

CSE 214 – Recitation 4: Queues [Fall 2022] - Student Version

1. [5 minutes] Consider the following statements:

- i. The stack data structure follows the FIFO (First In First Out) principle.
- ii. The enqueue operation inserts an element onto the front of the queue.
- iii. The dequeue operation removes an element from the rear of the queue.
- iv. The queue data structure follows the LIFO (Last In First Out) principle.
- v. One implementation of the queue data structure is to check whether an arithmetic expression has balanced parenthesis.

Which of the following is correct?

- A. (i) and (iv) are true
- B. (ii) and (iii) are true
- C. (i), (ii), (iii) and (iv) are true
- D. (v) is true
- E. (i), (ii), (iii), (iv) and (v) are true
- F. None of above

2. [2 minutes] A normal queue, implemented using a circular array, gets full when?

- A. $\text{front} == -1 \ \&\& \ \text{rear} == -1$
- B. $\text{rear} == \text{CAPACITY}$
- C. $\text{rear} == \text{CAPACITY} - 1$
- D. $\text{rear} == \text{front}$
- E. $(\text{rear} + 1) \% \text{CAPACITY} == \text{front}$
- F. None of above

3. [5 minutes] Consider a priority queue implemented using a sorted array. What is the worst case time complexity for the following operations?

- 1) Enqueue
- 2) Peek
- 3) Dequeue

4. [5 minutes] What is the worst-case complexity of the following:

| | Enqueue | Dequeue |
|--|---------|---------|
| Array with pointer to next available slot | | |
| Array without pointer to next available slot | | |

| | | |
|--|--|--|
| Circular Array | | |
| Singly linked list with head reference and head as the front | | |
| Singly linked list with head and tail references with the head as the front | | |
| Singly linked list with head and tail references with the tail as the front | | |
| Doubly linked list with head and tail references with the head or tail as the front | | |

5. [15 minutes] Write a method that removes and return the nth element in an IntQueue. Assume you have access to the basic queue methods, such as enqueue, dequeue, size, etc.

```
public int remove (IntQueue q, int n) {
    // Fill in code here
```

```
}
```

6. [5 minutes] Given a Queue implemented using a circular array with a capacity of 5 and a sequence of operations:

front = -1

rear = -1



- 1) enqueue(3)
- 2) enqueue(4)
- 3) enqueue(6)
- 4) enqueue(8)
- 5) dequeue()
- 6) dequeue()
- 7) enqueue(2)
- 8) enqueue(10)
- 9) enqueue(6)

What does the Queue look like after the above 7 operations?

What would happen if we try an extra operation “enqueue(9)” ?

7. [10 minutes] Write the following method to reverse a queue using recursion. Assume you have access to the following operations

- 1) enqueue(x) : Add an item x to rear of queue.
- 2) dequeue() : Remove an item from front of queue.
- 3) isEmpty() : Checks if a queue is empty or not.

```
IntQueue reverseQueue(IntQueue q) {  
    // fill in code here
```

```
}
```