CS 6301.502. Implementation of advanced data structures and algorithms

Fall 2017; Wed, Sep 20.

Long Project LP2: Euler tours in directed graphs

Ver 1.0: Initial description (Wed, Sep 20).

Due: 11:59 PM, Sun, Oct 8 (1st deadline), Sun, Oct 15 (2nd deadline).

Max excellence credits: 5.

Initial code base: ArrayIterator, Graph, GraphAlgorithm, Euler, LP2, Timer

* Submit before the first deadline to be eligible for excellence credit.
* For each group, only its last submission is kept and earlier submissions are discarded.
* Your code must be of good quality, and pass all test cases to earn excellence credits.
* Only the first 15 groups satisfying above conditions will be assigned excellence credits.

**Project Description**

A directed graph is called Eulerian if it is strongly connected, and the number of incoming edges at each node is equal to the number of outgoing edges from it. It is known that such a graph aways has a tour (a cycle that may not be simple) that goes through every edge of the graph exactly once. Such a tour (sometimes called a circuit) is called an Euler tour. In this project, write code that finds an Euler tour in a given graph using the algorithm described in class.

**Driver LP2.java**

Driver code for the project is provided. It works as follows. If a command line parameter is there, it is treated as the name of a file that has the input data. Otherwise the program reads its input from the console. If the command line has a second parameter (optional), it is taken as the start vertex of the tour. An optional third parameter can be used to suppress the output, or to print verbose output. Suppression of output is convenient when the graph is huge, and printing the output consumes a lot of time, skewing the actual running time of the program to find an Euler tour.

**Input/Output**

The input is given in the format expected by Graph.readDirectedGraph(). First two integers read are |V| and |E|. The next |E| lines of input have 3 integers each: u, v, and w, indicating an edge (u,v) with weight w. In this problem, the weights of edges are irrelevant. If the graph is not Eulerian, the method isEulerian() in Euler.java will output the message: Graph is not Eulerian, and provide a justification for that claim with something like "Graph is not strongly connected" or "inDegree = 5, outDegree = 3 at Vertex 37". Otherwise, the graph is Eulerian, and the program should output a tour, by printing a sequence of edges, starting at the chosen start vertex, visiting every edge of G exactly once.

**Euler.java**

This is the main class that you need to complete. Some skeleton code is provided. You can add additional methods as needed. Do not change the signatures of any of the methods called by the driver LP2.java. Do not remove printTours() from the body of findEulerTour() method.

Sample input:

9 13

1 2 1

2 3 1

3 1 1

3 4 1

4 5 1

5 6 1

6 3 1

4 7 1

7 8 1

8 4 1

5 7 1

7 9 1

9 5 1

Its output (not unique):

(1,2)(2,3)(3,4)(4,7)(7,8)(8,4)(4,5)(5,7)(7,9)(9,5)(5,6)(6,3)(3,1)