Homework of Data Structures

Chapter 3 Stack and Queue

Question 1

Following is C like pseudo code of a function that takes a number as an argument, and uses a stack S to do processing.

```
void fun(int n)
{
   Stack S;   // Say it creates an empty stack S
   while (n > 0)
   {
     // This line pushes the value of n%2 to stack S
     push(&S, n%2);

     n = n/2;
   }

   // Run while Stack S is not empty
   while (!isEmpty(&S))
     printf("%d ", pop(&S)); // pop an element from S and print it
}
```

What does the above function do in general?

A Prints binary representation of n in reverse order
B Prints binary representation of n
C Prints the value of Logn
D Prints the value of Logn in reverse order

Question 2

Which one of the following is an application of Stack Data Structure?

A Managing function calls
B The stock span problem
C Arithmetic expression evaluation
D All of the above

Question 3

Consider the following pseudocode that uses a stack

```
declare a stack of characters
while ( there are more characters in the word to read )
        //the work is "geeksquiz"
{
  read a character
  push the character on the stack
}
while ( the stack is not empty )
{
  pop a character off the stack
  write the character to the screen
}
```

A geeksquizgeeksquiz
B ziuqskeeg
C geeksquiz
D ziuqskeegziuqskeeg

Question 4

The following postfix expression with single digit operands is evaluated using a stack:

8 2 3 ^ / 2 3 * + 5 1 * -

Note that ^ is the exponentiation operator. The top two elements of the stack after the first * is evaluated are:

A 6, 1
B 5, 7
C 3, 2
D 1,5

Question 5

To evaluate an expression without any embedded function calls:

A One stack is enough
B Two stacks are needed
C As many stacks as the height of the expression tree are needed
D A Turing machine is needed in the general case

Question 6

The result evaluating the postfix expression 10 5 + 60 6 / * 8 – is

A 284
B 213
C 142
D 71

Question 7

What is the output of the program for the following input ?

5 2 * 3 3 2 + * +

A 15
B 25
C 30
D 150

Question 8

Suppose a stack is to be implemented with a linked list instead of an array. What would be the effect on the time complexity of the push and pop operations of the stack implemented using linked list (Assuming stack is implemented efficiently)?

A O(1) for insertion and O(n) for deletion
B O(1) for insertion and O(1) for deletion
C O(n) for insertion and O(1) for deletion
D O(n) for insertion and O(n) for deletion

Question 9

Consider the following statements:

i. First-in-first out types of computations are efficiently supported by STACKS.

ii. Implementing LISTS on linked lists is more efficient than implementing LISTS on an array for almost all the basic LIST operations.

iii. Implementing QUEUES on a circular array is more efficient than implementing QUEUES on a linear array with two indices.

iv. Last-in-first-out type of computations are efficiently supported by QUEUES.

Which of the following is correct?

A (ii) and (iii) are true
B (i) and (ii) are true
C (iii) and (iv) are true
D (ii) and (iv) are true

Question 10

Which of the following permutation can be obtained in the same order using a stack assuming that input is the sequence 5, 6, 7, 8, 9 in that order?

A 7, 8, 9, 5, 6
B 5, 9, 6, 7, 8
C 7, 8, 9, 6, 5
D 9, 8, 7, 5, 6

Question 11

The best data structure to check whether an arithmetic expression has balanced parenthesis is a

A Queue
B Stack
C Tree
D List

Question 12

The seven elements A, B, C, D, E, F and G are pushed onto a stack in reverse order, i.e., starting from G. The stack is popped five times and each element is inserted into a queue. Two elements are deleted from the queue and pushed back onto the stack. Now, one element is popped from the stack. The popped item is

A A
B B
C F
D G

Question 13

If the sequence of operations - push (1), push (2), pop, push (1), push (2), pop, pop, pop, push (2), pop are performed on a stack, the sequence of popped out values

A 2,2,1,1,2
B 2,2,1,2,2
C 2,1,2,2,1
D 2,1,2,2,2

Question 14

The five items: A, B, C, D, and E are pushed in a stack, one after other starting from A. The stack is popped four items and each element is inserted in a queue. The two elements are deleted from the queue and pushed back on the stack. Now one item is popped from the stack. The popped item is

A A
B B
C C
D D

Question 15

Consider the following operations performed on a stack of size 5 : Push (a); Pop() ; Push(b); Push(c); Pop(); Push(d); Pop();Pop(); Push (e) Which of the following statements is correct?

A Underflow occurs
B Stack operations are performed smoothly
C Overflow occurs
D None of the above

Question 16

Which of the following is not an inherent application of stack?

A Implementation of recursion
B Evaluation of a postfix expression
C Job scheduling
D Reverse a string

Question 17

Convert the following infix expression into its equivalent post fix expression

(A + B^ D) / (E – F) + G

A ABD^ + EF – / G+
B ABD + ^EF – / G+
C ABD + ^EF / – G+
D ABD^ + EF / – G+

Question 18

Following is C like pseudo code of a function that takes a Queue as an argument, and uses a stack S to do processing.

```
void fun(Queue *Q)
{
     Stack S;    // Say it creates an empty stack S

     // Run while Q is not empty
     while (!isEmpty(Q))
     {
          // deQueue an item from Q and push the dequeued item to S
          push(&S, deQueue(Q));
     }

     // Run while Stack S is not empty
     while (!isEmpty(&S))
     {
        // Pop an item from S and enqueue the poppped item to Q
        enQueue(Q, pop(&S));
     }
}
```

What does the above function do in general?

A Removes the last from Q
B Keeps the Q same as it was before the call
C Makes Q empty
D Reverses the Q

Question 19

Which of the following is true about linked list implementation of queue?

A In push operation, if new nodes are inserted at the beginning of linked list, then in pop operation, nodes must be removed from end.
B In push operation, if new nodes are inserted at the end, then in pop operation, nodes must be removed from the beginning.
C Both of the above
D None of the above

Question 20

Which of the following is true about linked list implementation of queue?

A. In push operation, if new nodes are inserted at the beginning of linked list, then in pop operation, nodes must be removed from end.

B. In push operation, if new nodes are inserted at the end, then in pop operation, nodes must be removed from the beginning.

C. Both of the above

D. None of the above

Question 21

Suppose a circular queue of capacity (n – 1) elements is implemented with an array of n elements. Assume that the insertion and deletion operation are carried out using REAR and FRONT as array index variables, respectively. Initially, REAR = FRONT = 0. The conditions to detect queue full and queue empty are

A. Full: (REAR+1) mod n == FRONT, empty: REAR == FRONT

B. Full: (REAR+1) mod n == FRONT, empty: (FRONT+1) mod n == REAR

C. Full: REAR == FRONT, empty: (REAR+1) mod n == FRONT

D. Full: (FRONT+1) mod n == REAR, empty: REAR == FRONT

Question 22

Consider the following pseudo code. Assume that IntQueue is an integer queue. What does the function fun do?

```
void fun(int n)
{
   IntQueue q = new IntQueue();
   q.enqueue(0);
   q.enqueue(1);
   for (int i = 0; i < n; i++)
   {
      int a = q.dequeue();
      int b = q.dequeue();
      q.enqueue(b);
      q.enqueue(a + b);
      ptint(a);
   }
}
```

A. Prints numbers from 0 to n-1

B. Prints numbers from n-1 to 0

C. Prints first n Fibonacci numbers

D. Prints first n Fibonacci numbers in reverse order.

Question 23

Suppose you are given an implementation of a queue of integers. The operations that can be performed on the queue are:
i. isEmpty (Q — returns true if the queue is empty, false otherwise.
ii. delete (— deletes the element at the front of the queue and returns its value.
iii. insert (Q, — inserts the integer i at the rear of the queue.
Consider the following function:

```
void f (queue Q) {
int i ;
if (!isEmpty(Q)) {
   i = delete(Q);
   f(Q);
   insert(Q, i);
  }
}
```

A Leaves the queue Q unchanged
B Reverses the order of the elements in the queue Q
C Deletes the element at the front of the queue Q and inserts it at the rear keeping the other elements in the same order
D Empties the queue Q

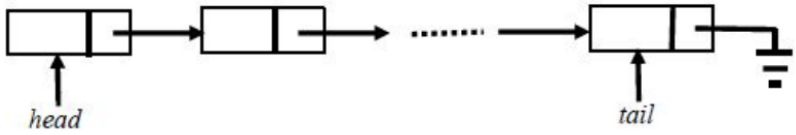
Question 24

Consider the following statements:
i. First-in-first out types of computations are efficiently supported by STACKS.
ii. Implementing LISTS on linked lists is more efficient than implementing LISTS on an array for almost all the basic LIST operations.
iii. Implementing QUEUES on a circular array is more efficient than implementing QUEUES on a linear array with two indices.
iv. Last-in-first-out type of computations are efficiently supported by QUEUES.
Which of the following is correct?

A ii) and (iii) are true
B (i) and (ii) are true
C (iii) and (iv) are true
D (ii) and (iv) are true

Question 25

A queue is implemented using a non-circular singly linked list. The queue has a head pointer and a tail pointer, as shown in the figure. Let n denote the number of nodes in the queue. Let 'enqueue' be implemented by inserting a new node at the head, and 'dequeue' be implemented by deletion of a node from the tail.



Which one of the following is the time complexity of the most time-efficient implementation of 'enqueue' and 'dequeue, respectively, for this data structure?

A $\Theta(1),\Theta(1)$
B $\Theta(1),\Theta(n)$
C $\Theta(n),\Theta(1)$
D $\Theta(n),\Theta(n)$

Question 26

Consider a standard Circular Queue 'q' implementation (which has the same condition for Queue Full and Queue Empty) whose size is 11 and the elements of the queue are q[0], q[1], q[2].....,q[10]. The front and rear pointers are initialized to point at q[2] . In which position will the ninth element be added?

A q[0]
B q[1]
C q[9]
D q[10]