**Algorithms**

**1)Bubble sort**

1. Repeat For J = 1 to N
2. Repeat For K = 1 to N-J
3. If (A[K] > A[K+1]) Then
4. Interchange A[K] and A[K+1]
5. [End of If]
6. [End of Step 2 For Loop]
7. [End of Step 1 For Loop]
8. Exit

**2)Insertion sort**

1. Repeat For J = 2 to N

2. Set TEMP = A[J]

3. Set K = J – 1

4. Repeat While (K >= 1) and (A[K] > TEMP)

5. Set A[K+1] = A[K]

6. Set K = K – 1

[End of While Loop]

7. Set A[K+1] = TEMP

[End of For Loop]

8.Exit

**3)Selection sort**

1. Repeat For J = 1 to N

2. Set MIN = J

3. Repeat For K = J+1 to N

4. If (A[K] < A[MIN]) Then

5. Set MIN = K

[End of If]

[End of Step 3 For Loop]

6. Interchange A[J] and A[MIN]

[End of Step 1 For Loop]

7. Exit

**4)push**

Procedure: PUSH (STACK,TOP,MAX,ITEM)

//This procedure pushes an ITEM onto stack.

1. [Stack already full]

If TOP=MAX, then Print: UNDERFLOW, and Return.

1. Set TOP: =TOP+1 [Increase TOP by 1]
2. SetSTACK [TOP]:=ITEM. [Insert ITEM in new TOP position]
3. Return.

**5)pop**

Procedure: POP (STACK,TOP,ITEM)

//This procedure Deletes the top element of STACK and assigns it to the variable ITEM.

1. [Stack Empty or Not]

If TOP=0, then Print: OVERFLOW, and Return.

1. Set ITEM: =STACK [TOP]. [Assigns Top element to ITEM]
2. Set TOP: =TOP-1 [Decrease TOP by 1]
3. Return.

**6)Write algorithm for infix to postfix evaluation?**

Algorithm Evaluate( ):

Description:

Here Pis apostfix expression and this algorithm evaluates it.

1.Add a “)” right parenthesis at the end of P.

2.Scan P from left to right and repeat steps 3 & 4 for each element of P until “)” is

Encountered.

3.If an operand is encountered, push it onto stack.

4.If an operator is encountered then:

(a)Pop the top two elements from stack, where A is the

Top element and B is the next to top element.

(b)Evaluate B A.

©Place the result of (b)back on stack.

[End of Step 4 If]

[End of step 2 For Loop]

5.Set VALUE equal to the top element on the stack.

6.Exit.

**7)Write algorithm to insert element In Queue?**

**Algorithm Insert( )**

1.If (REAR == N) Then[Check for overflow]

2.Print: Overflow

3.Else

4.If (FRONT and REAR == 0)Then[Check if QUEUE is empty]

(a)Set FRONT = 1

(b)Set REAR = 1

5.Else

6.Set REAR = REAR + 1[Increment REAR by 1]

[End of Step 4 If]

7.QUEUE[REAR] = ITEM

8.Print: ITEM inserted[End of Step 1 If]

9.Exit

**8)Write algorithm to insert element In Queue?**

**Algorithm Delete ( ):**

Description: Here QUEUE is an array with N

Locations. FRONT and REAR points to the front and

rear of the QUEUE.

1. If (FRONT == 0) Then [Check for underflow]

2. Print: Underflow

3. Else

4. ITEM = QUEUE[FRONT]

5. If (FRONT == REAR) Then [Check if only one

element is left]

(a) Set FRONT = 0

(b) Set REAR = 0 6.

Else

7. Set FRONT = FRONT + 1 [Increment FRONT by 1]

[End of Step 5 If]

8. Print: ITEM deleted [End of Step 1 If]

9. Exit.

**10)Algorithm Insert\_front (DEQUE)**

step1. Start

step2. Check the queue is full or not as if (r == max-1) &&(f==0)

step3. If false update the pointer f as f= f-1

step4. Insert the element at pointer f as Q[f] = element

step5. Stop

**11)Alogrithm insert\_back (DEQUE)**

Step1: Start

Step2: Check the queue is full or not as if (r == max-1)

&&(f==0) if yes queue is full

Step3: If false update the pointer r as r= r+1

Step4: Insert the element at pointer r as Q[r] = element

Step5: Stop

**12)Alogrithm Remove\_front (DEQUE)**

Step1: Start

Step2: Check the queue is empty or not as if (f == r) if yes queue is

Empty.

Step3: If false update pointer f as f = f+1 and delete element at

Position f as element = Q[f]

Step4: If ( f== r) reset pointer f and r as f = r = -1

Step5: Stop

**13)Alogrithm Remove\_back(DEQUE)**

Step1. Start

Step2. Check the queue is empty or not as if (f == r) if yes queue

Is empty

Step3. If false delete element at position r as element = Q[r]

Step4. Update pointer r as r = r-1

Step5. If (f == r ) reset pointer f and r as f = r= -1

Step6. Stop

**14)** **Algorithm to traverse array?**

1.Repeat For I = LB to UB

2.Apply PROCESS to A[I]

[End of For Loop]

3.Exit