Problem: Generation of a code for a given intermediate code

Aim: To convert an Arithmetic expression given in postfix notation to two address assembly language code

Algorithm

Step 1: Declare the necessary variables.

Step 2: Read the expression from the user.

Step 3: Read character by character and do the following until end of the expression is reached.

a) If the character is an alphabet, push it in the identifier stack.

b) Else if the character is an operator, temporarily store the corresponding instruction based upon the operator.

c) Then pop the second identifier from the stack and then the first from the stack and store them in temporary variable, say a, b respectively.

d) Then print the current instruction as"LDA" with the operand stored in variable b.

e) Then print the arithmetic instruction with the operand stored in the variable a.

f) Then print the print the instruction "STA" with a temporary operand.

g) Then push the temporary operand in the stack for future use.

Step 4: Stop the program execution.

Example Output

Enter the Postfix Expression: ABC*+

```
LDA   B
MUL   C
STA   T1
LDA   A
ADD   T1
STA   T2
```