## (5') Stack, Queue and Complexity Analysis 1

Your ID:

Each question has one or more correct answer(s). Select all the correct answer(s). For each question, you get 0 point if you select one or more wrong answers, but you get 0.5 point if you select a non-empty subset of the correct answers.

Note that you should write you answers of section 1 in the table below.

Question 1	Question 2	Question 3	Question 4	Question 5
D	A D	ABCD	A	ABC

Question 1. In the lectures of Week 2, suppose we implement a circular queue by using an array with the index range from 0 to (n-1), then what the size of this queue would be? We assume that the queue is non-empty.

- (A) rear front + 1
- (B) (rear front + 1)%n
- (C) (rear front + n)%n
- (D) (rear front + n)%n + 1

**Question 2.** Which of the following is known to be correct?

- (A) Stack is a linear data structure and the operations on stacks are more restricted, the same is true for queue.
- (B) Lists store elements in sequential locations in memory.
- (C) Both stacks and queues allow us insert or delete an element at the front.
- (D) We can use two queues to implement stack.

Question 3. Which of the following is/are applications of queue and stack respectively?

- (A) Queue: A resource shared by multiple users/processes; Stack: Handling function calls
- (B) Queue: Loading Balancing; Stack: Reverse-Polish Notation
- (C) Queue: Handling of interrupts in real-time systems; Stack: Compilers/Word Processors
- (D) Queue: IO Buffers; Stack: Arithmetic expression evaluation

Question 4. Read the following code, what function does it realize? void Q4(Queue &Q) Stack S;int d;InitStack(S); while (!QueueEmpty(Q)){

```
DeQueue(Q, d)
         Push(S,d);
      while(!StackEmpty(S))
         Pop(S,d);
        EnQueue(Q, d);
}
```

Your ID:

- (A) Use stack to reverse the queue.
- (B) Use queue to reverse the stack.
- (C) Use stack to implement the queue.
- (D) Use queue to implement the stack.

**Question 5.** Which of the following comparison is correct?

- (A)  $n^2 + n^3 = O(n^4)$
- (B)  $\log_2 n = \Theta(\log n)$
- (C)  $\log^2 n = \Omega(\log \log n)$
- (D)  $n! = \omega(n^n)$

## 2 (10') Stack and Queue

Question 6. (2') The following post-fix expression (Reverse-Polish Notation) with single digit operands is evaluated using a stack and the final result:

Note that  $\hat{\ }$  is the exponentiation operator. Please write down the corresponding in-fix notation:  $\frac{8}{2^3} + 2 * 3 - 5 * 1$ 

Question 7. (4')Describe how to implement a queue using a singly-linked list.

Check whether Queue is empty.

ENQUEUE: inserts an element at the end of the list. In this case we need to keep track of the last element of the list. We can do that with a sentinel.

DEQUEUE: removes an element from the beginning of the list.

Your Name:

Question 8. (1') If we use an array with size N to implement a normal queue, it gets full when rear=N-1

HW3

Question 9. (1') By implementing the following operations on stack, the value of x is aInitStack(st); Push(st,a); Push(st,b); Pop(st,x); Top(st,x);

Question 10. (2') When will "stack overflow" and "stack underflow" happen? (Give a short explanation.)

Overflow: Adding items to a full stack.

Underflow: Removing items from an empty stack.

## 3 (8') Complexity Analysis

Question 11. (3') Given a fraction of a code as the following, write down the time complexity for each for loop.

```
for (i=1; i< n; i*=2) {
  for(j=n; j>0; j/=2) {
          for (k=j; k< n; k+=2) {
                sum \neq (i + j*k)
```

 $O(\log n)$  $O(\log n)$ \_\_\_\_O(n)\_\_\_\_\_

Question 12. (5') Calculate the average processing time T(n) of the following recursive algorithm. Suppose that it takes one unit time for random (int n) to return a random integer which is uniformly distributed in the range [0,n]. Also note that T(0) = 0. *Hints: The equation*  $\frac{1}{1*2} + \frac{1}{2*3} + ... + \frac{1}{n*(n+1)} = \frac{n}{n+1}$  *might be needed.* 

```
int hw(int n) {
 if (n \le 0) return 0;
   else {
         int i = random(n-1);
         return hw(i) + hw(n-1-i);
}
```

The algorithm suggests that T(n) = T(i) + T(n-1-i) + 1. By summing this relationship for all the possible random values i = 0, 1, ..., n - 1, we obtain that in average nT(n) = 2(T(0) + T(1) + ... + T(n-2) + T(n-1)) + n. Because (n-1)T(n-1) = 2(T(0) + T(1) + ... + T(n-2)) + (n-1), the basic recurrence is as follows: nT(n) - (n-1)T(n-1) = 2T(n-1) + 1, or nT(n) = (n+1)T(n-1) + 1, or

 $\frac{T(n)}{n+1} = \frac{T(n-1)}{n} + \frac{1}{n*(n+1)}$ . The telescoping results in the following system of expressions:

$$\frac{T(n)}{n+1} = \frac{T(n-1)}{n} + \frac{1}{n*(n+1)}$$
.....
$$\frac{T(1)}{2} = \frac{T(0)}{1} + \frac{1}{1*2}$$

Since T(0) = 0, the explicit expression for T(n) is:

$$\frac{T(n)}{n+1} = \frac{1}{1*2} + \frac{1}{2*3} + \dots + \frac{1}{(n-1)*n} + \frac{1}{n*(n+1)}$$

So that T(n) = n