14. Evaluation of postfix expression

The postfix notation is used to represent algebraic expressions. The expressions written in postfix form are evaluated faster compared to infix notation as paranthesis is not required in postfix.

Steps for evaluating postfix expression

```
Step1: Accept postfix expression on string is post variable
Step2: For i in post:
        If i is an operand:
            push i in stack:
        Else:
            pop two elements from stack eg. x and Y
            perform operation with current operator on both the
            parents i.e x:y
            push the result back into the stack
       Find for loop.
Step 3: Finally pop out the result from stack
```

Evaluate below postfix expression: 234+*6−

	Value i	operation	stack
1)	2	Push 2 in stack	2
2)	3	push 3 in stack	3 / 2
3)	4	push 4 in stack	4 / 3 / 2
4)	+	Pop 2 elements from stack and Performing addition operation.	7 / 2
5)	*	Pop 2 elements from stack & Perform multiplication operation.	14
6)	6	Push 6 in stack	6 / 4
7)	−	Pop 2 elements from stack & Perform substraction operation.	8
8)		pop result from stack and display	

Code:

```
class evaluate_postfix:
  def __init__(self):
        self.item=[ ]
        self.size=-1
  def isEmpty(self):
      return self.item==[ ]
  def push(self,item):
     self.items.append(item)
     self.size+=1
  def pop(self):
     if self.isEmpty():
          return 0
     else:
          self.size-=1
          return self.item.pop()
  def stack(self):
     if self.isEmpty():
          return False
     else:
          return self.items[self.size]
  def evaluate(self, expr):
      for i in expr:
          if i in '0123456789':
               self.push(i)
          else:
               op1=self.pop()
               op2=self.pop()
               result=self.cal(op2,op1,i)
```

```
                self.push(result)
            return self.pop()
    def call(self,op2,op1,i):
        if i is 'x':
            return int(op2)x int(op1)
        elif i is '/':
            return int(op2)/int(op1)
        elif i is '+':
            return int(op2)+int(op1)
        elif i is '-':
            return int(op2)-int(op1)
s=evaluate_postfix
expr=input('enter the postfix expression')
value=s.evaluate(expr)
print('the result of postfix',expr,'is',value)
```

Output-

```
enter the postfix expression: 53+62/*35*+
the result of postfix expression 53+62/*35*+ is 39
```