Assignment-XI (Structure and Queue)

Submit your program in the Moodle System within the deadline specified. Provide the result in a separate output file (named, result_<assgn><no>.txt). Use standard output redirection feature to generate the output file.

Define a structure for storing a student-record with the following information about a student:

(i) Name of a student: A string.

(ii) Roll-number: A string.

(iii) cgpa: A real number between 0 to 10.

Implement a stack of student-records using a linked list by writing the following functions.

(i) InitializeStack(): It initializes a stack.

(ii) Push(.): Push a student's record in the stack.

(iii) Pop(.): Pop a record from the stack. It returns NULL if the stack is empty.

(iv) isEmptyStack(.): Checks whether a stack is empty. It returns 1 if it is empty, else returns 0.

Use two stacks of student-records to implement a queue of the student record with the following function of a queue.

(i) InitializeQueue(): It initializes a Queue.

(ii) AddQueue(.): Adds a student's record in the Queue.

(iii) DeleteQueue(.): Deletes a student's record from Queue. It returns NULL if the stack is empty.

(iv) isEmptyQueue(.): Checks whether the Queue is empty. It returns 1 if it is empty, else returns 0.

(v) DisplayQueue(.): Displays the present content of the Queue in order of their retrieval, i.e. most recently added element would be displayed last in the

sequence. It prints a message “Empty Queue”, if the queue is empty. The content of the queue remains the same after this operation. The function uses only operations of queue for accessing data stored in it.

(vi) FindMaxQueue(.): It returns an array of student records with maximum CGPA from a Queue. After the operation the queue remains the same. The function uses only operations of queue for accessing data stored in it.

Write a main program which interactively performs the following operations in a Queue in a loop.

(i) Add a student’s record.
(ii) Delete a student’s record.
(iii) Display Queue.
(iv) Print records of toppers.

Execute your program by performing following operations in a sequence. A student-record is described here within curly braces ({.}), each field separated by a comma.

(i) Add {Antony Mark, 17CS30012, 9.3)
(ii) Add {Muhammad Ali, 32GG210013, 8.6}
(iii) Add {Manohar Ghosh, 21ED20024, 7.5}
(iv) Add {Bandana Bharathi, 16EX120012, 5.6}
(v) Delete
(vi) Add {Sashi Bhatia, 14CH100009, 8.7)
(vii) Add {Bhaichung Lama, 12AE300012, 9.3}
(viii) Add {Rama Parui, 18EE100032, 8.6}
(ix) Add { K. S. Khandelwal, 20CY100023, 9.3}
(x) Add {Manohar Sapui, 17CS100023, 9.3}
(xi) Delete
(xii) Delete
(xiii) Delete

(xiv) Add {Samuel Barton, 18EE300019, 8.7}
(xv) Add {Geeta Maiti, 14CH1000016, 9.3}
(xvi) Add {Mohana Garg, 19EC100008, 8.2}
(xvii) Add {Sahil Gupta, 26NA100025, 7.8}
(xviii) Delete
(xix) Delete
(xx) Delete
(xxi) Add {Israfil Mandal, 26NA3000003, 8.4}
(xxii) Display Queue
(xxiii) Print record of toppers
(xxiv) Add {Neha Aditya, 16EX1000029, 9.5}
(xxv) Delete
(xxvi) Delete
(xxvii) Display Queue
(xxviii) Print record of toppers