

NANYANG TECHNOLOGICAL UNIVERSITY
School of Electrical & Electronic Engineering

IE2108 Data Structures and Algorithms

Tutorial No. 4 (Sem 1, AY2022-2023)

1. Suppose that S is a stack. List the content of the stack after each operation and show the output value if a value is returned from the operation.

Operation	Output	Bottom – Stack – Top
$S.\text{stack_init}()$		
$S.\text{empty}()$		
$S.\text{push}(8)$		
$S.\text{push}(-5)$		
$S.\text{pop}()$		
$S.\text{push}(2)$		
$S.\text{top}()$		
$S.\text{pop}()$		
$S.\text{empty}()$		
$S.\text{top}()$		

2. Suppose that Q is a queue. List the content of the queue after each operation and show the output value if a value is returned from the operation.

Operation	Output	Front – Queue – Rear
$Q.\text{queue_init}()$		
$Q.\text{empty}()$		
$Q.\text{enqueue}(8)$		
$Q.\text{enqueue}(-5)$		
$Q.\text{dequeue}()$		
$Q.\text{enqueue}(2)$		
$Q.\text{front}()$		
$Q.\text{dequeue}()$		
$Q.\text{empty}()$		
$Q.\text{front}()$		

3. Using the abstract data type *stack*, write a function $\text{invert}(S)$ to invert the contents of a stack S . You may use additional stacks in your function.
Note: If the number 3 is at the top of the stack S , after $\text{invert}(S)$, 3 will be at the bottom of S .
4. Suppose that start is a reference to the first node of a singly-linked list. Write an algorithm that begins at start and insert a value val . Specifically, the algorithm adds a node to the end of the linked list whose data field is val . Discuss the worst-case time complexity of your algorithm.

5. A pointer *start* points to the first element of a doubly-linked list *L*. Write an algorithm that deletes the smallest element in *L*.
6. Using the operations *front()*, *enqueue(val)* and *dequeue()*, write the pseudo-code of a **recursive algorithm** to append a queue *P* (which may be empty) onto the end of another queue *Q*, leaving *P* empty.
7. The pointer *start* points to the first element of a singly-linked list *L*. Write a **recursive algorithm** to return a reference to the first element that has a value that is greater than the next element in *L*. If no such element exists, return null. For example, if the linked list is arranged as in Fig. 1, then the algorithm will return a reference to the third element (which has the value 7).

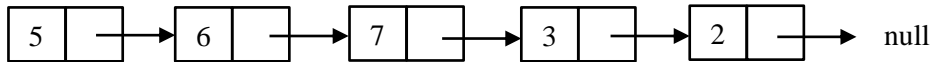


Fig. 1