ANVITA KUMAR
Roll No.: 2104097

```
//WAP to implement infix to postfix conversion using stack ADT

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdbool.h>
typedef struct Stack
{
int top;
unsigned capacity;
char* array;
}Stack;
Stack* stack = NULL;
Stack* createStack(unsigned capacity)
{
stack = malloc(sizeof(Stack)); // (Stack*)
if (!stack)
return NULL;
stack->top = -1;
stack->capacity = capacity;
stack->array = /*(int*)*/ malloc(capacity*sizeof(int));
return stack;
}
int isEmpty()
{
return stack->top == -1 ;
}
char peek()
{
return stack->array[stack->top];
```

```
}

char pop()

{

if (!isEmpty())

return stack->array[stack->top--] ;

}

void push(char op)

{

stack->array[++stack->top] = op;

}

int isOperand(char ch)

{

return (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z') || (ch >= '0' && ch <= '9');

}

int Prec(char ch)

{

switch (ch) {

case '+':

case '-':

return 1;

case '*':

case '/':

return 2;

case '^':

return 3;

}

return -1;

}

int infixToPostfix(char* exp)

{
```

ANVITA KUMAR
Roll No.: 2104097

```
int i, k;

Stack* stack = createStack(strlen(exp));

if(!stack)

return -1 ;

printf("Token\t\tStack\t\tPostfix String\n");

for (i = 0, k = -1; exp[i]; ++i) {

if (isOperand(exp[i]))

exp[++k] = exp[i];

else if (exp[i] == '(')

push(exp[i]);

else if (exp[i] == ')') {

while (peek() != '(')

exp[++k] = pop();

pop();

}

else {

while (!isEmpty() && Prec(exp[i]) <= Prec(peek()) && exp[i] != '^')

exp[++k] = pop();

push(exp[i]);

}

printf("%c", exp[i]);

if(stack->top == -1)

printf("%16c");

else

printf("%16c", stack->array[0]);

for (int i = 1; i <= stack->top; i++) {

printf("%c", stack->array[i]);

}

if (exp[0] != '(')

printf("%*c", 16-stack->top, exp[0]);
```

ANVITA KUMAR
Roll No.: 2104097

```
for (int i = 1; i <= k; i++) {

printf("%c", exp[i]);

}

printf("\n");

}

while (!isEmpty())

exp[++k] = pop();

exp[++k] = '\0';

printf( "%37s", exp );

}

int main()

{

char exp[15];

printf("Enter the infix expression: ");

scanf("%s", exp);

printf("\n");

infixToPostfix(exp);

return 0;

}
```