DOMS | Page No.
Date 1 / 01 / 2024

Lab: 2

Write a program to convert a given valid parenthesized infix arithmetic expression to postfix expression.

```
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
#include <string.h>
#include <stdlib.h>
#define MAX 100
char st[MAX];
int top = -1;

void push (char st[], char);
char pop (char st[]);
void InfixtoPostfix (char source[], char targ
                    -et[]);
int getPriority(char);

int main(){
    char infix[100], postfix[100];
    printf("\n Enter any infix expression:")
    gets(infix);
    strcpy(postfix, "");
    InfixtoPostfix(infix, postfix);
    printf("\n The corresponding postfix
        expression is :");
    puts(postfix);
```

```
    getch();
    return 0;
}

void InfixtoPostfix(char source[], char target
{
    int i=0, j=0;
    char temp;
    strcpy(target, "");
    while(source[i]!='\0'){
        if(source[i]=='('){
            push(st, source[i]);
            i++;
        }
    else if(source[i]==')'){
        while((top!=-1)&&(st[top]!='('))
        {
            target[j]=pop(st);
            j++;
        }
        if(top==-1){
            printf("\n INCORRECT EXPRESS
                    -ON");
            exit(1);
        }
        temp=pop(st);
        i++;
    }
```

```
else if( isdigit(source[i]) || isalpha(sour-
                                    - ce[i])) {
    target[j] = source[i];
      j++;
      i++;
  }
else if (source[i] == '+' || source[i] == '-' ||
  source[i] == "*" || source[i] == "/" || source
    [i] == "%") {
    while ((top != -1) && (st[top] != '(')
      && (getPriority(st[top]) > getPriori-
          - ty(source[i]))){
        target[j] = pop(st);
        j++;
      }
    push(st, source[i]);
      i++;
  }
else {
    printf("\n Incorrect element in
              expression");
      exit(1);
    }
  }
while ((top != -1) && (st[top] != '(')) {
    target[j] = pop(st);
      j++;
  }
target[j] = '\0';
}
```

```
int getPriority (char op)
{
  if(op=='/' || op=='*' || op=='%')
    return 1;
  else if (op=='+' || op=='-')
    return 0;
}

void push (char st[], char val) {
  if (top == MAX-1)
     printf("\n STACK OVERFLOW");
  else{
       top++;
      st[top] = val;
    }
}

char pop(char st[]) {
  char val=' ';
  if (top==-1)
    printf("\n STACK UNDERFLOW");
  else {
    val = st[top];
    top--;
   }
  return val;
}
```

Output :

Enter any infix expression : AB+C/D*E

The corresponding postfix expression is :

ABCDE*/+

Enter any infix expression : (A+B)-S*k

The corresponding postfix expression is :

AB+Sk*-

Enter any infix expression : (AB*CD){A+B}

Incorrect element in expression.

P

4/1/24

DOMS | Page No.
Date 1 / 1 / 24

WAP to simulate the working of a queue of integers using an array. Provide the following operations.
a) Insert b) Delete c) Display
The program should print appropriate messages for queue empty and queue overflow conditions.

```
#include <stdio.h>
#define MAX 5

void insert();
void delete();
void display();
int queue_array[MAX];
int rear = -1;
int front = -1;

main(){
    int choice;
    while(1){
        printf("1. Insert element to queue\n");
        printf("2. Delete element from queue\n");
        printf("3. Display all elements of queue\n");
        printf("4. Quit\n");
        printf("Enter your choice: ");
```

```
scanf("%d", &choice);
switch(choice){
  case 1: insert();
          break;
  case 2: delete();
          break;
  case 3: display();
          break;
  case 4: exit(1);
  default: printf("Wrong choice\n");
  }
 }
}
void insert(){
  int add_item;
  if (rear == Max-1){
      printf("Queue Overflow\n"); }
  else{
      if (front == -1){
          front = 0;}
      printf("Insert the element in queue
                  :");
      scanf("%d", &add_item);
       rear = rear+1;
      queue_array[rear] = add_item;
    }
}
```

```
void delete(){
    if (front == -1 || front > rear){
        printf("Queue Underflow \n");
    }
    else{
        printf("Element deleted from queue is
                %d \n", queue_array[front]
        front = front+1;
    }
}

void display(){
    int i;
    if (front == -1){
        printf("Queue is empty \n");
    else{
        printf("Queue is : \n");
        for(i= front; i<= rear ; i++){
            printf("%d", queue_array[i]);
        }
        printf("\n");
    }
}
```

Output:

1. Insert element to queue
2. Delete element from queue
3. Display all elements of queue
4. Quit

Enter your choice : 1
Insert the element in queue : 96
1. Insert element to queue
2. Delete element from queue
3. Display all elements of queue
4. Quit
Enter your choice : 1
Insert the element in queue : 56
1. Insert element to queue
2. Delete element from queue
3. Display all elements of queue
4. Quit
Enter your choice : 3
Queue is :
96 56
1. Insert ~~ye~~ element to queue
2. Delete element from queue
3. Display all elements of queue
4. Quit
Enter your choice : 2
Element deleted from queue is : 96
1. Insert element to queue
2. Delete element from queue
3. Display all elements of queue
4. Quit
Enter your choice : 3
Queue is :
56

9/11/24

1. Insert element to queue
2. Delete element from queue
3. Display all elements of queue
4. Quit

Enter your choice: 4

WAP to simulate the working of a circular queue of integers using an array. Provide the following operations.
a) Insert b) Delete c) Display
The program should print appropriate message for queue empty and queue overflow conditions.

```
#include<stdio.h>
#define MAX 5
int cqueue_arr[MAX];
int front=-1;
int rear=-1;

void insert(int item){
   if((front==0 && rear==MAX-1)||(front==
      rear+1)){
      printf("Queue Overflow\n");
   }
  if (front==-1)
   {
     front=0;
     rear=0;
   }
  else{
    if(rear=MAX-1)
       rear=0;
    else
        rear=rear+1;
  }
```

```
    cqueue_arr[rear] = item;
}

void deletion()
{
  if(front == -1)
  {
    print("Queue Underflow");
  }
  printf("Element deleted from queue is:
    %d \n", cqueue_arr[front]);
  if(front == rear)
  {
    front = -1;
    rear = -1;
  }
  else
    front = front + 1;
}
}

void display()
{
  int front_pos = front, rear_pos = rear;
  if(front == -1)
  {
    printf("Queue is empty \n");
  }
```

```
printf("Queue elements : \n");
if (front_pos <= rear_pos){
 while (front_pos <= rear_pos)
   {
    printf("%d", cqueue_arr [front_pos]);
    front_pos++;
   }
 }
else {
   while (front_pos <= MAX-1)
    {
     printf("%d", cqueue_arr[front_pos]);
     front_pos++;
     }
    front_pos = 0;
   while (front_pos <= rear_pos)
    {
     printf("%d", cqueue_arr [front_pos]);
     front_pos++;
     }
   }
   printf("\n");
}

int main(){
  int choice, item;
  do {
    printf("1. Insert \n");
```

```
printf("2. Deletion \n");
printf("3. Display \n");
printf("4. Quit \n");
printf("Enter your choice: ");
scanf("%d", &choice);

switch(choice)
 {
 case 1:
   printf("Input the element for inse
    -tion in queue: ");
   scanf("%d", &item);
   insert(item);
   break;
 case 2:
    deletion();
    break;
 case 3:
    display();
    break;
 case 4:
       break;
 default:
    printf("Wrong choice \n");
  }
}while(choice != 4);
return 0;
}
```

Output:

```
1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 1
Input the element for insertion in queue: 96
1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 1
Input the element for insertion in queue: 56
1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 3
Queue elements:
96 56
1. Insert
2. Delete
3. Display
4. Quit
Enter your choice: 2
Element deleted from queue is : 96
1. Insert
2. Delete
```

3. Display
4. Quit
Enter your choice: 4

7/1/24