Term Work - 2

Problem Definition :

Consider a calculator that needs to perform checking correctness of paranthesized arithmetic & convertid the same postfix expression for evaluation. Develop & execute a program in C using suitable DS to perform the same & print both expressions. The input expression consists of a single character operands & binary operators.

Aim:

Aim of this TW is to learn the implementation of Stacks in solving problems.

Theory:

* Stacks : stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO or FIFO

* Mainly following basic operations are performed in Stack
  - → Push : Adds an item in stack. If stack is full then it is said to overflow
  - → Pop : Removes an item from stack. If stack is empty, then stack is said to underflow
  - → Peek : Returns top element of stack.
  - → isEmpty : Returns true if the stack is empty

Program:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

struct Stack {
    int top, capacity;
    int *a;
}
int isEmpty (struct Stack *s) {
    return s->top == -1
}
char peek (struct Stack *s) {
    return s->top s->a[s->top];
}
char pop (struct Stack *s) {
    if (! isEmpty (s))
        return s->a[s->top--];
    return '$';
}
void push (struct Stack *s, char op) {
    s->a[++s->top];
}
int isOperand (char ch) {
    return (ch >='a' && ch<='z') || (ch >='A' && ch <='Z');
}
```

```
int Prec (char ch) {
   switch (ch) {
    case '+':
    case '-':  return 1;
    case '*':
    case '/' : return 2;
    case '^' : return 3;
   }
  return -1;
}
int infixToPostfix (char *exp) {
  struct Stack *s = (struct Stack*)malloc (sizeof (struct Stack));
  s→top = -1 ;  int i, k=0 ;
  s→capacity = strlen (exp);
  s→a = (int*) malloc (s→capacity * sizeof (int));
  if (!s)
    return -1;
  for (i=0, k=-1 ; i<l; ++i) {
    printf("\n\n")  if (isOperand (exp[i])
                        exp [++k] = exp[i];
    else if (exp[i] == '(')
        push (s, exp[i]);
    else if (exp[i] == ')') {
        while (!isEmpty (s) && peek(s) != '(')
              exp [++k] = pop(s)
        if (!isEmpty (s) && peek(s) != '(')
            return -1;
        else
            pop (s);
   }
```

```
    else {
        while (!isEmpty(s) && Prec(exp[i]) <= Prec(peek(s)))
            exp[++k] = pop(s);
        push(s, exp[i]);
    }
}
while (!isEmpty(s))
    exp[++k] = pop(s);
exp[++k] = '\0';
    printf("\n %s \n", exp);
}
int main() {
    char *exp;
    exp = (char*)malloc(1000*sizeof(char));
    printf("\n Enter an expression : ");
    scanf("%s", exp);
    infixToPostfix(exp);
    return 0;
}

// Written & executed in VS Code, Ubuntu 20.04.1 env.
```

References

Books:

* Richard F Gilberg, Behrouz A Fourouzan, Data Structures: A Pseudo Code Approach with C, Cengage 2007.

* Horowitz, Sahni, Anderson-Freed, Fundamentals of Data Structures in C, Universe Press 2nd Edition.

E-Resources:

* https://geeksforgeeks.org/

Conclusion.

In this TW, I learnt about stacks, basic operations of stacks & their implementation to solve problems. We also learned basic problem solving techniques & programming paradigms.