Algorithms and data structures

Labwork 3 - Stacks and Queues

November 7, 2023

After each labwork session:

- You will have one week (or 7 days) to complete the remaining exercises and upload your files to the "Labwork X"
- Compress all code source files in a zip file and rename it as FULLNAME-ID-Lab#no.zip (e.g NguyenVanA-BI10-070-Lab1.zip). Save your files according to the exercise number i.e Ex1.cpp, Ex2.c, etc. Incorrect filenames will result in no score for the respective exercises.
- Only code source files (.c or .cpp) should be in the zip files. Other files (.exe, .o) MUST be removed from the zip file.
- - Copy/Paste from any source is not tolerated. Penalty will be applied for late submissions.

NOTE: You must follow the guide. Incorrect zip file names, zip files containing other files (.exe), and copy/paste lead to heavy penalties.

Exercise 1

Suppose we would like to implement a queue system for a commercial website. Given that the website offers a list of items, each item has a limited quantity in stock. Implement a queue of customers, and each customer can buy only a few products among the available items.

- Determine the item name, quantity, and price in stock.
- Specify a queue of n customers; each customer can buy k products from one item (k is different for each customer).
- Customers take turns to enter (enqueue) and leave (dequeue) the queue according to the FIFO order to purchase wanted products.
- If a customer successfully purchases products, display a message and reduce the number of products in stock. Otherwise, display a warning message if the item has been run out.

Implement the above problems in C/C++ using a Queue data structure. Write a main function for testing all written functions (init(), display(), enqueue(), dequeue(),...).

Exercise 2

In this problem, we try to implement the navigation system in a web browser to back/forward a visiting website using a stack for Backward and another for Forward. Choose one method to implement a Stack data structure (Array-based or Stack using Linked List) to implement this problem.

- A website may contain a url and a title; suppose that we have visited a list of websites.
- The order of visited websites is stored in a Backward stack according to FILO.
- If we return to the previous website, we pop the Backward stack and push it into the Forward stack.
- And otherwise, when we forward to the next website, we retrieve the top website from the Forward stack and push it into the Backward stack.

Write a main function for testing all written functions (init(), display(), push(), pop(),...).