**Name:** _____ **Roll Number:** _____

**Ques.1** What is the result of the following operation? **(1 Mark)**

**Top (Push (stack, X))**

X

**Ques.2** Consider 2 sorted lists of size **'m'** and **'n'** respectively. Find the number of comparisons needed in the worst case by merge sort algorithm? **(1 Mark)**

$(m+n-1)$ or $(m+n)$

**Ques.3** The following postfix expression with single digit operands is evaluated using a stack **(1 Mark)**

4, 2, 2, *, /, 2, 3, ^, +, 5, -

The top 2 elements of the stack after the evaluation of ^ are 8 and 1.

**Ques.4** Suppose we are comparing implementations of insertion sort and merge sort on the same machine. For inputs of size 'n', insertion sort runs in $8n^2$ steps, while merge sort runs in 64nlogn steps. For which values of 'n' does insertion sort beat merge sort? **(2 Marks)**

$n <= 43$ by solving $8n^2 \leq 64n\log n$

**Ques.5 Choose the single correct option for the below multiple-choice questions:** **(1*3 = 3 Marks)**

**I.** Suppose we are sorting an array of eight integers using quicksort, and we have just finished the first partitioning with the array looking like this: 2 5 1 7 9 12 11 10

**Which statement is correct?**

**(a)** The pivot could be either the 7 or the 9 ✓
**(b)** The pivot is not the 7, but it could be the 9
**(c)** The pivot could be the 7, but it is not the 9
**(d)** Neither the 7 nor the 9 is the pivot.

**II.** Which algorithm is having highest space complexity?

**(a)** Bubble sort **(b)** Insertion Sort **(c)** Quick Sort **(d)** Merge Sort ✓

**III.** If the array is already sorted, then the running time for merge sort is: ?

**(a)** O(1) **(b)** O(n*log n) ✓ **(c)** O(n) **(d)** $O(n^2)$

**Ques.6 Indicate True/False:** **(3 Marks)**

1. Push is used to place elements on the bottom of a stack and Pop is used to remove elements from the top of a stack. False
2. Postfix expression is just a reverse of prefix expression. False
3. The most appropriate data structure to print a list of elements in reverse order is the Queue. False

**Ques.7** Translate the following Infix expression into Postfix Expression. Only write the final output here: **(1 Mark)**

**((a + b) + c * (d + e) + f )* (g + h)**

a b + c d e + * + f + g h + *

**Ques.8** A stack is to be implemented using an array. The associated declarations are: **(1 Mark)**

int stack [100];
int top = -1;

Give **one line** statement **(C Statement)** to perform push operation assuming that the stack is not full.

push( stack [++top]);

**Ques.9 What are the time complexities of each of the following tasks? (Choose from O(1), O(logn), O(n), O(n logn), $O(n^2)$, $O(2^n)$ )** **(4 Marks)**

(a) Popping an item off a stack containing 'n' items. O(1)

(b) Performing a Towers of Hanoi algorithm with 'n' disks. $O(2^n)$

(c) Using quicksort to sort an array of 'n' integers, in the worst case. $O(n^2)$

(d) Performing a bubble sort on an array of 'n' integers, in the worst case. $O(n^2)$

**Ques.10** Covert the following Infix Expression into Postfix Expression. Write only the final output here: **(1 Mark)**

A * (B + D) / E – F * (G +H / K)

A B D + * E / F G H K / + * –

**Ques.11** The array a contains the following 3 elements: **(2*2 = 4 Marks)**

| a = | a | b | c |
|---|---|---|---|

For each of the following two program fragments **indicate what they print** if functions and terms have their usual meaning:

**1.**
```
Stack s = create_stack(s);
for( i = 0; i < 3; i++)
{
    push(s, a[i]);
    for( j = 0; j < i; j++)
        push(s, a[j]);
 }
 while(!Is_Empty(s))
        printf("%c", pop(s));
```

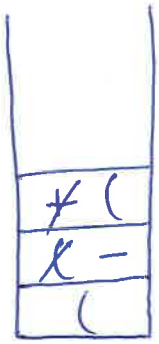
b, a, c, a, b, a

2. 
```
Queue q = create_queue(q);
for( i = 0; i < 3; i++)
{
    enqueue(q, a[i]);
    for( j = 0; j < i; j++)
        enqueue(q, a[j]);
 }
 while(!Is_Empty(q))
        printf("%c", dequeue(q));
```

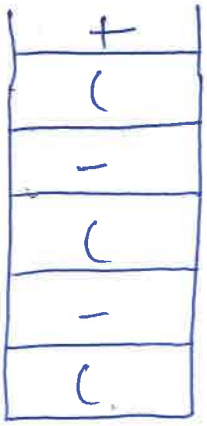
a, b, a, c, a, b

**Ques.12** Convert the following infix expression into a postfix expression using a stack. Draw 3 boxes (stacks) each containing 5 entries. Show the state of the stack at each of the indicated positions 1, 2 and 3 in the boxes. Also write down the final postfix expression. **(3 Marks)**

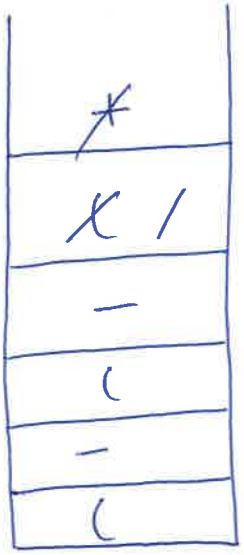
**( A + B ) – (  1  C – ( D + 2 E ) / F 3 ) * G**



(1)



(2)



(3)

**Ques. 4 Solution**

**Since you have to find when insertion sort beats merge sort**

$8n^2 <= 64nlogn$
$n^2 <= 8nlogn$
$n <= 8logn$

On solving $n-8logn = 0$, we get
$n = 43.411$

So, for **n <= 43** insertion sort works better than merge sort.