Academic year 2025-2026 (ODD Sem)

……

Date: 07/11/2025	**CIE - 1**	**Max. Marks:60**
Semester: III	**UG**	**Duration: 2 Hour**
Course Title: Data Structures and Applications		**Course Code:** IS233A

SCHEME & SOLUTIONS

S No	Solutions with Scheme	Marks
	PART A	

1.1	void push2(struct stack * s, int x)
	{
	if (s-> top2 – 1 == s->top1)
	{
	printf("Stack Overflow");
	exit(0)
	}
	s-> top2--;
	s->items[s->top2]=x;
	}
	Overflow condition : 1M
	Update and store : 1M
		2

1.2	int isQueueEmpty(struct queue *q)
	{
	if (q->rear == q->front)
	return1;
	return 0;
	}
		2

1.3	A B C + * D E / F G ^ ^ H K ^ * * I + 1M + * * A + B C * ^ / D E ^ F G ^ H K I 1M	**2**
1.4	Linear Data structure / Non-Linear Data Structure Elements are arranged sequentially, one after another. / Elements are connected in a hierarchical or network-like structure. Traversed in a single run — one element after the other. / Traversal is done through multiple paths (branches or links). Every element has a unique predecessor and successor (except first and last). / Each element can be connected to multiple elements (many-to-many relationships). Array, Linked List, Stack, Queue / Tree, Graph Simple and easy to implement. / More complex to implement and manage. Any two differences -2M	**2**

1.5 — **2**

```
int findLargest(int arr[], int n) {
    if (n == 1)
        return arr[0];
    int maxRest = findLargest(arr, n - 1);
    if (arr[n - 1] > maxRest)
        return arr[n - 1];
    return maxRest;
}
```

PART B

1. a

```
Initialize the operstk as empty stack
Scan in/p from left to right
while not end of input string
      symb= next input character
      If(symb is an operand)
            Append symb to the postfix string
      Else if (symb is ')')
           topsymb=pop(operstk)
           while(topsymb != '(' )
                 Append topsymb to postfix string
                 topsymb=pop(operstk)
           End while
       Else
          topsymb=stacktop(operstk)
          while(precedence(topsymb) > precedence(symb))
                 topsymb=pop(operstk)
                 Append topsymb to postfix string
          end while
          Push(operstk, symb)
    End if
End while // end i/p
while( !stackEmpty(operstk))                    // not stack empty
     topsymb=pop(operstk)
      Append topsymb to postfix string
End while
```

1 b		5

Step	Input Symbol	Stack	Output
1	i	—	i
2	+	+	i
3	)	+)	i
4	h	+)	i h
5	^	+)^	i h
6	g	+)^	i h g
7	^	+)^^	i h g
8	f	+)^^	i h g f
9	-	+)-	i h g f ^ ^
10	)	+)-)	i h g f ^ ^
11	e	+)-)	i h g f ^ ^ e
12	*	+)-)*	i h g f ^ ^ e
13	d	+)-)*	i h g f ^ ^ e d
14	-	+)-)-	i h g f ^ ^ e d *
15	c	+)-)-	i h g f ^ ^ e d * c
16	(	+)-	i h g f ^ ^ e d * c -
17	/	+)-/	i h g f ^ ^ e d * c -
18	b	+)-/	i h g f ^ ^ e d * c - b
19	*	+)-/*	i h g f ^ ^ e d * c - b
20	a	+)-/*	i h g f ^ ^ e d * c - b a
21	(	+	i h g f ^ ^ e d * c - b a * / -
22	—	Empty	i h g f ^ ^ e d * c - b a * / - +

Prefix expression is : +-/*ab-c*de^^fghi

2 a	**Properties**	5

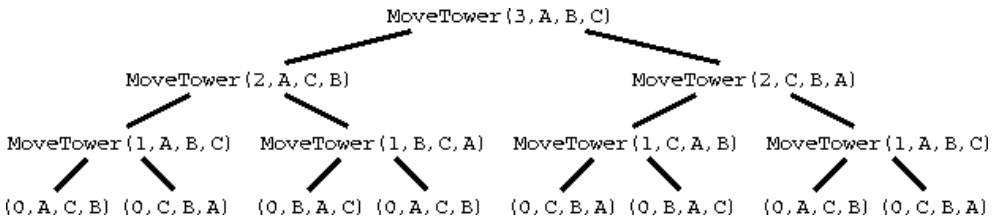
1. Base Case (Termination Condition): Every recursive function must have at least one condition to ***stop recursion***. Without it, the function would call itself infinitely (leading to stack overflow).
2. Recursive Case: The function calls itself with a smaller subproblem to move toward the base case.

```
int binarySearch(int arr[], int low, int high, int key) {
   if (low > high)
      return -1;

   int mid = (low + high) / 2;

   if (arr[mid] == key)
      return mid;  // Element found
   else if (key < arr[mid])
      return binarySearch(arr, low, mid - 1, key);
   else
      return binarySearch(arr, mid + 1, high, key);
}
```

First call : binarySearch(arr, 0, n - 1, key);
Properties 2 M + Function 3M

2b.	One of the call tree :3M  Moves 2M 1 Move disk 1 from A → C 2 Move disk 2 from A → B 3 Move disk 1 from C → B 4 Move disk 3 from A → C 5 Move disk 1 from B → A 6 Move disk 2 from B → C 7 Move disk 1 from A → C	5
1.	1.The main drawback of a linear queue lies in inefficient use of memory (space wastage) due to the way elements are inserted and deleted- . 1M Illustrative example 2M 2. Circular queue : Initialization- 1M (Insert , delete and delete logic according to the initialization used) Insert 2M Delete 2M Display 2M	10
4a.	With respect to the speed of evaluation , prefix and postfix expressions are faster than the infix expression because they allow direct ,single-pass evaluation without needing to check precedence ,associativity or parenthesis -2M Prefix expression evaluation Algorithm-4M	6

4b.

Step	Symbol Scanned	Action	Stack (Top → Bottom)
1	5	Push	5
2	3	Push	3, 5
3	*	Pop(3,5) → 3*5=15 → Push 15	15
4	3	Push	3, 15
5	2	Push	2, 3, 15
6	^	Pop(2,3) → 2^3=8 → Push 8	8, 15
7	4	Push	4, 8, 15
8	8	Push	8, 4, 8, 15
9	/	Pop(8,4) → 8/4=2 → Push 2	2, 8, 15
10	3	Push	3, 2, 8, 15
11	-	Pop(3,2) → 3-2=1 → Push 1	1, 8, 15
12	1	Push	1, 1, 8, 15
13	*	Pop(1,1) → 1*1=1 → Push 1	1, 8, 15
14	7	Push	7, 1, 8, 15
15	+	Pop(7,1) → 7+1=8 → Push 8	8, 8, 15
16	-	Pop(8,8) → 8-8=0 → Push 0	0, 15
17	+	Pop(0,15) → 0+15=15 → Push 15	**15 :Answer**

4

5

```
int isValid(char *s) {     -- 5M
    struct stack stack;
    stack.top=-1;
    for (int i = 0; s[i] != '\0'; i++) {
        char current = s[i];
        if (current == '{' || current == '(' || current == '[') {
            push(&stack, current);
        }

        else if (current == '}' || current == ')' || current == ']') {
            if (isEmpty(&stack)) {
                return 0;
            }
            char top = pop(&stack);

            if ((current == '}' && top != '{') ||
                (current == ')' && top != '(') ||
                (current == ']' && top != '[')) {
                return 0;
            }
        }
    }
return isEmpty(&stack);
}
```

Remaining Part of the Program : 5M
Stack Definition Declaration 1M
Push - 1.5M Pop -1.5 M
Main Program : Declaration , Input and Output 1M

10M

Academic year 2025-2026 (ODD Sem)