Keele University

School of Computing and Mathematics

Module - CSC-20037

Coursework Assignment 2017-2018 (DRAFT)

Design and code a Java Application that :

(a) Implements a Linked-List based Queue and Stack (25 Marks)

(b) Allows the user to select between working with the Stack or the Queue (10 Marks)

(c) Allows values to be loaded from a text file and inserted into the

selected data structure (15 Marks)

(d) Allows additional values to be inserted and removed through the GUI (10 Marks)

(e) Demonstrates how the stack can be used to reverse a list of values (05 Marks)

(f) Depicts the selected structure graphically (updated after each change) (25 Marks)

(g) Displays appropriate messages in an area of the GUI (10 Marks)

You may 'dismantle and reassemble' the drawing program constructed during the module practicals as a starting point for your application. More detailed requirements are presented below:

(a) The stack and queue should be implemented as a linked data structure in accordance with the ADTs discussed in lectures.

(b) A toggle button of some form should permit the user to work with either the Stack or the Queue.

(c) The program should automatically attempt to load a file 'stack.txt' into the stack data structure from the application directory when it is run. The file is a text file containing a string representation of an integer on each line. For marking purposes, a test file with 15 values will be used. It should also be possible to load a file from the GUI (emptying any existing structure first). Failure to find or load the file should be handled within the program with feedback for the user displayed in a message area.

(d) Values entered and removed via the GUI should result in an update of the graphical display (f) when the <ENTER> key is pressed (or a button pressed). The change to the structure should be highlighted in some manner.

(e) A button should trigger the emptying of a selected stack item by item with the values appended to a string and displayed within the message area. The button should be disabled if a queue is selected.

(f) The currently selected data structure should be displayed graphically. This does not need to be elaborate. The display should be updated on any change to the contents of the structure.

(g) The message area can be employed to indicate error states and other appropriate messages to the user.

Note that a Graphics object has a setFont() method to select a font and a drawString() to display text. The getFontMetrics() method returns a FontMetrics instance that can be used to calculate the dimensions of a string to be drawn to the display.

Within each marking category, 50% of marks will be available for functionality, 30% for the quality of coding and 20% for coding presentation and comments. Within each of these sub-marking categories marks will be awarded in accordance with the university's generic undergraduate marking scheme. Submission details will be published to the KLE in the near future. All of the coding should be your own work save for use of the Java AWT, SWING and IO libraries, and the application shell produced during the practicals.

If you fail to implement any of the GUI elements, you should make the functionality of your data structures clear in some other manner.

David Collins

October, 2017