Aim:

Write a program to evaluate a Postfix expression using Stack.

Algorithm:

**Step 1** – START

**Step 2** - declare an array 'stack' of size 100 of integer data type and initialize an integer variable 'top' = -1.

**Step 3** - define functions push(), pop(), checkIfNum(), and calculateExp().

**Step 4** - under the function push(), get a character 'x' from the expression to be added to the Stack; then continue.

**Step 5 -** increment the top by 1 and push the character 'x' into the address pointed by the top.

**Step 6** - under the function pop(), decrement the top by 1 and return the top element from the Stack.

**Step 7** - under the function checkIfNum(), get a character 'y' from the expression to evaluate whether it is numeric; then continue.

**Step 8** -check if y is between '0' and '9'. if True, return True; else, return False.

**Step 9** - under the function calculateExp(), let character 'a', character 'b' and character 'q' be its parameters. Declare a variable 'result' as an integer data type

**Step 10** - Introduce a Switch case statement having the variable 'q' as its condition; then continue.

**Step 11** - if q is equal to '+', set result = a+b; if q is equal to '-', set result = b-a; if q is equal to ‘\*’, set result = a\*b; if q is equal to ‘/’, set result = b/a; then push result to stack.

**Step 12** - declare a pointer variable \*p, array 'post\_exp' of size 100 of character data type, integer variables' n1', 'n2' and 'num'.

**Step 13** – Get valid postfix expression from user and store in array post\_exp. Then set p = post\_exp.

**Step 14** - Introduce a while loop having the condition \*p not equal to '\0'. If the condition is True, continue; else, exit the loop and go to step 18.

**Step 15** - check if \*p is numeric( goto Step 7); If True, set num = \*p – 48 and push num to Stack, then continue.

**Step 16** – Otherwise, remove the first two elements from Stack and store them in variables n1 and n2, respectively; then call calculateExp() function, taking arguments n1, n2 and \*p (go to Step 9).

**Step 17** – increment p by 1, then go to step 14.

**Step 18** – pop the element from the Stack and display it as the result of the expression.

**Step 19** - Stop

Result:

The program was executed successfully and achieved its aim of the program.