Lab Problem 1

COP3503 assigned July 16, 2021
Michael McAlpin, Instructor due July 25, 2021

1 Goal

Find the **Minimum Spanning Tree** *(MST)* in a graph using **Prim's** algorithm.

2 Problem

1. Input data will be in a text file that contains the graph as an *Adjacency List*. [1] The input file's name will be specified in the command line as the first parameter. The input file format is specified as follows:
   (a) The first record is a single integer with the maximum number of vertices. (*This is a one-relative number as the vertices are numbered from 0 to Maximum Number Of Vertices - 1.*)
   (b) The second record is a single integer with the number of edges in the input file.
   (c) Subsequent records contain the edge's specific data, as follows:
      i. The first element is an integer representing one vertex of the edge.
      ii. The second element is an integer representing the other vertex of the edge.
      iii. The last element is an floating point number representing the weight of the edge. [2]
      iv. Example edge record: **4 14 6.50** is an edge between vertices 4 and 14 with a weight of 6.5.
2. The expected output will be the *MST* for the given graph to include the *total weight of the MST*.
   (a) Individual input edges should be output to `STDERR`.[3]
   (b) The final weight of the **MST** should be output to `STDOUT`.

2.1 Design Approach

The design of the ***Prim MST*** should be based on the *pseudocode* shown in Lecture 14, as shown below.

[1] Note that one test case has one million $(1,000,000)$ vertices with $7,580,000$ edges.
[2] All edge weight calculations are made based on using a `double` floating point number for all explicit and calculated weights.
[3] This is to allow for output discrepancies that *may* occur due to edge queuing variety. Using a **MinQ** will produce a slight variation in output sequence compared to **MinHeap**.

```
MST-PRIM(G,w,r)
1.  for each u ∈ G.V
2.     u.key = ∞
3.     u.pi = NIL
4.  r.key = 0
5.  Q = G.V
6.  while Q ≠ ∅
7.     u = EXTRACT-MIN(Q)
8.     for each v ∈ G.Adj[u]
9.        if v ∈ Q and w(u,v) < v.key
10.          v.pi = u
11.          v.key = w(u,v)
```

Note that there is also a test input file named, `lec14Prim.txt` provided in the test data.

3 Submission

via WebCourses

1. The single Java source file, named `Lab01.java`.
2. IMPORTANT Make sure that your submission has your name in the comment block as shown below at the very front of the file. **Make sure that your team mate's name is also in that comment block.** The two team members **must** submit identical java files in order to receive full credit. Don't forget to include the *Academic Integrity Statement*, also shown below.

3.0.1 Code Requirements

- Header - the following comment block should be at the beginning of the source file.

```
/*============================================================================
 |   Assignment:  Lab 01 - Building a Prim's MST for an input graph
 |
 |       Author:  Your name(s) here
 |     Language:  Java
 |
 |   To Compile:  javac Lab01.java
 |
 |   To Execute:  java Lab01 filename
 |                     where filename is in the current directory and contains
 |                           a record containing the number of vertices,
 |                           a record containing the number of edges,
 |                           many records containing the following:
 |                              Source edge number (integer)
 |                              Destination edge number (integer)
 |                              Edge weight (double)
 |
 |         Class:  COP3503 - CS II Summer 2021
 |   Instructor:  McAlpin
```

```
|     Due Date:  per assignment
|
+=============================================================================*/
```

- The following *Academic Integrity Statement* should be at the end of the source file.

```
/*=============================================================================
|     I [your name] ([your NID]) affirm that this program is
| entirely my own work and that I have neither developed my code together with
| any another person, nor copied any code from any other person, nor permitted
| my code to be copied  or otherwise used by any other person, nor have I
| copied, modified, or otherwise used programs created by others. I acknowledge
| that any violation of the above terms will be treated as academic dishonesty.
+=============================================================================*/
```

- Note that it is **imperative** that if you are submitting a two persons team assignment, that both individuals' names are included in **both** comment blocks shown above.
- Both the **header** and **Academic Integrity Statement** are included as a simple text file in the ZIP. They are named **header.txt** and **integrityStatement.txt**.

4 Testing

There are four input files supplied with the assignment:

1. **lec14Prim.txt** which contains 5 vertices and 6 edges.
2. **in8v16e.txt** which contains 8 vertices with 16 edges.
3. **in250v1273e.txt** which contains 250 vertices with 1273 edges.
4. **in1Mv758Ke.txt** which contains $1,000,000$ vertices with $7,580,000$ edges.

The test script, `testall.sh` is included in the ZIP file. [4] It compares student outputs to the expected outputs in a correspondingly named `MST` file. It **does not** compare the edges of the **MST**. *Note: Use this as guidance for the total weight of the MST. Some variations in edge output are allowed, as there may be subtleties in the **MinQ** functions which might change output order.*

[4] To execute the `testall.sh` script enter the following command: `bash testall.sh`.

5 Sample output

Sample outputs are included in the assignment ZIP file[5]. They are named to correspond to the input file's name. The sample below is derived from Lecture 14's Prim Example problem. *Remember the output of the **MST** edge data represents the edges in the calculated **MST**.*

```
~/labs/L1/code/tst $ java Lab01 lec14Prim.txt
0-2 0.20000
0-1 0.30000
1-4 0.20000
2-3 0.40000
1.10000
~/labs/L1/code/tst $
```

Note that the output is displayed to **STDOUT** and in the case shown above includes the **STDERR** output. The example below redirects **STDERR** to the **NULL** file. (That is, the **MST** edges output will not be shown.)

```
~/labs/L1/code/tst $ java Lab01 lec14Prim.txt 2>/dev/null
1.10000
~/labs/L1/code/tst $
```

6 Grading

Grading will be based on the following rubric:

Table 1: Grading Rubric

Percentage	Description
-100	Cannot compile on *Eustis*
-100	Cannot read input files as specified in the command line.
-100	Does NOT specify team member name or *no explicit statement that this is a solo submission.*
- 50	Does not output the vertices and edges for the **MST** to **STDERR**.
- 50	Cannot successfully calculate **MST** for given input file.

[5]The **MST** edge outputs for the input files are NOT included