JAGirKar

JUNAID. GIRKAR
6000419005 7

20.10.2020

DATA STRUCTURES TT2

A-3

Q1 Write a program to convert an infix expression to postfix expression

ANS

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
char st[100];
int top = -1;
void push (char val)
{
    if (top == 99)
    {
        printf ("stack overflow");
    }
    else
    {
        top ++;
        st [top] = val;
    }
}
char pop()
{
    if (top == -1)
    {
        return '';
    }
    else
```

```
        {
            return  st[top--];
        }
}
int priority (char val)
{
    if (val == '*' || val == '/' || val == '%')
    {
        return (2);
    }
    else if (val == '+' || val == '-')
    {
        return (1);
    }
    else
    {
        return (0);
    }
}
void main()
{
    char infix[100], postfix[100], temp;
    int i=0, j=0, flag=1;
    printf("Enter an infix expression : \n");
    gets(infix);
    while (infix[i] != '\0')
    {
        if (infix[i] == 'c')
        {
            push(infix[i]);
        }
```

```
else if (isalnum (infix [i]))
{
    postfix [j] = infix [i] ;
    j++ ;
}
else if (infix [i] == ')')
{
    while (top != -1 && st [top] != 'c')
    {
        postfix [j] = pop ();
        j++ ;
    }
    if (top == -1)
    {
        printf ("\n Invalid Expression");
        flag = 0 ;
    }
    temp = pop();
}
else if (infix [i] == '+' || infix [i] == '-' || infix[i] == '*'
|| infix [i] == '/' || infix [i] == '%')
{
    while (top != -1 && st [top] != 'c' && (priority (st [top])
    >= priority (infix [i])))
    {
        postfix [j] = pop ();
        j++ ;
    }
    push (infix [i]);
}
```

```
        else
        {
            printf("\n Incorrect element");
            flag = 0;
        }
        i++;
    }
    while (top != -1)
    {
        postfix[j] = pop();
        j++;
    }
    postfix[j] = '\0';
    j=0
    while (postfix[] != '\0')
    {
        if (postfix[j] == 'c')
        {
            printf("Invalid Expression");
            flag = 0;
        }
        j++;
    }
    if (flag == 1)
    {
        printf("\n Postfix Expression :");
        puts (postfix);
    }
}
```

Q2

ANS

```
#include <stdio.h>
#include <stdlib.h>
Struct list {
    int data;
    Struct list *left;
    Struct list *right;
} typedef list;

void add (list **root, int val) {
    if (*root == NULL) {
        list *block = (list*) malloc (sizeof (list));
        block -> data = val;
        block -> left = NULL;
        block -> right = NULL;
        *root = block;
    } else {
        if ((*root) -> data > val) {
            add (&(*root) -> left, val);
        } else {
            add (&(*root) -> right, val);
        }
    }
}
```

```
void main(){
    list *root = NULL;
    add(&root, 50);
    add(&root, 75);
    add(&root, 60);
    add(&root, 25);
    add(&root, 17);
    add(&root, 20);
    add(&root, 45);
    add(&root, 30);
```

Q3

ANS

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 100
int queue [MAX];
int front = -1, rear = -1;
void insert (void);
int delete_element (void);
int peek ();
void display ();

int main()
{
    int option = 0, val;
    while (option != -1)
    {
        printf("\n Enter your option");
        scanf ("%d", &option);
        switch (option)
        {
            case 1: insert();
            break;
            case 2: val = delete_element ();
            if (val != -1)
            {
                printf ("\n Element deleted is %d", val);
            }
            break;
```

```
            case 3 : val = peek ();
            if (val != -1)
                printf ("Element is %d", val);
            break;

            case 4 : display ();
            break;
        }
    }
    return 0;
}
void insert ()
{
    int num;
    printf ("\n Enter the value to be insexted ");
    scanf ("%d", &num);
    if (rear == MAX-1 && front == 0)
    {
        printf ("\n Underflow error");
    }
    else if (rear == MAX-1 && front != 0)
    {
        rear = 0;
    }
    else if (rear == -1 && front == -1)
    {
        front = rear = 0;
    }
```

```
        else
        {
            rear ++ ;
        }
        queue [rear ] = num ;
}
int delete_element()
{
    int val ;
    if (front == -1)
    {
        printf ("\n Underflow error ");
        return -1 ;
    }
    else if (front == MAX -1)
    {
        val = queue [front] ;
        front = 0 ;
        return val ;
    }
    else if ( front == rear)
    {
        val = queue [front ] ;
        front = rear = -1 ;
        return val ;
    }
    else
    {
        val = queue [front] ;
        front ++ ;
```

```
        return val ;
    }
}   else
```

```
int peek ()
{
    if (front == -1)
    {
        printf ("\n Underflow Error");
        return -1 ;
    }
    else
    {
        return queue [front];
    }
}
void display ()
{
    int i;
    printf ("\n Elements in Queue are \n");
    for (i=front ; i<= rear ; i++)
    {
        printf (" %3d ", queue [i]);
    }
}
```