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

```
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?

- Removes the last from Q
- Keeps the Q same as it was before the call
- Makes Q empty
- Reverses the Q

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

- In Insertoperation, if new nodes are inserted at the beginning of linked list, then in deleteoperation, nodes must be remov
- In insertoperation, if new nodes are inserted at the end, then in deleteoperation, nodes must be removed from the begin
- Both of the above
- None of the above

3) Suppose you are given an implementation of a queue of integers. The operations that can be performed on returns true if the queue is empty, false otherwise.ii. delete (Q) â?? deletes the element at the front of the qu insert (Q, i) â?? 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);
}
}
```

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

4) Consider a standard Circular Queue 'q' implementation (which has the same condition for Queue Full and Qu and the elements of the queue are q[0], q[1], q[2].....,q[10]. The front and rear pointers are initialized to point the ninth element be added?

- q[0]
- q[1]
- q[9]
- q[10]

5) Consider the following pseudocode that uses a stack

```
declare a stack of characters
while ( there are more characters in the word to read )
{
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
}
```

What is output for input â??geeksquizâ???

- Geeksquizgeeksquiz
- ziuqskeeg
- geeksquiz
- ziuqskeegziuqskeeg

6) A single array A[1..MAXSIZE] is used to implement two stacks. The two stacks grow from opposite ends of the (topl< top 2) point to the location of the topmost element in each of the stacks. If the space is to be used effi stack fullâ?? is

- (top1 = MAXSIZE/2) and (top2 = MAXSIZE/2+1
- top1 + top2 = MAXSIZE
- (top1= MAXSIZE/2) or (top2 = MAXSIZE
- top1= top2 -1

7) A priority queue can efficiently implemented using which of the following data structures? Assume that the r (operation to see the current highest priority item) and extraction (remove the highest priority item) operatic

- Array
- Linked List
- Heap Data Structures like Binary Heap, Fibonacci Heap
- None of the above

8) The result evaluating the postfix expression 10 5 + 60 6 / * 8 â?? is

- 284
- 213
- 142
- 71

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

```
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?

- Prints binary representation of n in reverse order
- Prints binary representation of n
- Prints the value of Logn

- Prints the value of Logn in reverse order

10) Following is an incorrect pseudocode for the algorithm which is supposed to determine whether a sequenc

declare a character stack

while (more input is available)

{

read a character

if (the character is a '(')

push it on the stack

else if (the character is a ')' and the stack is not empty)

pop a character off the stack

else

print "unbalanced" and exit

}

print "balanced"

Which of these unbalanced sequences does the above code think is balanced?

- (((()
- ())(()
- (()()))
- â?µ
  (())()

11) Which of the following option is not correct?

- If the queue is implemented with a linked list, keeping track of a front pointer, Only rear pointer s will change during an in
- Queue data structure can be used to implement least recently used (LRU) page fault algorithm and Quick short algorithm.
- Queue data structure can be used to implement Quick short algorithm but not least recently used (LRU) page fault algoritl
- Both 3 and 4

12) Consider the following statements

a. First-in-first out types of computations are efficiently supported by STACKS.
b. Implementing LISTS on linked lists is more efficient than implementing LISTS on an array for almo
c. Implementing QUEUES on a circular array is more efficient than implementing QUEUES on a linea
d. Last-in-first-out type of computations are efficiently supported by QUEUES.

Which of the following is correct?

- First-in-first out types of computations are efficiently supported by STACKS.
- Implementing LISTS on linked lists is more efficient than implementing LISTS on an array for almost all the basic LIST oper
- Implementing QUEUES on a circular array is more efficient than implementing QUEUES on a linear array with two indices.
- Last-in-first-out type of computations are efficiently supported by QUEUES.

13) Assume that the operators +, -, Ã? are left associative and ^ is right associative. The order of precedence (fr
The postfix expression corresponding to the infix expression a + b Ã? c - d ^ e ^ f is

- abc Ã? + def ^ ^ -

- abc Ã? + de ^ f ^ -
- ab + c Ã? d - e ^ f ^
- - + a Ã? bc ^ ^ def

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

- Managing function calls
- The stock span problem
- Arithmetic expression evaluation
- â?µ
  All of the above

15) Consider the following pseudo code. Assume that IntQueue is an integer queue. What does the function fu

```
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);
}
}
```

- Prints numbers from 0 to n-1
- Prints numbers from n-1 to 0
- Prints first n Fibonacci numbers
- Prints first n Fibonacci numbers in reverse order

16) Which of the following is true about linked list implementation of stack?

- In push operation, if new nodes are inserted at the beginning of linked list, then in pop operation, nodes must be remove
- In push operation, if new nodes are inserted at the end, then in pop operation, nodes must be removed from the beginni
- Both of the above
- None of the above

17) 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 a

- 6,1

- 5,7
- 3,2
- 1,5

18) Consider the following C program:

```
#include
#define EOF -1
void push (int); /* push the argument on the stack */
int pop (void); /* pop the top of the stack */
void flagError ();
int main ()
{ int c, m, n, r;
while ((c = getchar ()) != EOF)
{ if (isdigit (c)
push (c);
else if ((c == '+') || (c == '*'))
{ m = pop ();
n = pop ();
r = (c == '+') ? n + m : n*m;
push (r);
}
else if (c != ' ')
flagError ();
}
printf("% c", pop ());
```

What is the output of the program for the following input? 5 2 * 3 3 2 + * +

- 15
- 25
- 30
- 150

19) Suppose a circular queue of capacity (n â?? 1) elements is implemented with an array of n elements. Assum operation are carried out using REAR and FRONT as array index variables, respectively. Initially, REAR = FRO queue full and queue empty are

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