Hannah Madsen

Bipartite.java

1) What is the underlying data structure(s) used?

A boolean array (color)
Another boolean array (marked)
An integer array (edgeTo)
A stack of type Integer (cycle)

2) What is their purpose in the code?

A boolean array (color) - holds the color values of vertices and is used to determine whether vertice are in the same set
Another boolean array (marked) - holds the values of whether a vertice has already been visited
An integer array (edgeTo) - holds path of all the vertices
A stack of Integer type (cycle) - holds the path of all the vertices in an odd-length cycle

3) What is the main function doing to test the code?

The main function generates a graph with 13 nodes and 13 edges between those nodes that are composed of random integers, where the random values range from 0 to 26. Then prints out each node and its edges. Next it creates a new Bipartite object out of the graph. If the graph is bipartite, it prints out each node and its color. If the graph is not bipartite, then it must have and odd length cycle, so all the integers in the cycle are then printed out.

4) What (if any) other classes are used in the program?

Graph.java
GraphGenerator.java
Stack.java

5) What is those classes' purpose?

Graph.java - Used to store all the random integers as vertices, and their edges.
GraphGenerator.java - Used to create a graph in which our random integers can be stored in.
Stack.java - used to create an object that stores all the vertices in the odd length cycle

Hannah Madsen

Cycle.java

1) What is the underlying data structure(s) used?

A boolean array (marked)
An integer array (edgeTo)
A stack of Integer type (cycle)

2) What is their purpose in the code?

A boolean array (marked) - holds the values of whether a vertice has already been visited
An integer array (edgeTo) - holds path of all the vertices
A stack of Integer type (cycle) - holds all the vertices in a cycle, if a graph has one

3) What is the main function doing to test the code?

Creates an In class object to read in the data files, which is then used to create a Graph object. Next a Cycle object is created out of the Graph object. Then the hasCycle method is called to see if a cycle exists in the graph, and if a cycle is found to exist the integers in the cycle are printed out.

4) What (if any) other classes are used in the program?

Graph.java
Stack.java
In.java
StdOut.java

5) What is those classes' purpose?

Graph.java - Used to store all the data points in the file as vertices, and their edges between one another.
Stack.java - used to create an object that stores all the vertices in the cycle
In.java - used to read in the data in the files
StdOut.java - used to print out the data in the cycle(s)

Hannah Madsen

Screenshots of output:

Bipartite.java

Output for an odd length cycle

```
26 vertices, 26 edges
0: 10
1: 6 23 24 11
2:
3: 6 7
4: 19 8 24
5: 24 18 10
6: 7 1 3
7: 6 3
8: 4
9: 16
10: 0 5 14
11: 13 1
12: 15 14
13: 11 21
14: 16 10 12 24 17
15: 12
16: 14 9
17: 14
18: 5
19: 4
20: 24
21: 13
22: 25
23: 24 1
24: 5 23 1 20 4 14
25: 22

Graph has an odd-length cycle: 6 7 3 6

Process finished with exit code 0
```

Output for a bipartite graph

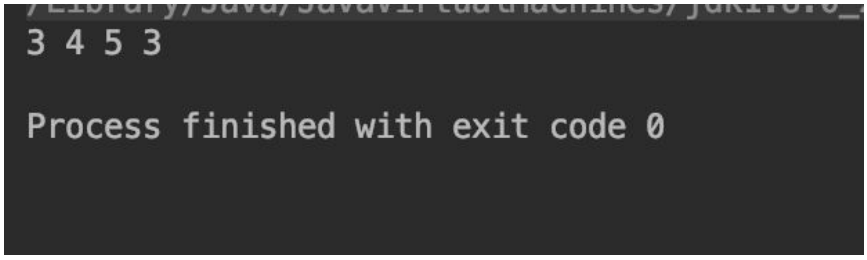
```
26 vertices, 26 edges
0: 5
1: 19
2: 19 11
3: 22
4: 25 24 25
5: 0 17
6: 25 8
7: 14 10
8: 21 19 6
9: 25
10: 7 19 11
11: 12 22 2 10
12: 11
13: 20 17 24
14: 7
15: 22
16:
17: 13 5
18:
19: 8 10 1 2
20: 23 13
21: 8
22: 15 23 11 3
23: 20 22
24: 13 4
25: 6 4 4 9
```

```
Graph is bipartite
0: false
1: false
2: false
3: true
4: true
5: true
6: true
7: true
8: false
9: true
10: false
11: true
12: false
13: true
14: false
15: true
16: false
17: false
18: false
19: true
20: false
21: true
22: false
23: true
24: false
25: false
```

Hannah Madsen

Cycle.java

tinyG.txt



mediumG.txt