DAILY ONLINE ACTIVITIES SUMMARY

Date:	**22-06-2020**	**Name:**	**Rakesh M Kotian**
Sem & Sec	**8 th sec-b**	**USN:**	**4al16cs072**

Online Test Summary

Subject	**sms**		
Max. Marks	**30**	**Score**	

Certification Course Summary

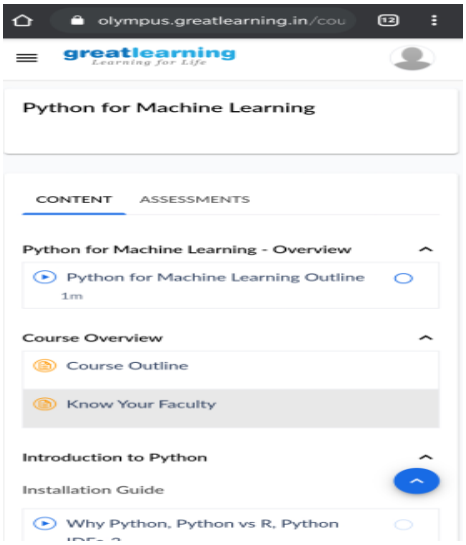
Course	**Python for machine learning**		
Certificate Provider	**Great learning**	**Duration**	**6 hours**

Coding Challenges

Problem Statement: Infix to postfix	
Status:solved	
Uploaded the report in Github	**yes**
If yes Repository name	**Rakeshkotian08**
Uploaded the report in slack	**yes**

Online Test Details: (Attach the snapshot and briefly write the report for the same)

Certification Course Details: (Attach the snapshot and briefly write the report for the same)




```
// CPP Program to convert prefix to Infix
#include <iostream>
#include <stack>
using namespace std;

// function to check if character is operator or not
bool isOperator(char x) {
  switch (x) {
  case '+':
  case '-':
```

```
  case '/':
  case '*':
    return true;
  }
  return false;
}

// Convert prefix to Infix expression
string preToInfix(string pre_exp) {
  stack<string> s;

  // length of expression
  int length = pre_exp.size();

  // reading from right to left
  for (int i = length - 1; i >= 0; i--) {

    // check if symbol is operator
    if (isOperator(pre_exp[i])) {

      // pop two operands from stack
      string op1 = s.top();   s.pop();
      string op2 = s.top();   s.pop();

      // concat the operands and operator
      string temp = "(" + op1 + pre_exp[i] + op2 + ")";

      // Push string temp back to stack
      s.push(temp);
    }

    // if symbol is an operand
    else {

      // push the operand to the stack
      s.push(string(1, pre_exp[i]));
    }
  }

  // Stack now contains the Infix expression
  return s.top();
}

// Driver Code
int main() {
  string pre_exp = "*-A/BC-/AKL";
  cout << "Infix : " << preToInfix(pre_exp);
  return 0;
}
```