1-1-24

LAB-2

(I) Write a program to simulate the working of stack using an array with the following: (a) Push (b) Pop (c) Display
→ The program should print appropriate messages for stack overflow, stack underflow

```
#include<stdio.h>
#include<stdlib.h> #define SIZE 4
void push(int);
void pop();
void display();
int st[SIZE], top=-1;

int main(){
  int value, choice;
  while(1)
{ printf("Enter 1 to push an element\n
          Enter 2 to pop an element\n
          Enter 3 to display an element\n
          Enter 4 to exit the program");
scanf("%d", &choice);
switch(choice){
    case 1: printf("Enter a value\n");
            scanf("%d", &value);
            push(value);
            break;
    case 2: pop();
            break;
    case 3: display();
            break;
    case 4: exit(0);
```

```
default : printf ("wrong input - try again");
}
}
void push(int value) {
    if (top == SIZE-1) {
        printf ("overflow condition"); }
    else {
        top++;
        stack[top] = value;
    }
}

void pop() {
    if (top == -1) {
        printf("Stack is empty : underflow"); }
    else {
        printf(" The deleted item is %d", stack[top] )
        top = top-1; }
}

void display () {
    int i;
    if (top == -1) {
        printf (" empty stack: ");
    }
for (int i = top; i >= 0; i--) {
        printf ("%d \t", stack[i]); }
}
```

```
case '+': case '-': case '*': case '/': case '^':
while (pred(stack[top]) >= pred(symbol))
{ temp = pop();
postfix[pos++] = temp; }
push(symbol); break;
default: postfix[pos++] = symbol;
}
index++; }
while(top > -1) { temp = pop();
postfix[pos++] = temp;
}}
void push(char symbol) {
top = top+1;
stack[top] = symbol;
}
char pop() {
char symb;
symb = stack[top];
top = top-1;
return(symb);
}
int pred(char symbol) {
int p; switch(symbol) {
case '^': p=3; break;
case '*': case '/': p=2; break;
case '+': case '-': p=1; break;
case '(': p=0; break;
}
return(p);
}
```

11) WAP to convert a given valid parantheses infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide)

```
#include <stdio.h>
#include <stdlib.h>
int index=0, pos=0, top=-1, length;
char symbol, temp, infix[20], postfix[20], stack[20];
void infixtopostfix();
void push(char symbol);
char pop();
int pred(char symb);
void main(){
printf("Enter infix expression \n");
scanf("%s", infix);
infixtopostfix();
printf("Postfix expression: \n%s", postfix); }
void infixtopostfix()
{ length= strlen(infix);
  push('#');
  while(index++ <length)
{ symbol = infix[index++];
  switch(symbol){
  case '(': push(symbol);
      break;
case ')': temp= pop();
      while(temp != '(')
      {
postfix[pos]= temp;
  pos++; temp= pop(); }
      break;
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
 4)Exit
1
20
The value pushed is 201)Push
 2)Pop
 3) Display
 4)Exit
3
20
10
1)Push
 2)Pop
 3) Display
 4)Exit
2
The value deleted is 201)Push
 2)Pop
 3) Display
 4)Exit
3
10
1)Push
 2)Pop
 3) Display
 4)Exit
```