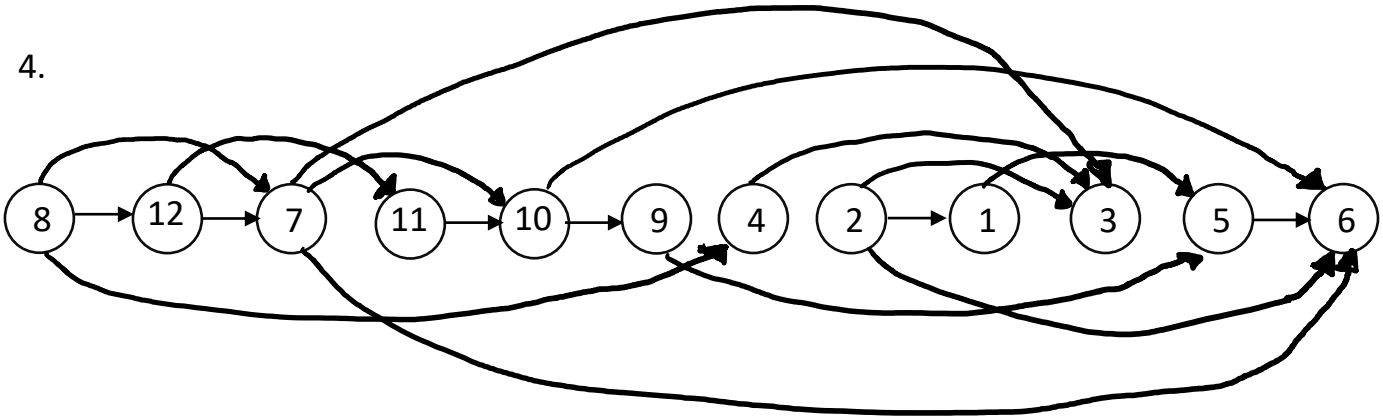


Back Edge:

Forward Edge:

Cross Edge:

4.



5. No, because there is a cycle between these nodes:

2-5-9-10-7-12-8-3-2