College C codes > DSA-ASS-7 > C 7_1_postix_eval.c > main()

```c
/* Note: Input should be taken from the user
Write a C/C++ code to evaluate the Postfix Expression using stack.
Example :--
Input : 34*25*+
Output : 22          */

#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <stdlib.h>
#include <math.h>

#define MAX 100

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

void Push(int x){
    if(top > MAX-1){
        printf("ERROR : Stack overflow.\n");
        return;
    }
    stack[++top] = x;
}

int Pop(){
    if(top == -1){
        printf("ERROR : Stack underflow.\n");
        return -9999;
    }
    return stack[top--];
}

int IsEmpty(){
    if(top == -1){
        return 1;
    }
    else{
        return 0;
    }
}
```

```
42
43  float calculate(char *postfix){
44      int len = strlen(postfix);
45      int i = 0, temp, res;
46      while(i < len){
47          res=0, temp=0;
48          if(postfix[i] == '('){
49              int dig = 0;
50              while(postfix[++i] != ')'){
51                  dig = dig*10 + ((int)(postfix[i]) - 48);
52              }
53              Push(dig);
54          }
55          else if(isdigit(postfix[i])){
56              Push(((int)(postfix[i]) - 48));
57          }
58          else{
59              switch(postfix[i]){
60                  case '+' : res += Pop() + Pop();
61                             Push(res);
62                             break;
63                  case '*' : res += Pop() * Pop();
64                             Push(res);
65                             break;
66                  case '-' : temp = Pop();
67                             res += Pop() - temp;
68                             Push(res);
69                             break;
70                  case '/' : temp = Pop();
71                             res += Pop() / temp;
72                             Push(res);
73                             break;
74                  case '%' : temp = Pop();
75                             res += Pop() % temp;
76                             Push(res);
77                             break;
78                  case '^' : temp = Pop();
79                             res += pow(Pop(),temp);
80                             Push(res);
81                             break;
82              }
83          }
84          i++;
85      }
86      return stack[top];
87  }
```

```
88
89  int main(){
90      char postfix[MAX];
91      printf("Enter Postfix Expression : ");
92      scanf("%s",postfix);
93      printf("Result = %f",calculate(postfix));
94      return 0;
95  }
```

TERMINAL COMMENTS

```
Enter Postfix Expression : (15)711+-/3*211++-
Result = 5.000000
```

College C codes > DSA-ASS-7 > C 7_2_ll_queue.c > IsEmpty()

```c
/* Implement a queue using a linked list with two pointers: front and rear.
Elements can only be inserted via the rear pointer and deleted via the front pointer.
The queue has the following basic operations :--

• enqueue() - Insert an element in the queue.
• dequeue() - Remove the element from the queue.
• peek() - Return the element at the front node of the queue
• lenqueue() - Return the length of the queue.
• isempty() - Checks if the queue is empty. */

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

struct queue{
    int data;
    struct queue *next, *prev;
};

struct queue *createnode(int x){
    struct queue *newnode = (struct queue *)malloc(sizeof(struct queue));
    newnode->data = x;
    newnode-> next =NULL;
    newnode->prev =NULL;
    return newnode;
}

struct queue *front = NULL;
struct queue *rear = NULL;
int c = 0; // no. of queue elements

int isempty(){
    if(front == NULL){
        return 1;
    }
    else{
        return 0;
    }
}
```

```c
void enqueue(int x){
    if(c == 0){
        front = createnode(x);
        rear = front;
        c++;
        return;
    }
    struct queue *newnode = createnode(x);
    newnode->next = front;
    front->prev = newnode;
    front = newnode;
    c++;
}

void dequeue(){
    if(isempty()){
        printf("ERROR : Queue Underflow.");
        return;
    }
    struct queue *temp = rear;
    rear = rear->prev;
    rear->next = NULL;
    free(temp);
    c--;
}

int peek(){
    if(isempty()){
        printf("ERROR : Queue Underflow.");
        return -1;
    }
    else{
        return front->data;
    }
}

int lenqueue(){
    return c;
}
```

```c
void display(){
    struct queue *temp = front;
    printf("Queue : ");
    while(temp != NULL){
        printf("%d  ",temp->data);
        temp = temp->next;
    }
    printf("\n");
}

int main(){
    int ch;
    while(1){
        printf("\nEnter Your Choice :--\n1 - Insert an element in the queue\n2 - Remove Element from the queue\n3 - Display the element at the front of the queue\n4 - Check the length
        of queue\n5 - Checks if the queue is empty\n6 - Display the queue\n0 - Exit.\nChoice : ");
        scanf("%d",&ch);
        switch(ch){
            case 1 : int n;
                     printf("Enter the number you want to insert : ");
                     scanf("%d",&n);
                     enqueue(n);
                     break;
            case 2 : dequeue();
                     break;
            case 3 : printf("Element in the front of the queue : %d\n",peek());
                     break;
            case 4 : printf("Length of queue : %d\n",lenqueue());
                     break;
            case 5 : if(isempty()){
                        printf("Queue is Empty\n");
                     }
                     else{
                        printf("Queue is NOT empty\n");
                     }
                     break;
            case 6 : display();
                     break;
            case 0 : exit(0);
                     break;
            default : printf("Wrong Input! Please Try Again.\n");
        }
    }
    return 0;
}
```

```
Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 1
Enter the number you want to insert : 2

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 6
Queue : 2

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 1
Enter the number you want to insert : 8

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 1
Enter the number you want to insert : 15
```

```
Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 6
Queue : 15  8  2

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 2

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 6
Queue : 15  8

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 3
Element in the front of the queue : 15
```

```
Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 4
Length of queue : 2

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 5
Queue is NOT empty

Enter Your Choice :--
1 - Insert an element in the queue
2 - Remove Element from the queue
3 - Display the element at the front of the queue
4 - Check the length of queue
5 - Checks if the queue is empty
6 - Display the queue
0 - Exit.
Choice : 0
PS C:\Users\shuvr\OneDrive\Documents\CODING>
```