**Evaluate Postfix**

Problem Statement:

Write a C program to evaluate a postfix expression. Note that each operand is 1 digit length.

• Input example :

1. 62/3-42\*+
2. 7223^4-132\*+^\* +4-
3. 6662++3+++++^
4. 60/

• Output example :

1. 8.00
2. 32771.00
3. Invalid Input
4. Unknown

Proposed C Code:

/\* ------- main.c ------- \*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <math.h>

double \*stack;

int top = -1;

int size = 10000;

void init()

{

stack = (double \*)calloc(size, sizeof(double));

}

void push(double item)

{

stack[++top] = item;

}

double pop()

{

return stack[top--];

}

double peek()

{

return stack[top];

}

int isEmpty()

{

if (top == -1)

{

return 1;

}

else

{

return 0;

}

}

double prec(double a, double b, char ch)

{

switch (ch)

{

case '+':

return a + b;

case '-':

return a - b;

case '\*':

return a \* b;

case '/':

return (float)a / b;

case '^':

return pow(a, b);

}

}

int main()

{

init();

char \*s;

int x = 0;

s = (char \*)malloc(10000 \* sizeof(char));

scanf("%[^\n]s", s);

double ans;

for (int i = 0; i < strlen(s); i++)

{

if (s[i] == ' ')

{

continue;

}

if (s[i] >= '0' && s[i] <= '9')

{

push((double)s[i] - '0');

}

else if (s[i] == '+' || s[i] == '-' || s[i] == '\*' || s[i] == '/' || s[i] == '^')

{

if (isEmpty())

{

printf("Invalid Input");

x = 1;

break;

}

double b = pop();

if (isEmpty())

{

printf("Invalid Input");

x = 1;

break;

}

double a = pop();

if (s[i] == '/' && b == 0.0)

{

printf("Unknown");

x = 1;

break;

}

else

{

ans = prec(a, b, s[i]);

}

push(ans);

}

}

if (top > 0)

{

printf("Invalid Input");

x = 1;

}

if (x == 0)

{

printf("%.2lf", ans);

}

return 0;

}

/\* ---------------------- \*/

Conclusion:

The proposed algorithm has a runtime of O(n), where n is the size of the input string.

Limitations and assumptions for this algorithm include:

1. The maximum that can be inserted to the stack is 100000.
2. Here only ‘+’,’-‘,’\*’,’/’,’^’ only these five operators can be evaluated.
3. Only 1 digit operands(between 0 to 9) can be used.