Ex. No.:06	Evaluating Arithmetic Expression	Date:04/04/2024

Write a C program to evaluate Arithmetic expressions using stack.

Algorithm:

1) Start
2) Create an empty stack to store operands. 3) Iterate through each character in the expression: a. If the character is a digit, push it onto the stack.
b. If the character is an operator (+, -, *, /), pop two operands from the stack, perform the operation, and push the result back onto the stack.
4) After processing all characters, the final result will be the only element left in the stack.
5) Return this result as the evaluation of the arithmetic expression. 6) Stop

PROGRAM:

```
#include <stdio.h>

#include <string.h>

#include<ctype.h>

int top = -1; int

stack[100]; void

push (int data) {

stack[++top] = data;

} int pop () { int

data; if (top == -

1) return -1; data

= stack[top];

stack[top] = 0;

top--; return

(data);

}

int main()

{

char str[100];

int i, data = -1, operand1, operand2, result;

printf("Enter ur postfix expression:");
```

```
fgets(str, 100, stdin); for (i = 0; i <
strlen(str); i++)
 { if
(isdigit(str[i]))
{
data = (data == -1) ? 0 : data; data
= (data * 10) + (str[i] - 48);
continue;

} if (data != -
1)
{
push(data);
}
if (str[i] == '+' || str[i] == '-'|| str[i] == '*' || str[i] == '/')
{
operand2 = pop(); operand1 = pop();
if (operand1 == -1 || operand2 == -1)
break; switch (str[i])
{
case '+':
result = operand1 + operand2;
push(result); break; case '-':
result = operand1 - operand2;
push(result); break; case '*':
result = operand1 * operand2;
push(result); break; case '/':
result = operand1 / operand2;
push(result); break;
}



}
data = -1;
```

```
} if (top ==

0)

printf("The answer is:%d\n", stack[top]); else

printf("u have given wrong postfix expression\n");

return 0;
}
```

OUTPUT:

RESULT: Thus, the program was successfully executed.