Informatics II, Spring 2023, Exercise 7

Publication of exercise: April 17, 2023
Publication of solution: April 23, 2023
Exercise classes: April 24 - 28, 2023

Task 1.

1. Given an empty stack S, illustrate the result of each operation in the sequence push(4), push(1), push(3), pop(), push(8), and pop(S).
2. Given an empty queue Q, illustrate the result of each operation in the sequence enqueue(4), enqueue(1), enqueue(3), dequeue(), enqueue(8), and dequeue().
3. Explain how to implement two stacks in one array $A[]$ in such a way that neither stack overflows unless the total number of elements in both stacks together is n. The push and pop operations should run in $O(1)$ time.
4. Explain how to implement a queue Q' using two stacks. Analyze the running time of the enqueue and dequeue operations on Q'.
5. Explain how to implement a stack S' using two queues. Analyze the running time of the pop and push operations on S'.

Task 2. Implementation of stacks and queues in C

1. Write a C program that implements a stack using an array. Your C program should contain push and pop functions, and examples to call implemented functions.
2. Write a C program that implements a queue using an array. Your C program should contain enqueue and dequeue functions, and examples to call implemented functions.
3. Write a C program that implements a stack using a singly linked list. The push and pop operations should still take $O(1)$ time.
4. Write a C program that implements a queue using a singly linked list. The enqueue and dequeue operations should still take $O(1)$ time.

Task 3. [2022 Final Exam] Provide pseudocode for a function `reverseEven` that takes a queue Q of integers as a parameter, and modifies Q by reversing the order of the even integers in the queue, while keeping the odd integers in place.

For example given the queue: Q = [14, 17, 16, 18, 21, 7, 28, 40]. The function `reverseEven` modifies the queue to: Q = [40, 17, 28, 18, 21, 7, 16, 14]

You are only allowed to use the following abstract data types and functions:
Queue:

`Q = initQueue()`: initializes a Queue
`enQueue(Q,x)`: inserts value x to queue Q (in the end)
`deQueue(Q)`: removes value from queue Q (in the beginning)
`queueSize(Q)`: returns the number of elements in queue Q

Stack:
`S = initStack()`: initializes a stack
`push(S,x)`: pushes value x onto stack S
`pop(S)`: removes value from stack S
`stackSize(S)`: returns the number of elements in stack S