SET – 1B

Stacks and Queues

1. Compute the postfix equivalent of the following infix arithmetic expression where a+b*c+d*e↑f; where ↑ represents exponentiation. Assume normal operator precedence.

	Infix expression	postfix operation	stack
1	a+b*c+d*e↑f		
2	+b*c+d*e↑f	a	
3	b*c+d*e↑f	a	+
4	*c+d*e↑f	ab	+
5	c+d*e↑f	ab	+*
6	+d*e↑f	abc	+*
7	d*e↑f	abc*+	+
8	*e↑f	abc*+d	+
9	e↑f	abc*+d	+*
10	↑f	abc*+de	+*
11	f	abc*+de	+* ↑
12		abc*+def	+* ↑

So postfix expression will be (after emptying the stack)

abc*+def↑*+

2. Suppose one character at a time comes as an input from a string of letters. There is an option either to (i) print the incoming letter or to (ii) put the incoming letter on to a stack. Also a letter from top of the stack can be popped out at any time and printed. The total number of total distinct words that can be formed out of a string of three letters in this fashion, is

(A)
(B)
(C)

Total no of distict word can be 5.

because total no words starting with a would be 2. (abc ,acb)

Total no of words starting with 2 (bca,bac)

Total no of words starting with c will be 1 (cba) because to print c as first letter we have to push a and b in stack

So total no of words formed = 2+2+1

3. The following sequence of operations is performed on a stack :

PUSH (10), PUSH (20), POP, PUSH (10), PUSH (20), POP, POP, POP, PUSH (20), POP.

The sequence of values popped out is:

A. 20, 10, 20, 10, 20
B. 20, 20, 10, 10, 20
C. 10, 20,20,10,20
D. 20,20,10,20,10
E. None of the above

[B]

String	**Status of stack**	**Status of array(output)**
Push(10)	10	
Push(20)	10 20	
Pop	10	20
Push(10)	10 10	20
Push(20)	10 10 20	20
Pop	10 10	20 20
Pop	10	20 20 10
Pop		20 20 10 10
Push(20)	20	20 20 10 10
Pop		20 20 10 10 20

4. A stack is use to pass parameters to procedures in a procedure call.

(a) If a procedure P has two parameters as described in procedure definition:

procedure P(var x: integer; y: integer);

and if P is called by:

P(a,b)

State precisely in a sentence what is pushed onto stack for parameters a and b.

[QUESTION IS HIGHLY AMBIGUOUS]

In the generated code for the body of procedure P, how will the addressing of formal parameters y and y differ?

5. Which of the following permutation can be obtained in the output (in the same order) using a stack assuming that the input is the sequence1, 2, 3, 4, 5 in that order?

(A) 3, 4, 5, 1, 2
(B) 3, 4, 5, 2, 1
(C) 1, 5, 2, 3, 4
(D) 5, 4, 3, 1, 2

[B] As it can be verified as- push(1) push(2) push(3) pop push(4) pop push(5) pop pop pop

6.

The postfix expression for the infix expression

A + B*(C+D)/F + D*E is

(a) AB+CD+*F/D+E*
(b) ABCD+*F/DE*++
(c) A*B+CD/F*DE++
(d) A+*BCD/F*DE++

Possible Answer:

Infix expression	Post fix operation	Operater stack
A+B*(C+D)/F+D*E		
+B*(C+D)/F+D*E	A	
B*(C+D)/F+D*E	A	+
*(C+D)/F+D*E	AB	+
(C+D)/F+D*E	AB	+
C+D)/F+D*E	AB	+*(
+D)/F+D*E	ABC	

D)/F+D*E	ABCD	
)/F+D*E	ABCD	+*(+
/F+D*E	ABCD+	+*
F+D*E	ABCD+*	+/
+D*E	ABCD+*F	+/
D*E	ABCD+*F/+	+
*E	ABCD+*F/+	+*
E	ABCD+*F/+DE	+*
	ABCD+*F/+DE*+	

7. 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 is more efficient than implementing QUEUES
iv Last-in-first-out QUEUES 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

[A]

List perform almost all basic operation in O(1) accept merging

But array will take O(n) for all operations.

Here basic operation means insert delete at given position and

Queue using linked list are efficient in using memory. They do not waste space like in array implementation .

8. Compute the postfix equivalent of the following infix expression.

3 * log (x + 1) – a/2

3X1+log*a2/-   (assuming log as a operator not as a function and precedence of log is greatest)

9. A queue Q containing n items and an empty stack S are given. It is required to transfer all the items from the queue to the stack, so that the item at the front of the queue is on the top of the stack, and the order of all the other items is preserved. Show this how this can be done in O(n) time using only a constant amount of additional storage. Note that the only operations which can be performed on the queue and stack are Delete, Insert, Push and Pop. Do not assume any implementation of the queue or stack.

It can be done as –

Step1. Delete all the nodes of queue and push all the elements in stack sequentially.

Step2. Now queue is empty then pop every element form stack and inset it into queue

Step3. Now again repeat the step 1.

10. Which of the following is essential for converting an infix expression to the postfix form efficiently?

(A) An operator stack   (B) An operand stack

(C) An operand stack and an operator stack   (D) A parse tree

[A] Operater stack will be used to store operaters as the appear based on three rules

1. if operater ‘x’ is on top and operater ‘y’comes which has a higher precedence than x the it will also be pushed to stack..
2. if x and y have same precedence then x will be poped out before y is pushed
3. same as rule 2 if x has a higher precedence than y.

11. A priority queue Q is used to implement a stack S that stores characters. PUSH(C) is implemented as INSERT(Q, C, K) where K is an appropriate integer key chosen by the implementation. POP is implemented as DELETEMIN(Q). For a sequence of operations, the keys chosen are in

(A) non-increasing order   (B) non-decreasing order

(C) strictly increasing order   (D) strictly decreasing order

[D] Implementing stack using priority queue require first element inserted in stack will be deleted at last, and to implement it using deletemin() operation of queue will require first element inserted must have highest priority so the key must be in decreasing order.

12. Suppose a stack implementation supports, in addition to PUSH and POP, an operation REVERSE, which reverses the order of the elements on the stack.

(a) To implement a queue using the above stack implementation, show how to implement ENQUEUE using a single operation and DEQUEUE using a sequence of 3 operations (2)

(b) The following postfix expression, containing single digit operands and arithmetic operators + and *, is evaluated using a stack.

5 2 * 3 4 + 5 2 * * +

a)

i)Implementation of enqueue is as-

Step1: push the element in stack

ii) implementation of dequeue is as-

Step1: reverse

Step2: pop

Step3:reverse

Show the contents of the stack

i After evaluating 5 2 * 3 4 +
ii After evaluating 5 2 * 3 4 + 5 2
iii At the end of evaluation

b)

Input in postfix	Stack<- top	evaluation
5	5	
2	52	
*	10	5*2=10
3	10 3	
4	10 3 4	
+	10 7	3+4=7
5	10 7 5	
2	10 7 5 2	
*	10 7 10	5*2=10
*	10 70	7*10=70

+	80	
Null		

i)contents of stack are 10 7

ii)contents of stack are10 7 5 2

iii)contents of stack null and the final answer is 80

13. What is the minimum number of stacks of size n required to implement a queue of size n?
   A. One
   B. Two
   C. Three
   D. Four

[B] First stack will be used to store the queue with rear at top..

We need to push element in to stack 1 in order to insert it in queue.

For deletion we will empty stack 1 in stack 2.that will put front on top so we can pop from stack 2 in order to effect deletion.

14. Let S be a stack of size $n \geq 1$. Starting with the empty stack, suppose we Push the first n natural numbers in sequence, and then perform n Pop operations. Assume that Push and POP operations take X seconds each, and Y seconds elapse between the end of one such stack operation and the start of the next operation. For $m \geq 1$, define the stack-life of m as the time elapsed from the end of Push(m) to the start of the Pop operation that removes m from S. The average stack-life of an element of this stack is

   (A) $n(X + Y)$
   (B) $3Y + 2X$
   (C) $N(X + Y) – X$
   (D) $Y + 2X$

[A]

Push1() and pop1() are push and pop operations in 1st stack

Push2() and pop2() are push and pop operations in 2nd stack

Using 1st stack we can insert value same as queue (push1())

And for deletion we have to pop all the elements of 1st stack into push it into 2nd stack and then pop one element form 2nd stack.(pop1()push2()…..until 1st stack is emptied then pop2()…for desired number of deletion)

15. The following sequence of operations is performed on a stack :

PUSH (10), PUSH (20), POP, PUSH (10), PUSH (20), POP, POP, POP, PUSH (20), POP.

The sequence of values popped out is:

F. 20, 10, 20, 10, 20
G. 20, 20, 10, 10, 20
H. 10, 20,20,10,20
I. 20,20,10,20,10
J. None of the above

[G]

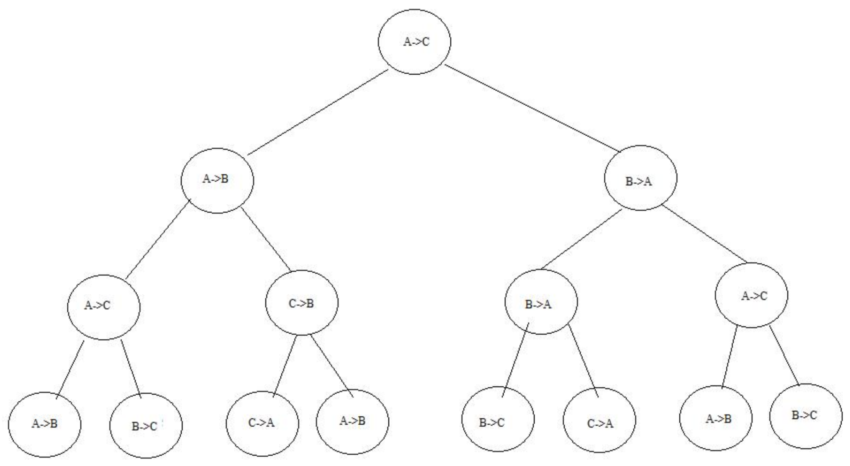
String	**Status of stack**	**Status of array(output)**
Push(10)	10	
Push(20)	10 20	
Pop	10	20
Push(10)	10 10	20
Push(20)	10 10 20	20
Pop	10 10	20 20
Pop	10	20 20 10
Pop		20 20 10 10
Push(20)	20	20 20 10 10
Pop		20 20 10 10 20

16. Consider three pegs A, B, C and four disks of different sizes. Initially, the four disks are stacked on peg A, in order of decreasing size. The task is to move all the disks from peg A to peg C with the help of peg B. the moves are to be made under the following constraints:

[i] In each step, exactly one disk is moved from one peg to another.
[ii] A disk cannot be placed on another disk of smaller size. If we denote the movement of a disk from one peg to another by y → y, where y, y are A, B or C, then represent the sequence of the minimum number

of moves to accomplish this as a binary tree with node labels of the form (y → y) such that the in-order traversal of the tree gives the correct sequence of the moves..

If there are n disks, derive the formula for the total number of moves required in terms of n



For one disk the steps involved is A->C ie 1 step

For two disks the steps involved are A->B,A->C,B->C ie 3steps

For three disks the steps involved are A->C,A->B,C->B,A->C,B->A,B->C,A->C ie 7steps

For 4 disks the steps involved are A->B,A->C,B->C,A->B,C->A,C->B,A->B,A->C,B->C,B->A,C->A,B->A,A->B,A->C,B->C ie 15 steps

So for 1 disk		2^1-1=1
	2 disks	2^2-1=3
	3 disks	3^2-1=7
	4 disks	4^2-1=15
	…….	……….
	…….	……….
	n disks	2^n-1 (by induction)

17. A stack is use to pass parameters to procedures in a procedure call.

(b) If a procedure P has two parameters as described in procedure definition:

```
procedure P(var x: integer; y: integer);
```

and if P is called by:

P(a,b)

State precisely in a sentence what is pushed onto stack for parameters a and b.

In the generated code for the body of procedure P, how will the addressing of formal parameters y and y differ?

[QUESTION IS HIGHLY AMBIGUOUS]

18. Which of the following permutation can be obtained in the output (in the same order) using a stack assuming that the input is the sequence1, 2, 3, 4, 5 in that order?

(E) 3, 4, 5, 1, 2
(F) 3, 4, 5, 2, 1
(G) 1, 5, 2, 3, 4
(H) 5, 4, 3, 1, 2

[F] As it can be verified as- push(1) push(2) push(3) pop push(4) pop push(5) pop pop pop

19. Consider the following C program:

```
#include<stdio.h>

#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( );

                                    are = (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 + * +

(A) 15 (B) 25 (C) 30 (D) 150

[B] Sequence of operations will be

1. push(5)
2. push(2)
3. m = 2;n=5 ;r=5*2
4. push(r) /*push(10)*/
5. push(3)
6. push(3)
7. push(2)
8. m=2; n= 3;r=3+2
9. push(r) /* push(5)*/
10. r=5*3
11. push(r) /*push(15)*/
12. m=15;n = 10;r=15+10
13. push(r)
14. printf("%d",pop());

Which will print 25

20. Suppose you are given an implementation of a queue of integers. The operations that can be performed on the queue are:

isEmpty(Q)	– returns true if the queue is empty, false otherwise.
delete(Q)	– deletes the element at the front of the queue and returns its value.
insert(Q, i)	– 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);
        }
    }
```

What operation is performed by the above function f?

(A) Leaves the queue Q unchanged
(B) Reverse the order of 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.

[B] Due to recursion the last element deleted will be the first to be inserted so element that was at rear will be at front in new queue and the second last element will be at second position hence the queue will be reversed.

21. Let *w* be the minimum weight among all edge weights in an undirected connected graph. Let *e* be a specific edge of weight *w*. Which of the following is **FALSE**?

(A) There is a minimum spanning tree containing *e.*
(B) If *e* is not in a minimum spanning tree *T,* then in the cycle formed by adding *e* to T, all edges have the same weight.
(C) Every minimum spanning tree has an edge of weight *w*.
(D) *e* is present in every minimum spanning tree.

[D]

22. Consider the following C function:

```
int  f(int n)
    {
     static int are = 0;
     If (n < = 0) return 1;
     If (n > 3)
         { r = n;
           return f (n – 2) + 2;
         }
     return f(n – 1) + r;
}
```

What is the value of f(5)?

(A) 3 (B) 7 (C) 9 (D) 18

[D]

f(5) will return f(3)+2

f(3) will return f(2)+5

f(2) will return f(1)+5

f(1) will return f(0)+5

f(0) will return 1

f(1) will return 6

f(2) will return 11

f(3) will return 16

f(5) will return 18

23. 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

[A]

Symbols read so far	Operand stack
8	8
8 2	8 2
8 2 3	8 2 3
8 2 3^	8 8
8 2 3^/	1
8 2 3^/2	1 2

8 2 3^/2 3	1 2 3
8 2 3^/2 3 *	1 6

24. An implementation of queue Q, using stacks S1 and S2 is given below:

```
void insert (Q, x) {
        push (S1, x);
}
void delete (Q) {
        if (stack-empty (S2)) then
                if (stack-empty (S1)) then {
                        print ( " Q is empty" );
                        return;
                }
                else while (! (stack-empty (S1))) then {
                                x = pop (S1);
                                push (S2, x);
                }
        x = pop (S2);
}
```

Let n insert and m $(\leq n)$ delete operations be performed in an arbitrary order on an empty queue Q. let x and y be the number of push and pop operations performed respectively in the process. Which one of the following is true for all m and n?

(A) $n + m \leq x < 2n$ and $2m \leq y \leq n + m$ (B) $n + m \leq x < 2n$ and $2m \leq y \leq 2n$

(C) $2m \leq x < 2n$ and $2m \leq y \leq n + m$ (D) $2m \leq x < 2n$ and $2m \leq y \leq 2n$

[A]

Extreme case will be when all the elements are inserted and then deleted

For each insert we have a push operation

And for first delete we have n push operations if n elements are in stack, so n+m<=x in worst case

And no of pushes is also 2n : n for n insert and n for first delete

So (n+m)<= x < 2n

Also n pops for first delete and m-1 one pops for m-1 deletes so y= n+m-1 in extreme case y< n+m-1

25. A function f defined on stacks of integers satisfies the following properties.

f(ϕ) = 0 and

f(push(**S**, i) = max(f(**S**), 0) + i for all stacks **S** and integers i.

If a stack **S** contains the integers 2, -3, 2, -1, 2 in order from bottom to top, what is f(**S**)?

(A) 6 (B) 4 (C) 3 (D) 2

[C] Answer is 3

Step1:top s=2

Step2: tops= -1

Step3: tops= 2

Step4: top s=1

Step5: top s=3

26. Assume that the operators +, -, × are left associative and ^ is right associative. The order of precedence (from highest to lowest) is ^, ×, +, -. The postfix expression corresponding to the infix expression a+b×c-d^e^f is

(A) abc×+def ^ ^ -
(B) abc×+de ^ f ^ -
(C) ab+c×d-e ^f ^
(D) -+a×bc ^ ^ def

[A]

Infix string	**Stack**	**Postfix array**
a		a
+	+	a
b	+	ab
*	+*	ab

c	+*	abc
-	-	abc*+
d	-	abc*+d
^	-^	abc*+d
e	-^	abc*+de
^	-^^ (not poped because of right associative)	abc*+de
f	-^^	abc*+def
	-^	abc*+def^
	-	abc*+def^^
		abc*+def^^-

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

(A) Queue
(B) Stack
(C) Tree
(D) List

[B] We can put every opening parenthesis in stack and pop every time we get a closed one. In the ened if we get stack empty then we had balanced parenthesis else not.