CS & IT ENGINEERING

Data Structures Through Python

STACK

Lecture No.- 04

By- Satya sir

Recap of Previous Lecture

Infix to Prefix conversion

1) scan X from Right to left

(')' == '(' and '(' == ')')

2. Infix to Postfix conversion Procedure

$OP_T > OP_S$: Pop OP_T, append to Y, Push OP_S

$OP_T <= OP_S$: Push OP_S on to Stack.

3. Prefix Expression = Reverse (Y)

Topics to be Covered

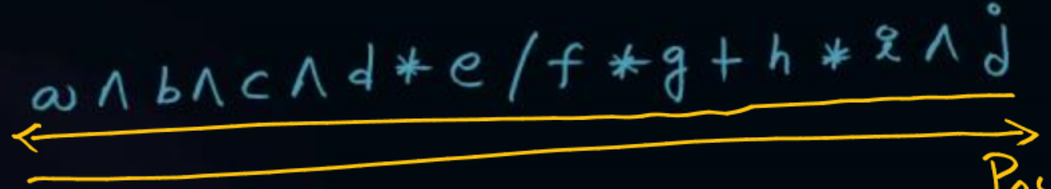
Expression Evaluation

- Postfix Expression
- Prefix Expression

H/W Problem

Convert infix Expression to Prefix and Postfix Exp.

Infix Exp : $a \wedge b \wedge c \wedge d * e / f * g + h * i \wedge j$



Postfix

Prefix

$y: j\,i \wedge h * g f e d c b a \wedge \wedge \wedge * / * +$

$Z: + * / * \wedge \wedge \wedge a b c d e f g * h \wedge i j$

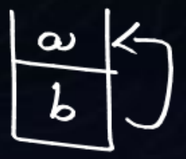
$y: a b \wedge c \wedge d \wedge e * f / g * h\,i\,j \wedge * +$

Topic : Expression Evaluation

Postfix Expression Evaluation

Procedure

1. Let 'X' be given Postfix Expression, 'S' be empty stack.
2. Scan 'X' from Left to Right, one Element at a time
3. a) If scanned element == operand, Push it on to stack, S.
   b) If scanned Element == operator:
      i) Pop the Top 2 operands from stack.
         Let a be top, b be second top.
      ii) Apply scanned operator as "b op a"
      iii) Push back result on to stack, S.



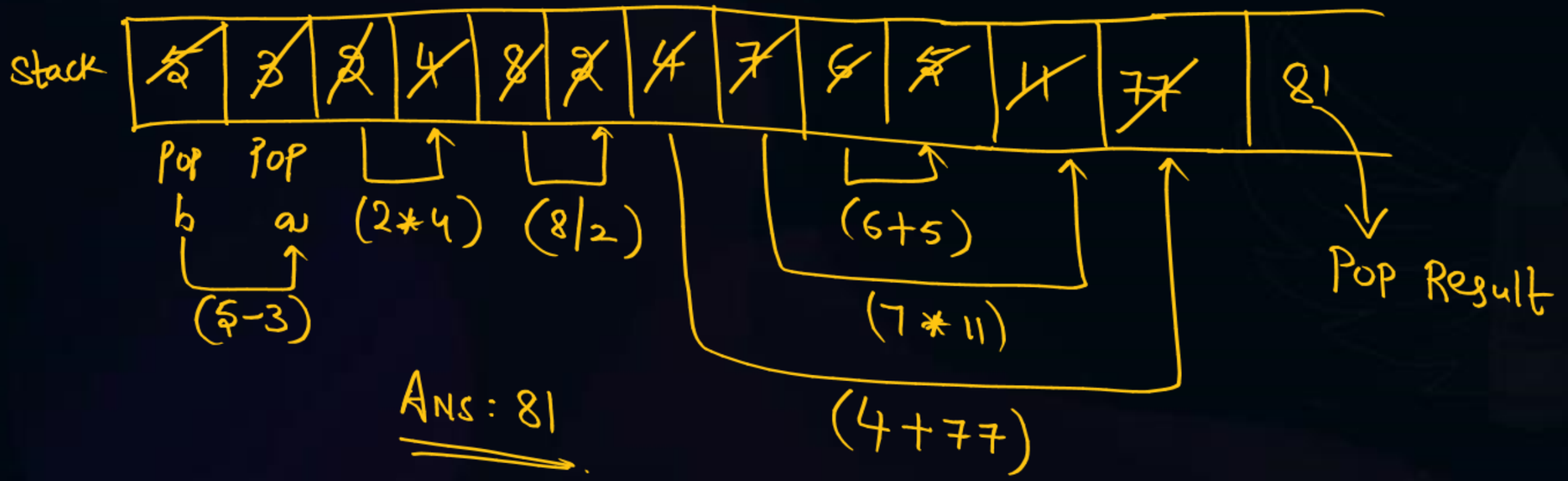
4) Repeat steps ② & ③ until Complete Expression is scanned.
5) Result will be on top. Pop the result.

Topic : Expression Evaluation

Example - 1

Evaluate the Postfix Expression

5 3 - 4 * 2 / 7 6 5 + * +



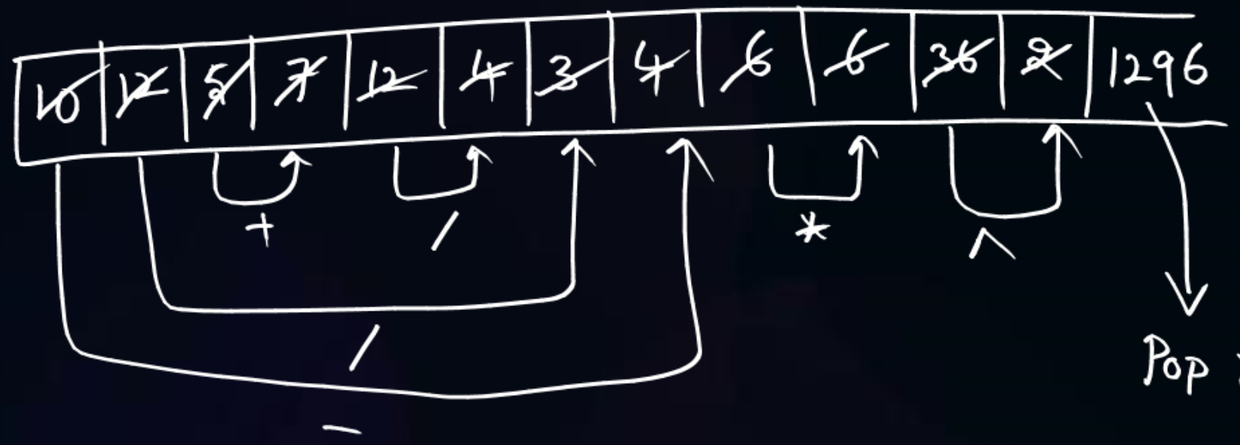
Ans : 81

Topic : Expression Evaluation

Example-2

The Result after Evaluating Postfix Expression is **1296**

10 12 5 7 + 4 / / − 6 * 2 ^



$(36^2) = 36 * 36$

$= 1296$

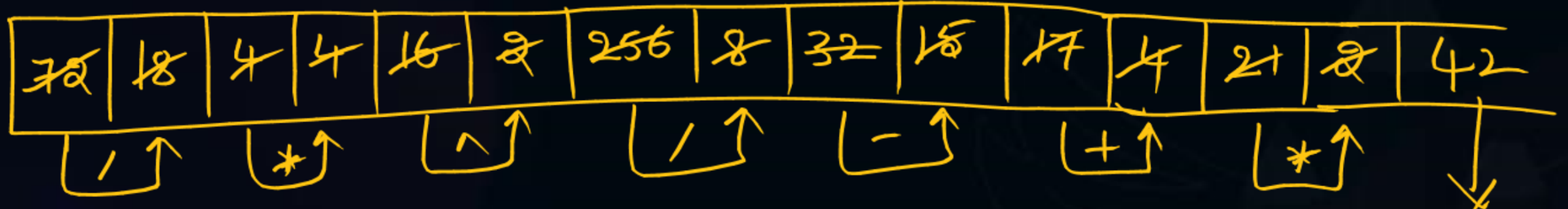
Topic : Expression Evaluation

Example-3

Evaluate Postfix Expression: 72 18 / 4 * 2 ^ 8 / 15 - 4 + 2 *

Result = _____

Stack:



Pop Result.

Topic : Expression Evaluation

Prefix Expression Evaluation

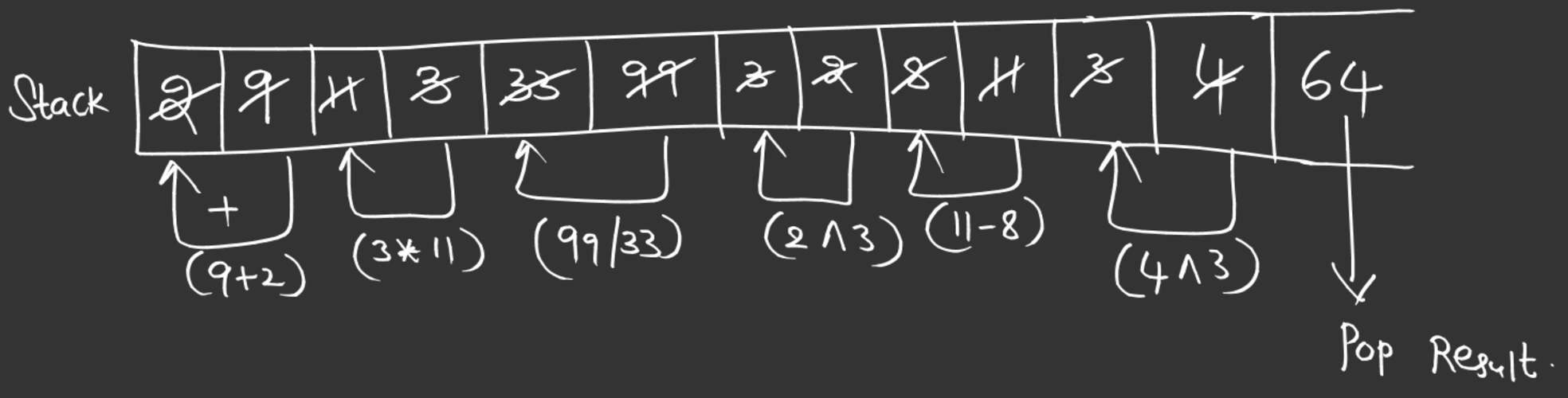
1. Let 'X' be given Prefix Expression, 'S' be Empty Stack.
2. Scan X from Right to Left, one Element at a time.
3. a) If Element == Operand, Push it on to S.
   b) If Element == operator, i) Pop, top 2 operands. Let 'a' be top, 'b' be second top.

   ii) Perform operation as (a operator b)
   iii) Push back result on to S.
4. Repeat (2) & (3) until End
5. Final Result is on top, Pop result.

Example - 1

Evaluate Prefix Expression:

^ 4 - 11 ^ 2 / 99 * 3 + 9 2

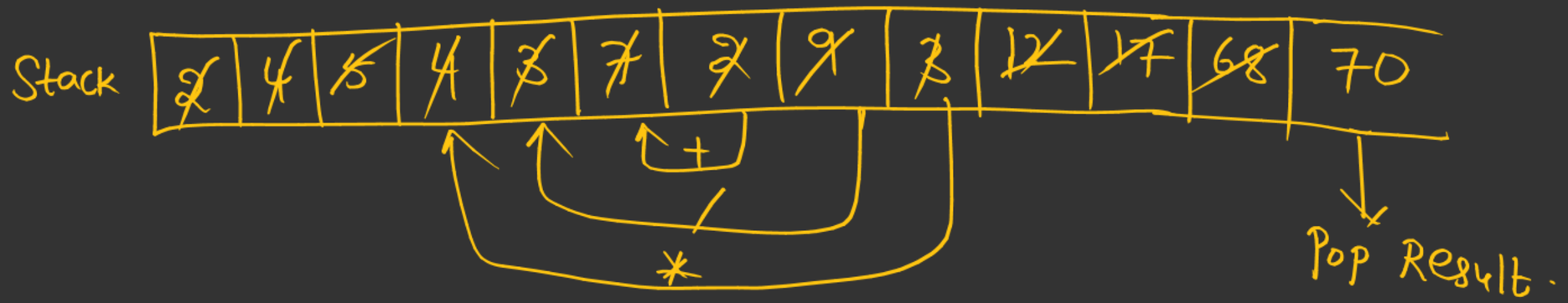


Example-2

Evaluate Prefix Expression :

+ * + * / + 2 7 3 4 5 4 2

Result = 70



Topic : Expression Evaluation

Postfix Expression Evaluation

```
a = list(input("Enter the postfix expression: ").split(" "))
stack = []
for i in a:
    if(len(stack)==1 and i==a[-1]):
        stack[0]= -stack[0]
        break
    if(i in ["+", "-", "*", "/"]):
        b = stack.pop()
        c = stack.pop()
        if(i=="+"):
            stack.append(c+b)
        elif(i=="-"):
            stack.append(c-b)
        elif(i=="*"):
            stack.append(c*b)
        elif(i=="/"):
            stack.append(c/b)
        else:
            stack.append(int(i))
print("The result of the expression: ",stack[0])
```

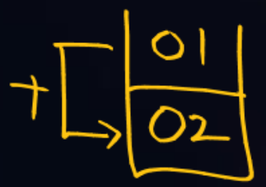
Topic : Expression Evaluation

Prefix Expression Evaluation

```
def is_operand(c):
        return c.isdigit()
```

True if operand

```
def evaluate(expression):
        stack = []
        for c in expression[::-1]:

                if is_operand(c):
                        stack.append(int(c))

        else:
        o1 = stack.pop()
        o2 = stack.pop()

    if c == '+':
        stack.append(o1 + o2)

    elif c == '-':
        stack.append(o1 - o2)

    elif c == '*':
            stack.append(o1 * o2)

        elif c == '/':
            stack.append(o1 / o2)

    return stack.pop()

if __name__ == "__main__":
    test_expression = "+5*46"
    print(evaluate(test_expression))
```



Topic : Expression Evaluation

The following postfix expression with single digit operands is evaluated using a stack:

8 2 3 ^ / 2 3 * + 5 1 * -

Note that ^ is the exponentiation operator. The top two elements of the stack after the first * is evaluated are:

A. 6, 1

B. 5, 7

C. 3, 2

D. 1, 5

Topic : Expression Evaluation

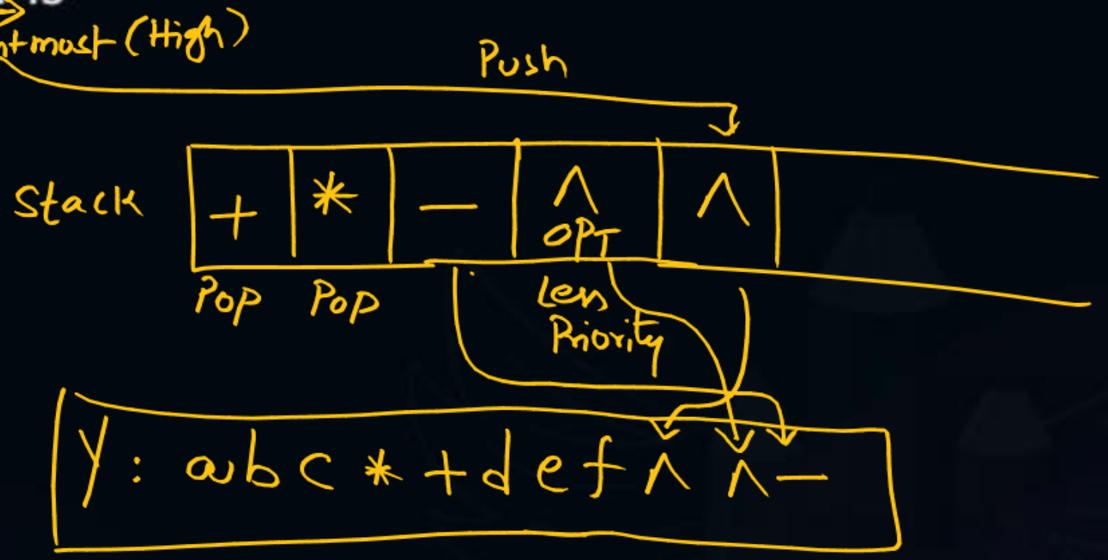
Assume that the operators +, -, × are left associative and '^' is right associative. The order of precedence (from highest to lowest) is ^, x , +, -. The postfix expression corresponding to the infix expression a + b × c - d ^ e ^ f is

A. abc × + def ^ ^ -

B. abc × + de ^ f ^ -

C. ab + c × d - e ^ f ^

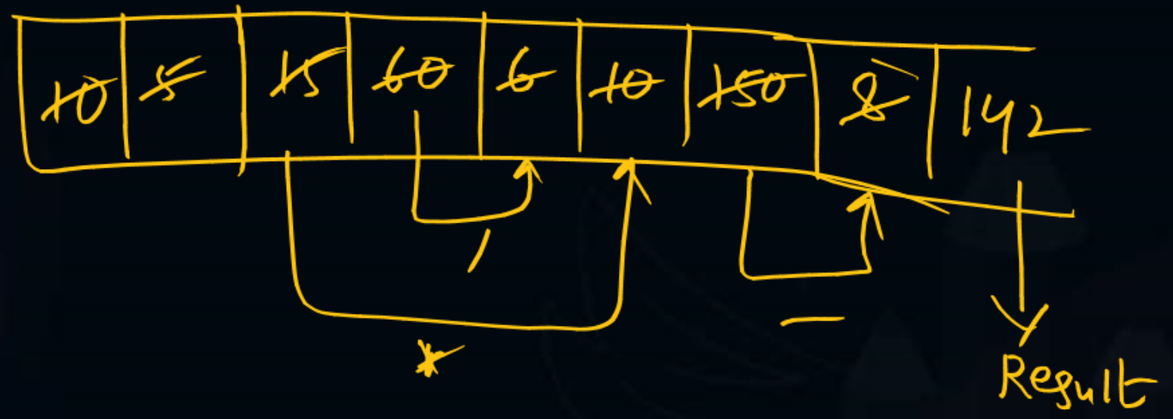
D. - + a × bc ^ ^ def



Topic : Expression Evaluation

The result evaluating the postfix expression 10 5 + 60 6 / * 8 – is

A. 284
B. 213
C. 142
D. 71



2 mins Summary

Expression Evaluation

- Postfix exp
- Prefix exp.

H/W: Evaluate

Postfix Expression : 3 4 * 7 + 3 / 6 * 2 - 4 7 + *

Prefix Expression : - 8 20 * 4 + 2 * 9 / 19 3

THANK - YOU