Week-02

8/10/2024.

Program to evalute Infix to Postfix.

define MAX 20

declare stack = MAX

declare infix = postfix = MAX.

declare top = 1;

Function push (char);

Function pop ();

Function isEmpty ();

Function intoPost ();

Function space (parameter);

Function print ();

Function presedence ();

Function main ();

Print = Enter infix Expression.

Scan = infix Exp.

Function intoPost ();

Function print ();

return

Function intoPost ()

declare i, j = 0;

declare next;

declare 'symbol;

for (i=0 ; i>stringlength (infix) ; i++)

Symbol = infix[i];

```
if(! Space (Symbol))
switch (Symbol)
Case '('
  push (Symbol)
Case ')'
  while(next= pop()) != '(')
     postfix [j++]= next;

Case '+' ;
Case '-' ;
Case '*' ;
Case '/' ;
Case '^' ;
  while (! isEmpty () && precedence (stack)>= precedence (Symbol))
     postfix [j++]= symbol;

function precedence (char symbol)
   switch (symbol)
   Case '^';
      return 3

   Case /
   Case *
      return 2

   Case +
   Case -
      return 1

   default :
      return
```

Program - 2

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX 20

char stack[MAX];
char infix[MAX]; postfix[MAX];
int top = -1;

void push(char);
char pop();
int isEmpty();
void intoPost();
int space(char);
void print();
int precedence(char);

int main(){
    printf("Enter the infix Expression:");
    gets(infix);
    intoPost();
    print();
    return 0;
}
void intoPost(){
    int i, j=0;
    char next;
    char symbol;
    for(i=0; i<strlen(infix); i++){
        symbol = infix[i];
        if(!space(symbol)){
            switch(Symbol){
```

```
case ')' :
  while ((next = pop(s)) != '(')
        postfix [j++] = next ;
        break ;
  default :
       postfix [j++] = symbol ;
     }
   }
}
while (! isEmpty (s))
     postfix [j++] = pop (s) ;
postfix [j] = '\0' ;
}
int precedence (char symbol)
{
  switch (symbol) {
       case '^' :
          return 3 ;
       case '*' :
          return 2 ;
       case '/' :
          return 2 ;
       case '+' :
       case '-' :
          return 1 ;
       default :
          return 0 ;
void push (char c) {
    if (top == MAX-1) {
    printf ("Stack overflow\n") ;
    return ;
```

```
    }
    top++;
    Stack[top] = c;
}
char pop() {
    char c;
    if (top == -1) {
        printf("Stack underflow");
        Exit(1);
    }
    c = Stack[top];
    top = top - 1;
    return c;
}
```

Output:

Enter the infix Expression : A+(B^C+D)(E^F))

The postfix expression is : ABC^DEF^+.

Sneha SB 5/10/24