Experiment No. 2

Aim: Convert an Infix expression to postfix expression using stack ADT.

Theory:

To convert infix expression to postfix expression, we will use the stack data structure. By scanning the infix expression from left to right, when we will get any operand, simply add them to the postfix form, and for the operator and parenthesis, add them in the stack maintaining the precedence of them.

Algorithm

infixToPostfix(infix)

Input − Infix expression.

Output − Convert infix expression to postfix form.

```
Begin
  initially push some special character say # into the stack
  for each character ch from infix expression, do
    if ch is alphanumeric character, then
      add ch to postfix expression
    else if ch = opening parenthesis (, then
      push ( into stack
    else if ch = ^, then          //exponential operator of higher precedence
      push ^ into the stack
    else if ch = closing parenthesis ), then

      while stack is not empty and stack top ≠ (,
        do pop and add item from stack to postfix expression
      done
      pop ( also from the stack
    else
      while stack is not empty AND precedence of ch <= precedence of stack top element, do
        pop and add into postfix expression
      done
```

push the newly coming character.

done

while the stack contains some remaining characters, do

pop and add to the postfix expression

done

return postfix

End

Conclusion: (Students write conclusion in your own words. U have to describe what u you understood from the experiment and the concept of the experiment. Conclusion carry 4 marks out of 10