COP3530 Project 3 – Linked lists

Due Date: Friday, 3/13/2020 11:59 PM

Turn in:

Submit the zipped Eclipse program including at least Project3.java, Stack.java, Queue.java, Country.java and Countries3.csv. The zip file should be named <your last name>_Project3.zip (for example, Liu_Project3.zip). The program should be well documented in the format of doc comments in Java. Detailed formats are found at http://www.oracle.com/technetwork/articles/java/index-137868.html.

Requirements:

1. Reuse your **Country** class from Project 1 and 2 (make changes if necessary)
2. Create a class named **Stack** that will implement a stack of country objects using a **double-ended singly linked list**. Support the following methods.
    a. Constructor that creates the stack.
    b. A **push** method to push a country on the stack. **(This should be an O(1) method.)**
    c. A **pop** method to pop a country off the stack and return it. **(This should be an O(1) method.)**
    d. A **printStack** method to print the stack from top of the stack to bottom.
    e. An **isEmpty** method that returns true if the stack is empty, false otherwise
    f. An **isFull** method that always returns false
3. Create a class named **Queue** that implements a double-ended queue of country objects using a **double-ended doubly linked list**. Support the following methods:
    a. Constructor that creates the queue.
    b. An **insertEnd** method to insert a country at the end of the queue.
    c. An **insertFront** method to insert a country at the front of the queue.
    d. A **removeEnd** method to remove a country from the end of the queue and return it.
    e. A **removeFront** method to remove a country from the front of the queue and return it.
    f. An **intervalDelete** method to delete from the queue country objects with GDP per capita that belong to the input interval. So this method takes two input parameters representing this interval. Return true if anything was found and deleted, and false otherwise.
    g. A **printQueue** method to print the queue in order front to end
    h. An **isEmpty** method that returns true if the queue is empty, false otherwise
    i. An **isFull** method that returns false always
4. Create a class named **Project3** that will:
    a. Like in project 2, let us again define five groups of countries based on GDP per capita: *POOR* countries as those with GDP per capita <1 thousand, *FAIR* no less than 1 thousand but <5 thousand, *GOOD* no less than 5 thousand but <20 thousand, *VGOOD* no less than 20 thousand but <50 thousand, and *EXCELLENT* no less than 50 thousand.

b. Read a file (csv) of countries and create a single stack of country objects containing countries from regions **POOR**, **GOOD**, and **EXCELLENT** (discard any countries not in those groups, do not modify the input file.).

c. Print the stack starting with the top of the stack.

d. Create a queue. Pop items from the stack, one at a time, and insert them to the queue: insert the first popped country in the front, the second popped country at the end, the third popped country at the front, the fourth popped country at the end, and so forth.

e. Print the queue from front to end.

f. Delete by calling the intervalDelete method countries with GDP per capita between 50,000 and 70,000 in the queue and print the queue again.

g. Remove items from the queue, one at a time, and push them on the stack. Remove one country from the front, then one from the back, then the front, then the back, and so forth.

h. Print the stack, then exit the program.

The queues and the stack should be printed in the following format:

```
Stack Contents:  or  Queue Contents:

Name          Code            Capitol      Population      GDP            HappinessRank
------------------------------------------------------------------------------------------
Chad          TCD             N'Djamena    15477751       11302544205     137
Vietnam       VNM             Hanoi        95540395       2.45E+11        94
```

Provide comments in this form for the **Country, Stack,** and **Queue** classes:

Comments for the class:

```
/**
 * Detailed description of the class.
 *
 * @author <your name>
 * @version <date you last changed the class>
 */
```

Public method comments:

```
/**
 * Description of the purpose of the method, the meaning of the
 * input parameters (if any) and the meaning of the return values * (if any).
 *
 * @param  parameter  description of the parameter (one for each)
 * @return description of the return value
 */
```

Provide comments in this form for the Project3 class.

```
/**
 * COP 3530: Project 3 – Linked lists
 * <p>
 * Description of the class using as many lines as needed
 * with <p> between paragraphs. Including descriptions of the
 * input required and output generated.
 *
 * @author <your name>
 * @version <the date you last modified the program>
 */
public class Project3
{
```

Example:

```
COP3530 Project 3 – Xudong Liu

Linked lists
Enter the file name: Countries3.csv


Stack Contents:

Name          Code        Capitol      Population      GDP            HappinessRank
-------------------------------------------------------------------------------------------------------
Hungary ...
Belize ...
. . .

Queue Contents:

Name          Code        Capitol      Population      GDP            HappinessRank
-------------------------------------------------------------------------------------------------------
Costa Rica ...
Tajikistan ...
. . .

Queue Contents:

Name          Code        Capitol      Population      GDP            HappinessRank
-------------------------------------------------------------------------------------------------------
Costa Rica ...
Tajikistan ...
. . .

Stack Contents:

Name          Code        Capitol      Population      GDP            HappinessRank
-------------------------------------------------------------------------------------------------------
Madagascar ...
Hungary ...
. . .
```