Assignment: Priority Queue

Due: Monday, April 15 at 11:59pm.

This assignment would allow students to understand and use Priority Queue algorithms in the program. In addition, this assignment is a great opportunity to review reading text files, handling exceptions, and callback objects to solve a specific problem.

The project data source is a text file that contains a set of records describing country specific information. Each record consists of the number of the country, name of the country, its population, and its area. Your task is to write a program that reads from such a file and prints.

(a) The country with the largest area.
(b) The country with the largest population.

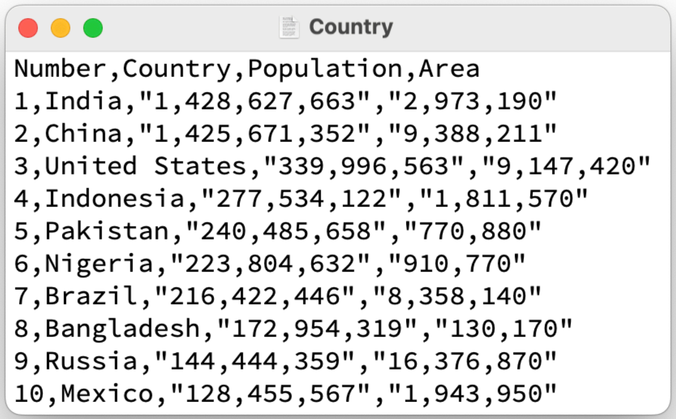
Deliverable 1: Create a template based Priority Queue data structure. Add the method insert, deleteMin, and isEmpty functions. The underlying data structure is a heap (array based binary tree). In the insert function, create code so that when the array is full it reallocates another level.

Deliverable 2: The insert function takes two parameters, the object to be inserted (of T type) and a lambda expression (as a callback object) that the percolateUp algorithm will use to compare two objects (of T type). Similarly, the deleteMin function takes one parameter, which is a lambda expression (as a callback object) that the percolateDown algorithm will use to compare two objects (of T type).

Deliverable 3: Create a Country class with name, number, area, population instance variables. Create a constructor that takes four parameters. Further, create getters and setters for each instance variables. In addition, implement the toString function.

Deliverable 4: Use appropriate object to read the file one line at a time. As shown in the following, in each row the values are separated by a comma. Therefore, use appropriate delimiter so that you can read these values easily. After reading a row from the file, use the four values to create an object of the Country class. Add the object in the Priority Queue (the Priority Queue will store Country Objects). Please note, since the Priority Queue is template based, it should be able to store Objects of any Class.

Deliverable 5: When reading the values from the file, you must convert the area and the population values into integers. While reading the file and while converting these values into integers, use exception handling. Basically, use exception handling throughout your project. Use try with resources whenever appropriate.



```
Number,Country,Population,Area
1,India,"1,428,627,663","2,973,190"
2,China,"1,425,671,352","9,388,211"
3,United States,"339,996,563","9,147,420"
4,Indonesia,"277,534,122","1,811,570"
5,Pakistan,"240,485,658","770,880"
6,Nigeria,"223,804,632","910,770"
7,Brazil,"216,422,446","8,358,140"
8,Bangladesh,"172,954,319","130,170"
9,Russia,"144,444,359","16,376,870"
10,Mexico,"128,455,567","1,943,950"
```

Deliverable 6:
(a) Create lambda expressions by implementing the Comparator interface to compare two Country class objects based on the population. Use the lambda expression to insert the Country objects in the Priority Queue variable. Further, in a loop, use the same lambda expression to obtain the Country object one a time by calling the deleteMin function. The loop will end when there are no more elements in the Priority Queue variable. Inside the loop, display the obtained Country object. Therefore, the output of the loop will display the Country objects in descending order based on the pupation of the country. Take a screenshot (screenshot A)

(b) Create lambda expressions by implementing the Comparator interface to compare two Country class objects based on the area of the Country. Use the lambda expression to insert the Country objects in the Priority Queue variable. Further, in a loop, use the same lambda expression to obtain the Country object one a time by calling the deleteMin function. The loop will end when there are no more elements in the Priority Queue variable. Inside the loop, display the obtained Country object. Therefore, the output of the loop will display the Country objects in descending order based on the area of the country. Take a screenshot (screenshot B)

Prepare and submit all the program files. Add the screenshot A, and B, and C in the pdf file. When you are satisfied with your programs submit it on the BrightSpace website.

Please note, in your programs implement the Priority Queue algorithm based on the content of the lectures. Do not use algorithms that we have not taught in the class.

Criterion	Details	Point Calculations
Deliverable 1 & 2:	Priority Queue data structure. The underlying data structure is a heap (array based binary tree).	Template based Priority Queue Data Structure that implements the 10: Insert function (including reAllocate function) 10: deleteMin function 10: The insert function takes two parameters, deleteMin function takes one parameter (lambda expression to compare the objects)
Deliverable 3, 4 & 5:	Read text file, convert values, handle exception, create objects, add the objects to the Priority Queue variable. Use try with resources whenever appropriate.	5: Creates Country class with needed methods 5: Text file data was read properly in an indefinite loop. The numeric values were converted to integers 10: Country class object was created by using the read values and added to the Priority Queue variable. 5: Exceptions were handled throughout the program. Try with resources was used for appropriate objects
Deliverable 6:	For (a), (b) Insert method and delete methods were called by using Lambda expressions. Further the Country objects were displayed by using an indefinite loop in descending order	10: Correct lambda expressions were passed to the insert, and deleteMin methods as a parameter for (a), and (b) 10: In a loop, appropriate callback objects (lambda expression) were passed as a parameter to the deleteMin method to display the Country objects in descending order of population (a) and area (b).
Code quality	Identifier names, class names, proper use of public/private, ample comments in main, etc.	-15: incoherent, inconsistent coding style -10: JavaDoc comments were not used throughout the code
Test	Note, whether the programs compile and run	-74: The portion of the code of the assignment is a copy -75: Uses Priority Queue algorithms that have not been taught in the class -20 x: The program does not compile -10: Two screenshots of the program run were not attached