VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

LAB REPORT

CSE2011 – DATA STRUCTURES AND ALGORITHMS LAB

(B.Tech. CSE Specialisation in Bioinformatics)

WINTER SEMESTER 2020-2021

Name:	**ALOK MATHUR**
Reg. No:	**20BCB0086**
Slot:	**L51+L52**
Faculty Name:	**SRIVANI A Ma'am**

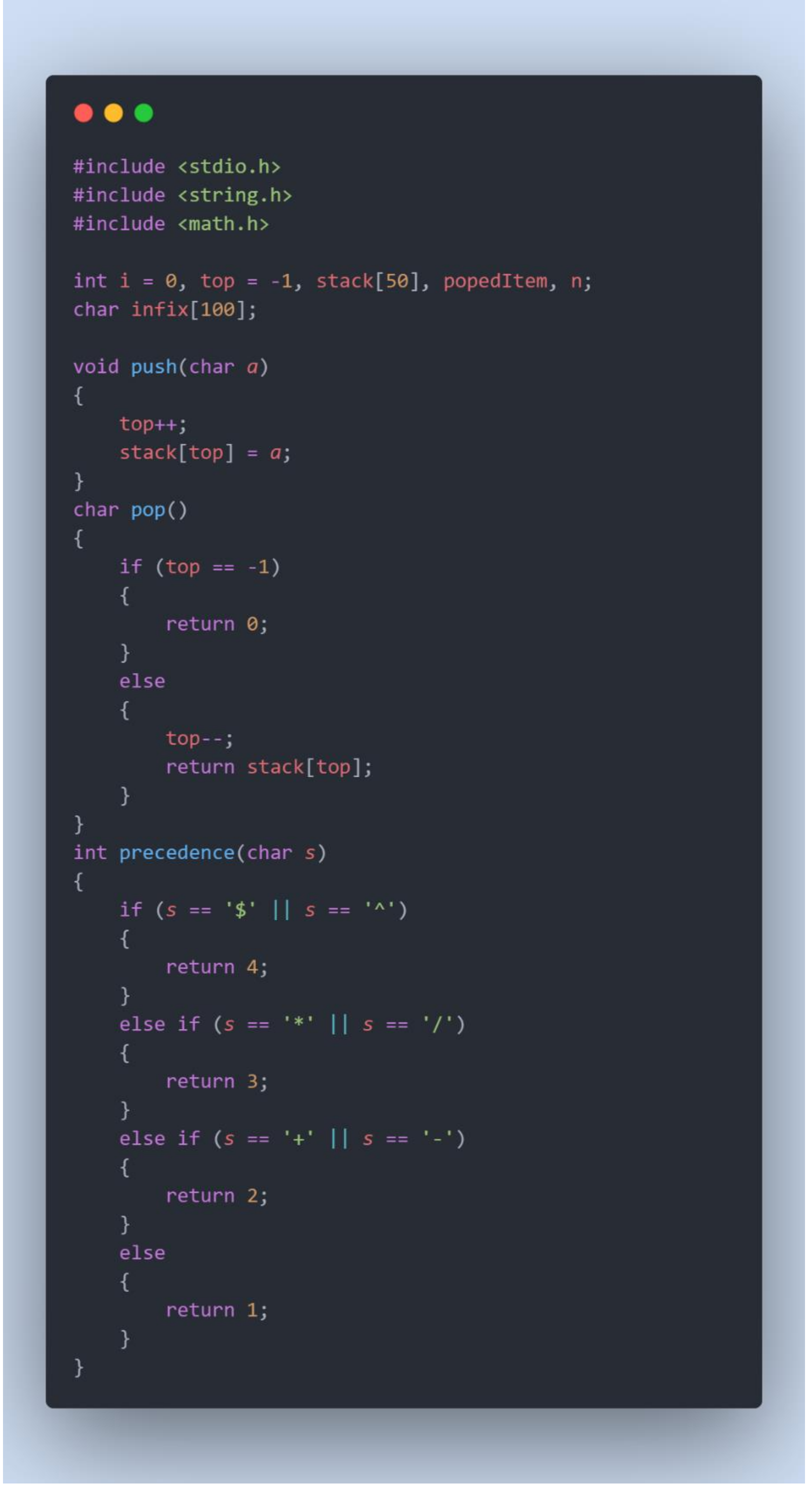
VIT – A Place to Learn; A Chance to Grow

QUESTIONS, CODE && OUTPUT

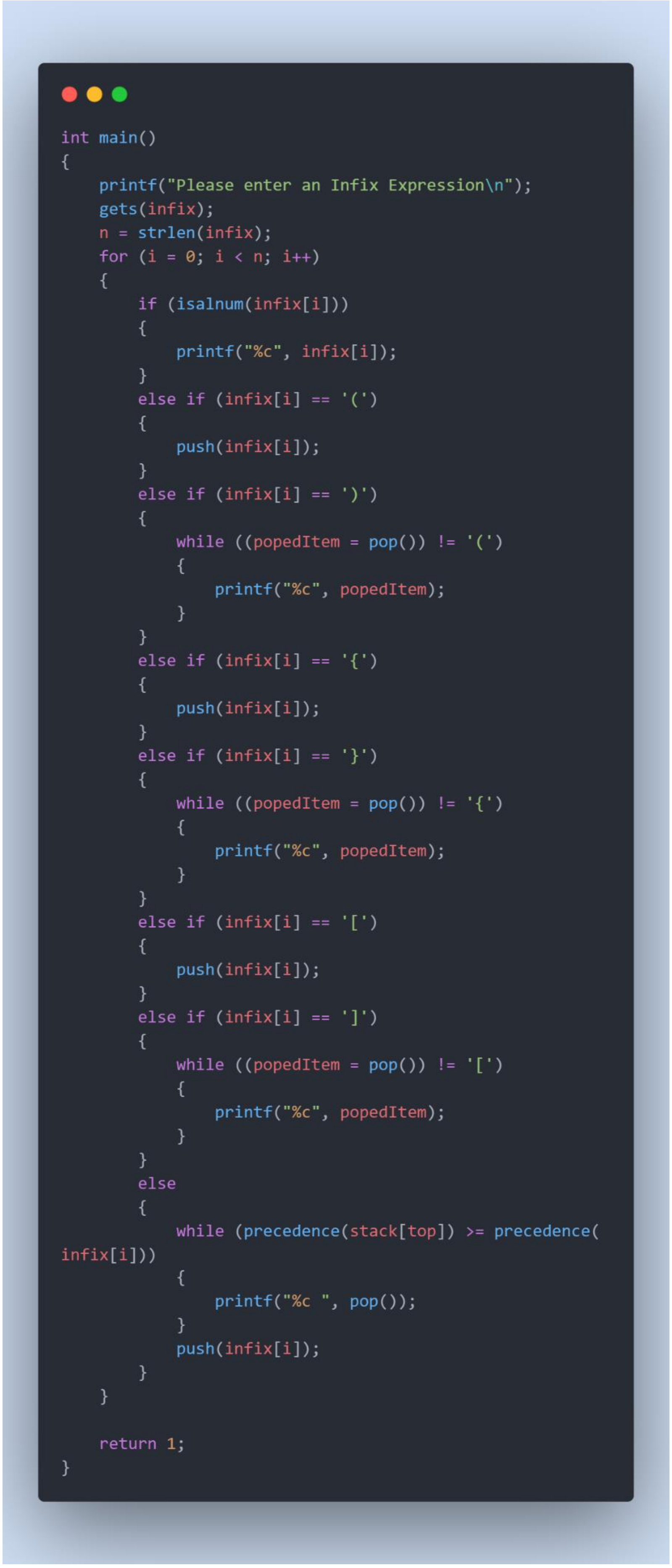
1. Stack applications

i) Infix to Postfix Conversion

CODE

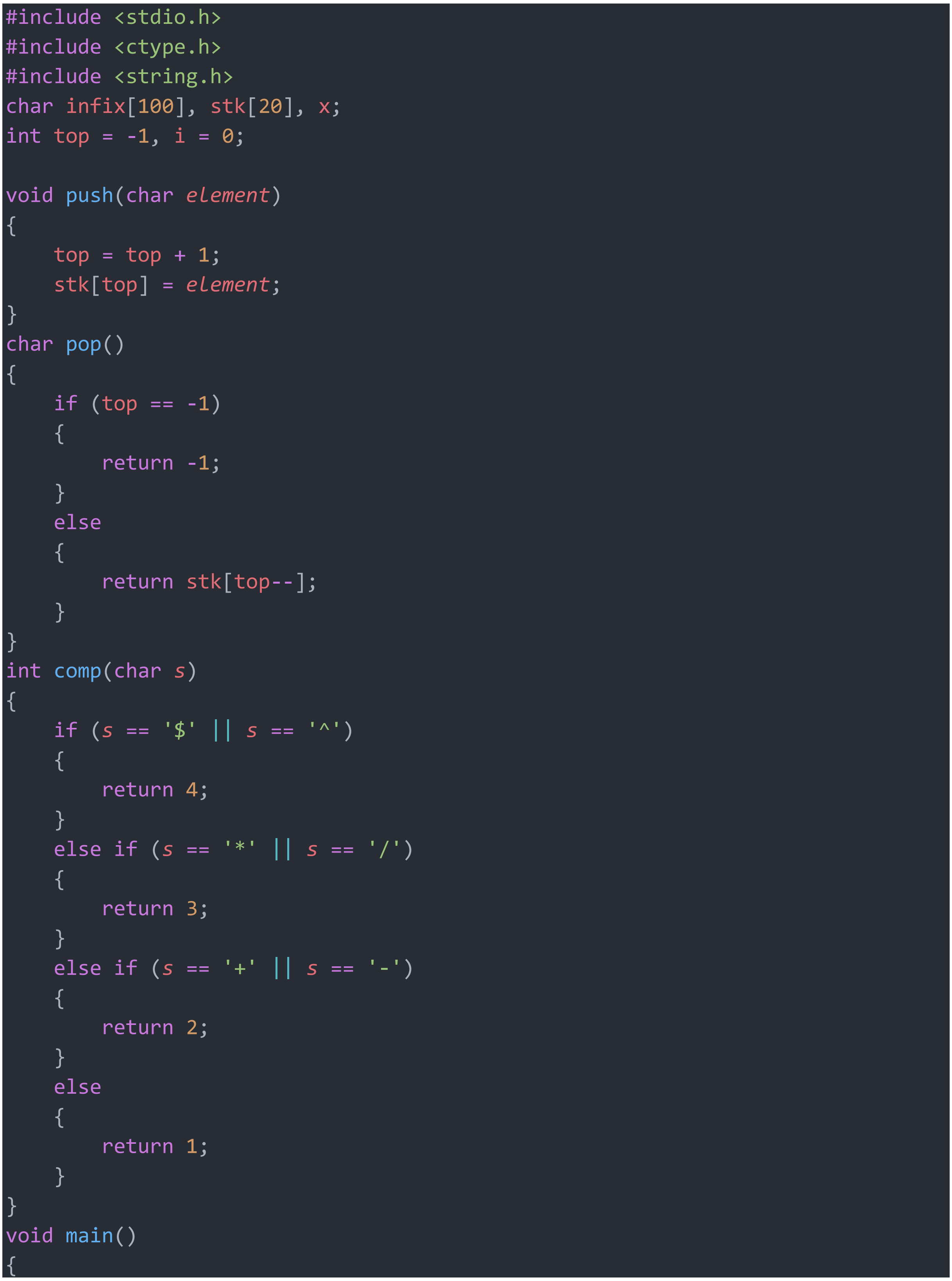


```c
#include <stdio.h>
#include <string.h>
#include <math.h>

int i = 0, top = -1, stack[50], popedItem, n;
char infix[100];

void push(char a)
{
    top++;
    stack[top] = a;
}
char pop()
{
    if (top == -1)
    {
        return 0;
    }
    else
    {
        top--;
        return stack[top];
    }
}
int precedence(char s)
{
    if (s == '$' || s == '^')
    {
        return 4;
    }
    else if (s == '*' || s == '/')
    {
        return 3;
    }
    else if (s == '+' || s == '-')
    {
        return 2;
    }
    else
    {
        return 1;
    }
}
```

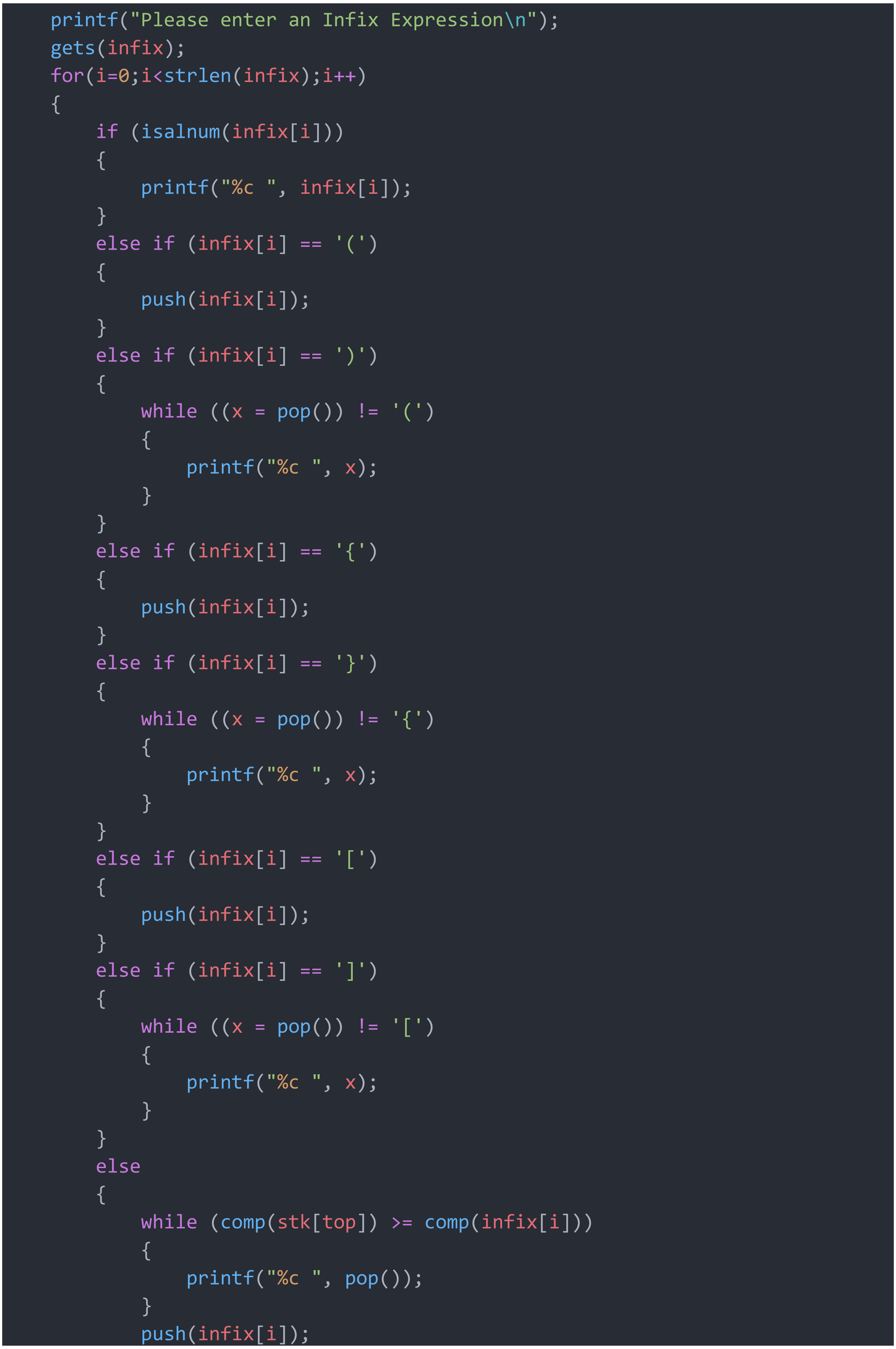


```c
int main()
{
    printf("Please enter an Infix Expression\n");
    gets(infix);
    n = strlen(infix);
    for (i = 0; i < n; i++)
    {
        if (isalnum(infix[i]))
        {
            printf("%c", infix[i]);
        }
        else if (infix[i] == '(')
        {
            push(infix[i]);
        }
        else if (infix[i] == ')')
        {
            while ((popedItem = pop()) != '(')
            {
                printf("%c", popedItem);
            }
        }
        else if (infix[i] == '{')
        {
            push(infix[i]);
        }
        else if (infix[i] == '}')
        {
            while ((popedItem = pop()) != '{')
            {
                printf("%c", popedItem);
            }
        }
        else if (infix[i] == '[')
        {
            push(infix[i]);
        }
        else if (infix[i] == ']')
        {
            while ((popedItem = pop()) != '[')
            {
                printf("%c", popedItem);
            }
        }
        else
        {
            while (precedence(stack[top]) >= precedence(
infix[i]))
            {
                printf("%c ", pop());
            }
            push(infix[i]);
        }
    }

    return 1;
}
```

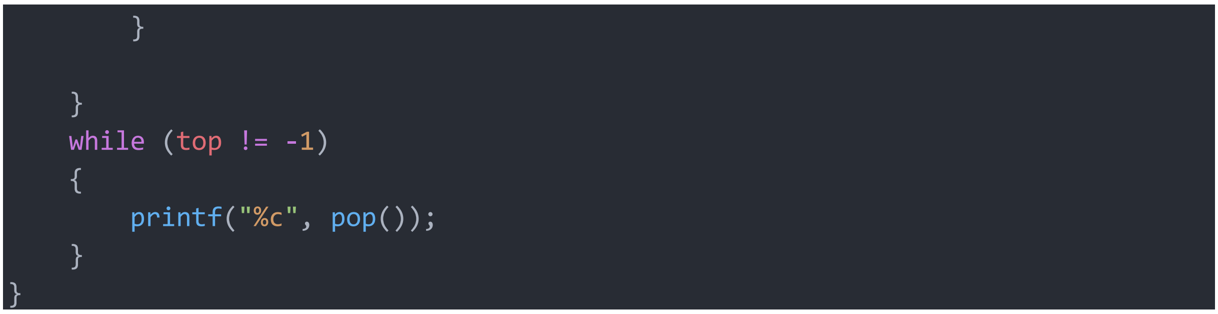
Complete Code



```c
#include <stdio.h>
#include <ctype.h>
#include <string.h>
char infix[100], stk[20], x;
int top = -1, i = 0;

void push(char element)
{
    top = top + 1;
    stk[top] = element;
}
char pop()
{
    if (top == -1)
    {
        return -1;
    }
    else
    {
        return stk[top--];
    }
}
int comp(char s)
{
    if (s == '$' || s == '^')
    {
        return 4;
    }
    else if (s == '*' || s == '/')
    {
        return 3;
    }
    else if (s == '+' || s == '-')
    {
        return 2;
    }
    else
    {
        return 1;
    }
}
void main()
{
```

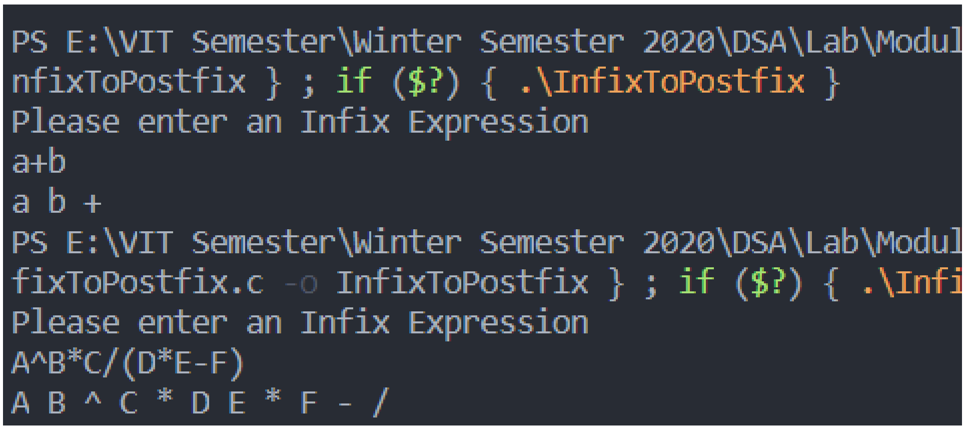


```
    printf("Please enter an Infix Expression\n");
    gets(infix);
    for(i=0;i<strlen(infix);i++)
    {
        if (isalnum(infix[i]))
        {
            printf("%c ", infix[i]);
        }
        else if (infix[i] == '(')
        {
            push(infix[i]);
        }
        else if (infix[i] == ')')
        {
            while ((x = pop()) != '(')
            {
                printf("%c ", x);
            }
        }
        else if (infix[i] == '{')
        {
            push(infix[i]);
        }
        else if (infix[i] == '}')
        {
            while ((x = pop()) != '{')
            {
                printf("%c ", x);
            }
        }
        else if (infix[i] == '[')
        {
            push(infix[i]);
        }
        else if (infix[i] == ']')
        {
            while ((x = pop()) != '[')
            {
                printf("%c ", x);
            }
        }
        else
        {
            while (comp(stk[top]) >= comp(infix[i]))
            {
                printf("%c ", pop());
            }
            push(infix[i]);
```



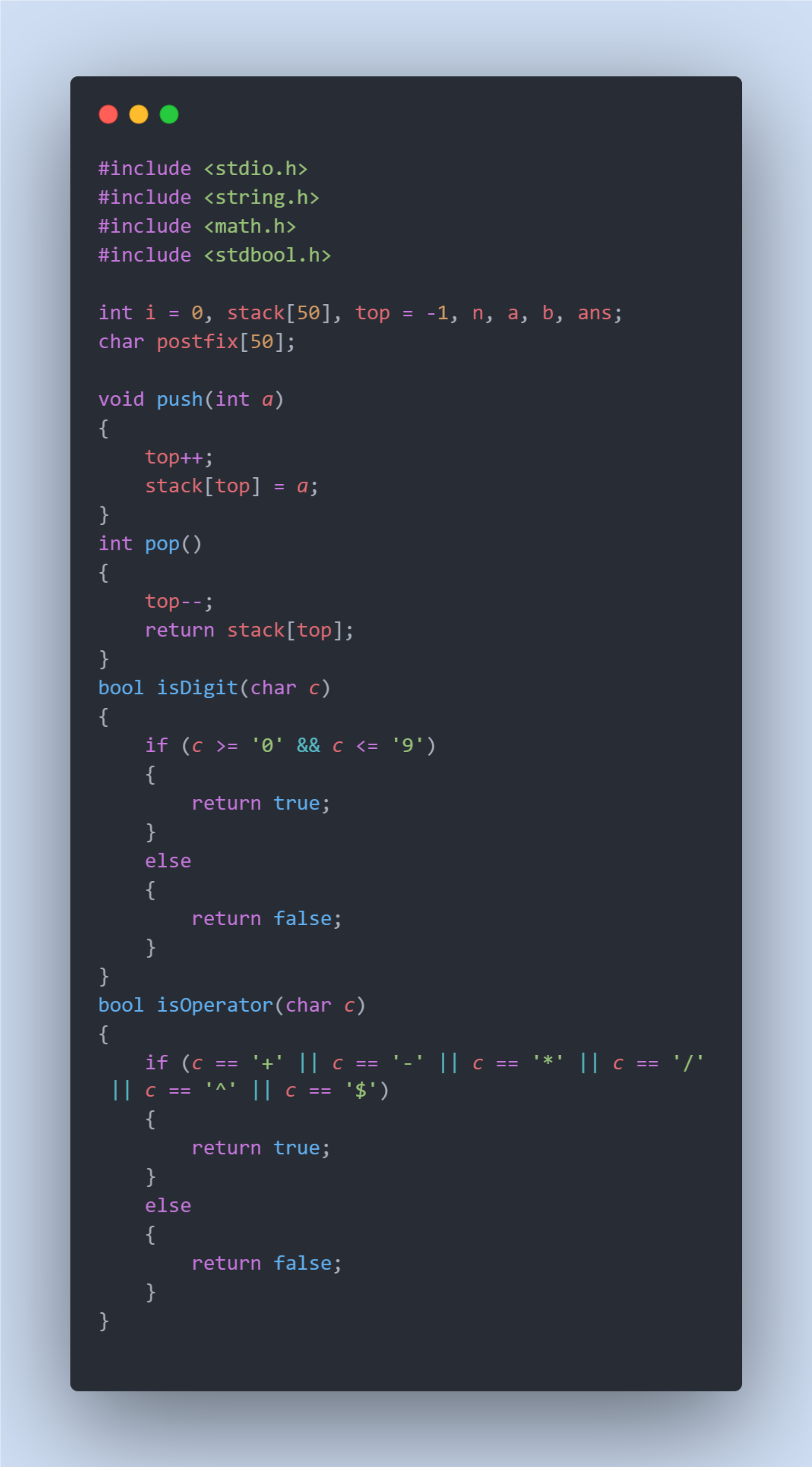
```
        }

    }
    while (top != -1)
    {
        printf("%c", pop());
    }
}
```

OUTPUT

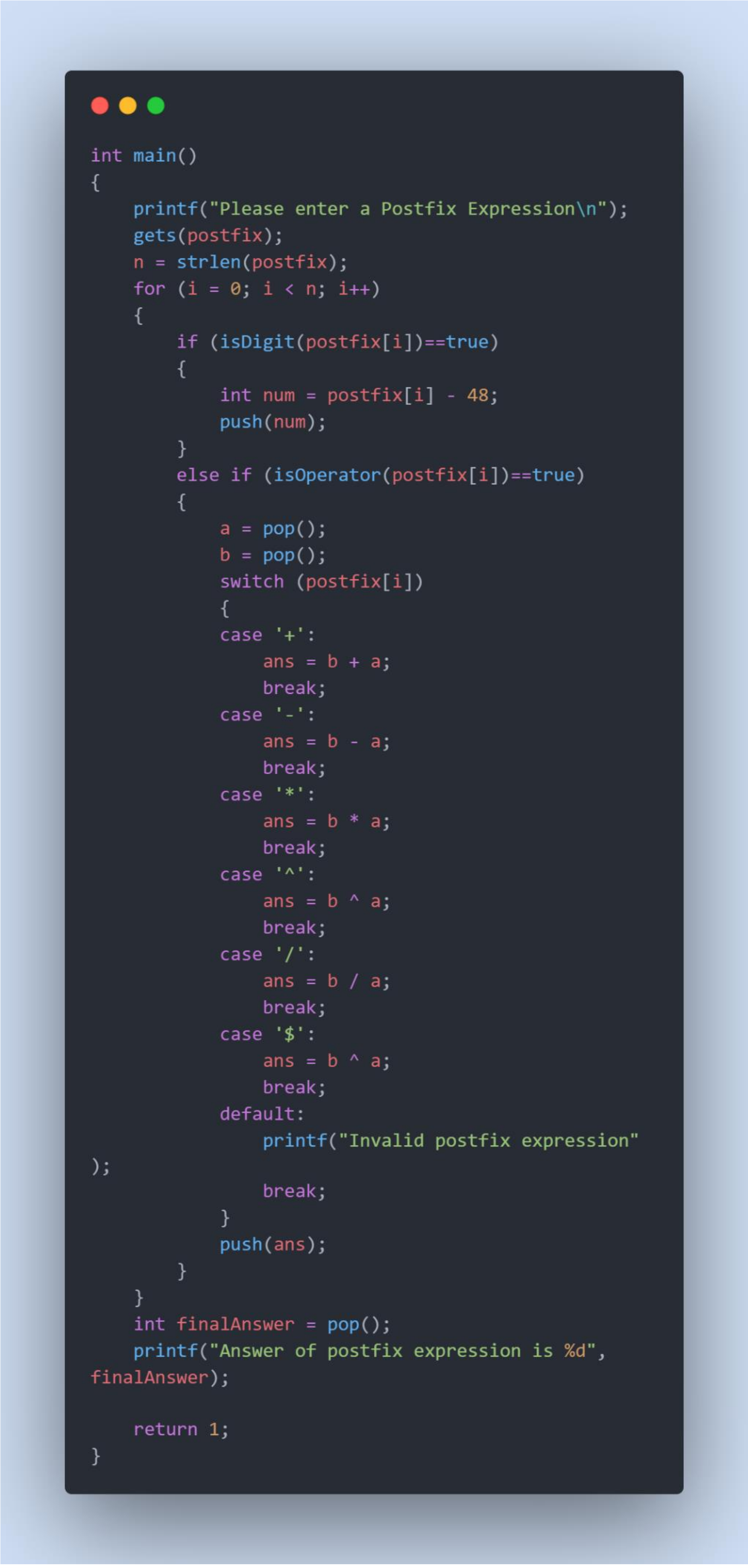


```
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Modul
nfixToPostfix } ; if ($?) { .\InfixToPostfix }
Please enter an Infix Expression
a+b
a b +
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Modul
fixToPostfix.c -o InfixToPostfix } ; if ($?) { .\Infi
Please enter an Infix Expression
A^B*C/(D*E-F)
A B ^ C * D E * F - /
```

ii) Evaluation of Postfix expression



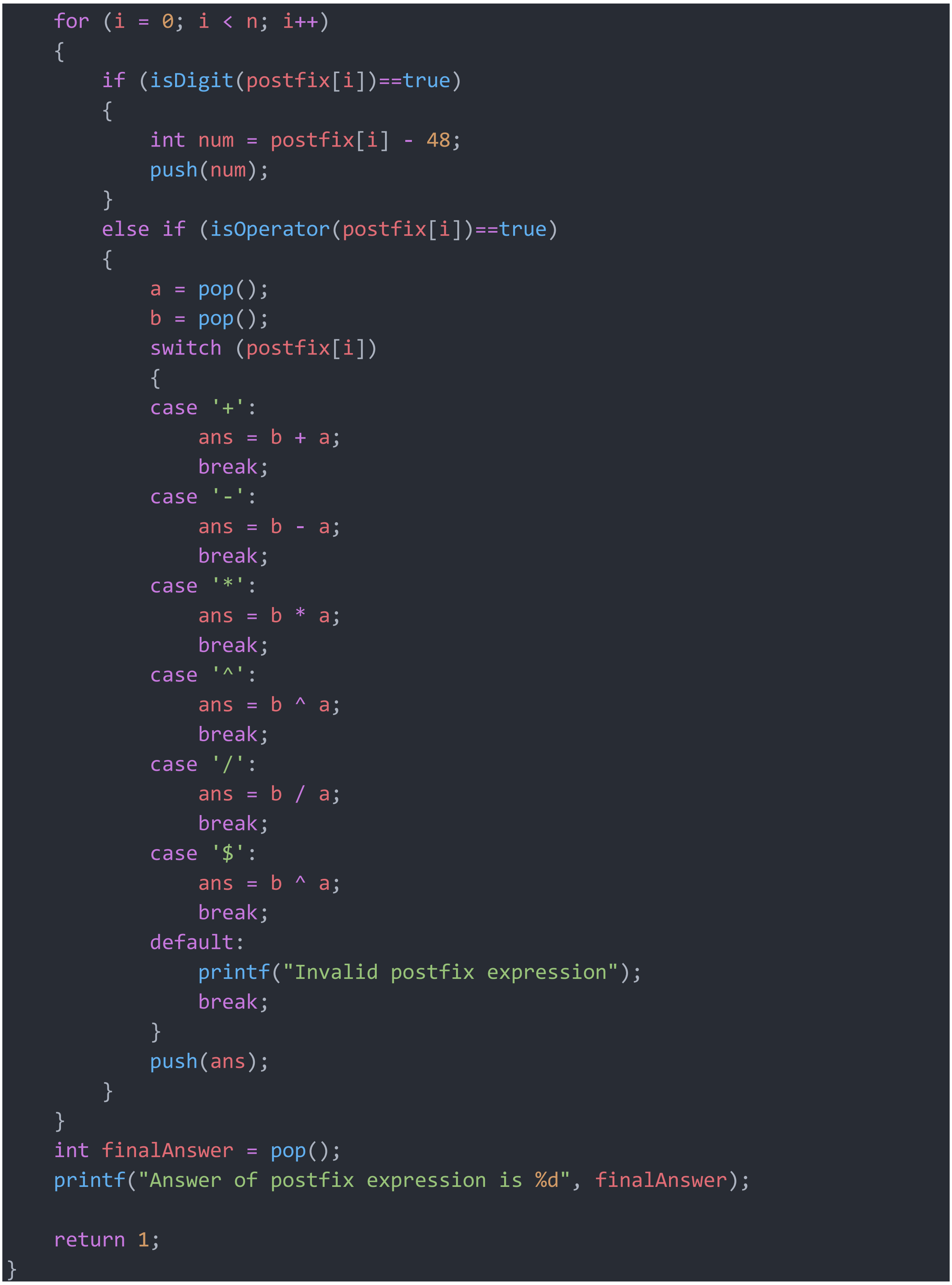
```c
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdbool.h>

int i = 0, stack[50], top = -1, n, a, b, ans;
char postfix[50];

void push(int a)
{
    top++;
    stack[top] = a;
}
int pop()
{
    top--;
    return stack[top];
}
bool isDigit(char c)
{
    if (c >= '0' && c <= '9')
    {
        return true;
    }
    else
    {
        return false;
    }
}
bool isOperator(char c)
{
    if (c == '+' || c == '-' || c == '*' || c == '/'
 || c == '^' || c == '$')
    {
        return true;
    }
    else
    {
        return false;
    }
}
```



```c
int main()
{
    printf("Please enter a Postfix Expression\n");
    gets(postfix);
    n = strlen(postfix);
    for (i = 0; i < n; i++)
    {
        if (isDigit(postfix[i])==true)
        {
            int num = postfix[i] - 48;
            push(num);
        }
        else if (isOperator(postfix[i])==true)
        {
            a = pop();
            b = pop();
            switch (postfix[i])
            {
            case '+':
                ans = b + a;
                break;
            case '-':
                ans = b - a;
                break;
            case '*':
                ans = b * a;
                break;
            case '^':
                ans = b ^ a;
                break;
            case '/':
                ans = b / a;
                break;
            case '$':
                ans = b ^ a;
                break;
            default:
                printf("Invalid postfix expression"
);
                break;
            }
            push(ans);
        }
    }
    int finalAnswer = pop();
    printf("Answer of postfix expression is %d",
finalAnswer);

    return 1;
}
```

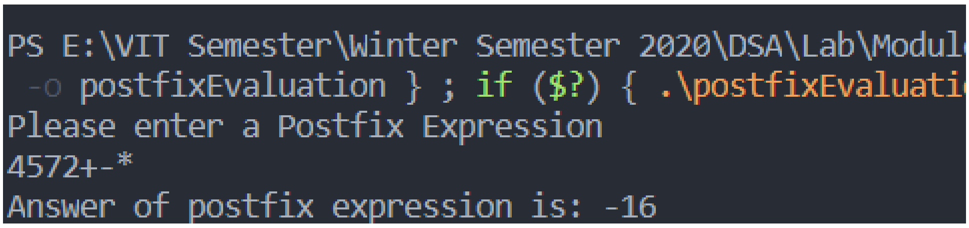
Complete Code

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdbool.h>

int i = 0, stack[50], top = -1, n, a, b, ans;
char postfix[50];

void push(int a)
{
    top++;
    stack[top] = a;
}
int pop()
{
    return stack[top--];
}
bool isDigit(char c)
{
    if (c >= '0' && c <= '9')
    {
        return true;
    }
    else
    {
        return false;
    }
}
bool isOperator(char c)
{
    if (c == '+' || c == '-' || c == '*' || c == '/' || c == '^' || c == '$')
    {
        return true;
    }
    else
    {
        return false;
    }
}

int main()
{
    printf("Please enter a Postfix Expression\n");
    gets(postfix);
    n = strlen(postfix);
```



```
    for (i = 0; i < n; i++)
    {
        if (isDigit(postfix[i])==true)
        {
            int num = postfix[i] - 48;
            push(num);
        }
        else if (isOperator(postfix[i])==true)
        {
            a = pop();
            b = pop();
            switch (postfix[i])
            {
            case '+':
                ans = b + a;
                break;
            case '-':
                ans = b - a;
                break;
            case '*':
                ans = b * a;
                break;
            case '^':
                ans = b ^ a;
                break;
            case '/':
                ans = b / a;
                break;
            case '$':
                ans = b ^ a;
                break;
            default:
                printf("Invalid postfix expression");
                break;
            }
            push(ans);
        }
    }
    int finalAnswer = pop();
    printf("Answer of postfix expression is %d", finalAnswer);

    return 1;
}
```

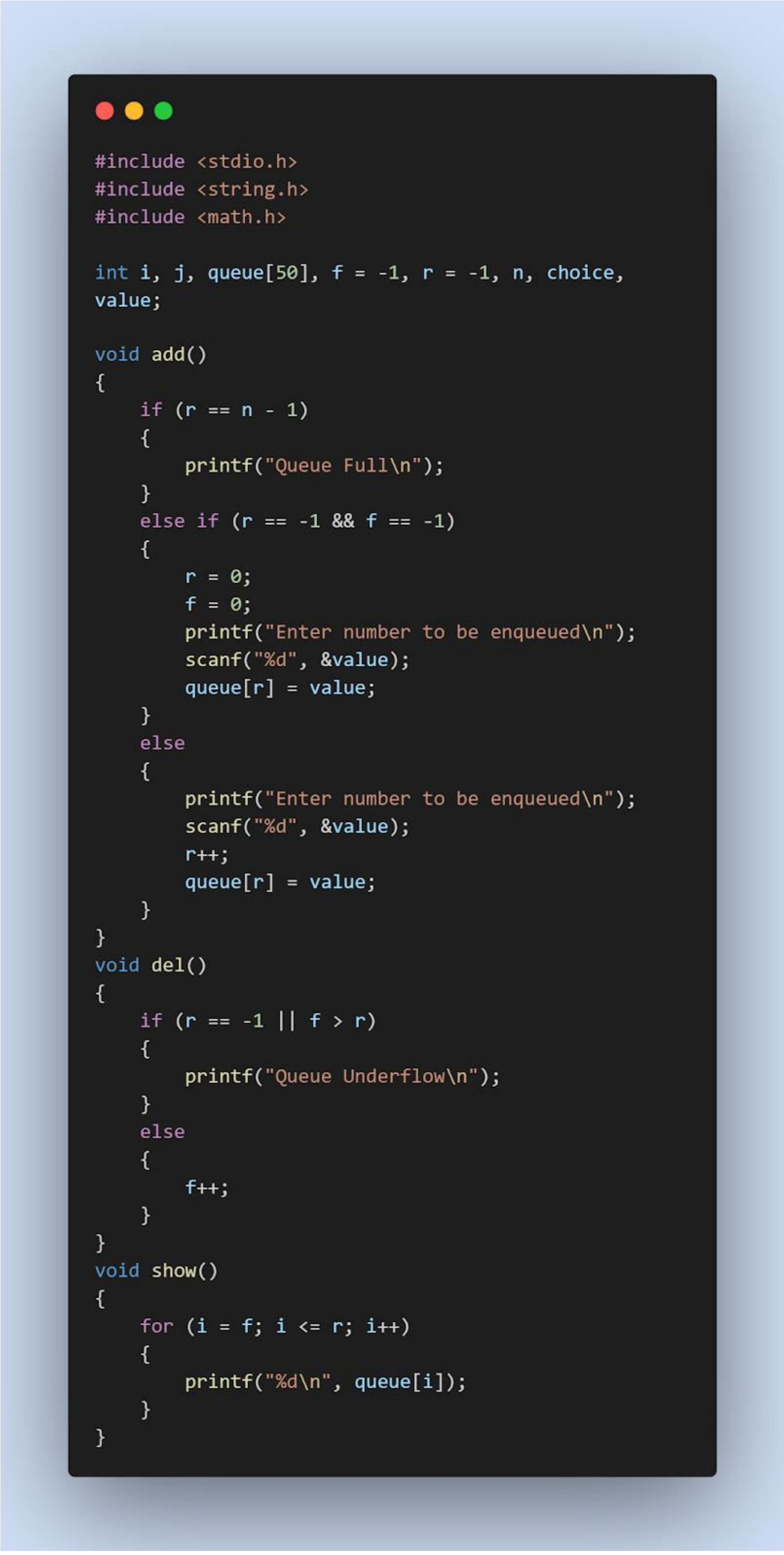
OUTPUT



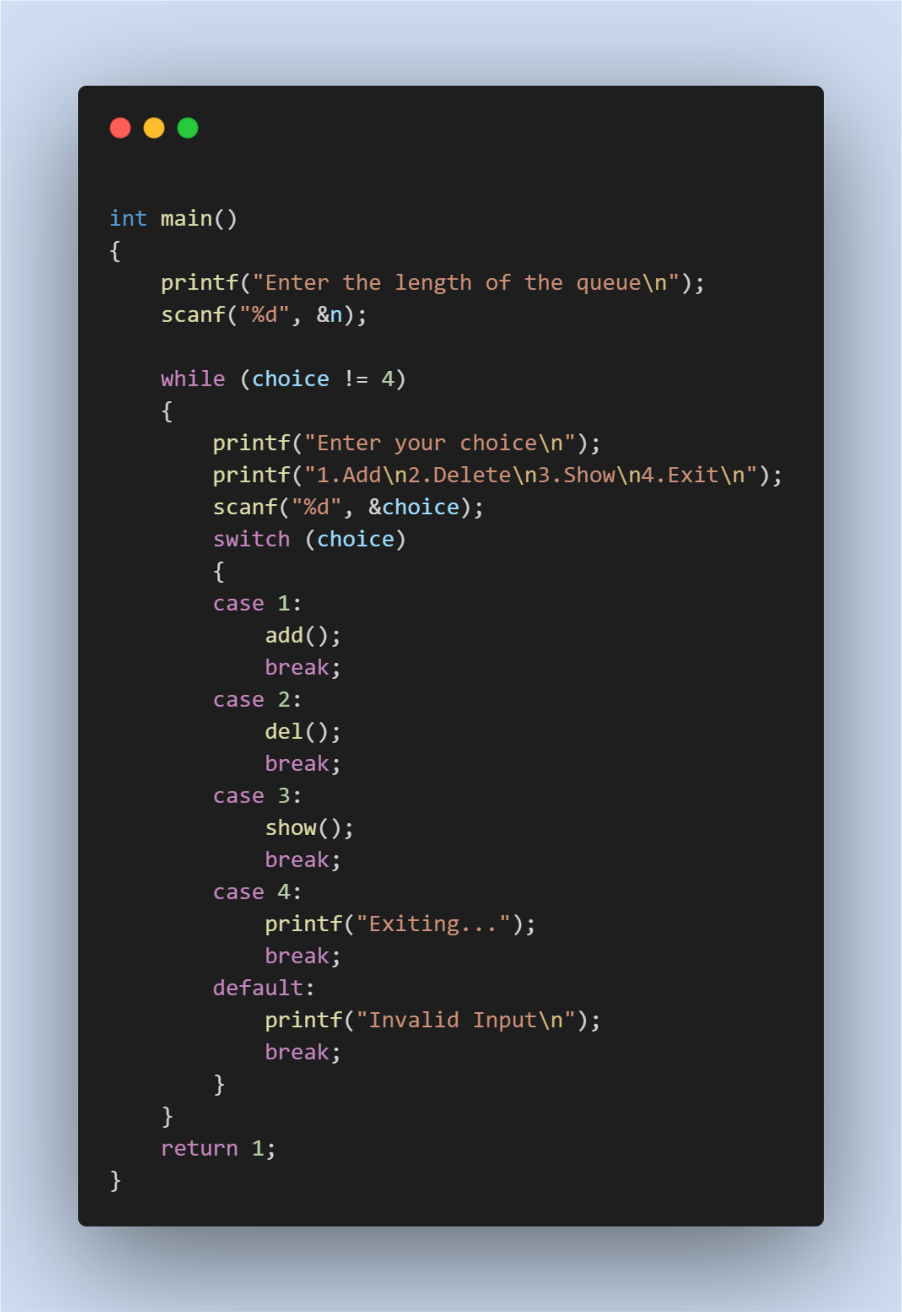
```
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Modul
 -o postfixEvaluation } ; if ($?) { .\postfixEvaluati
Please enter a Postfix Expression
4572+-*
Answer of postfix expression is: -16
```

2. Write a menu driven program to perform static implementation of a queue data structure with all possible functions.

CODE

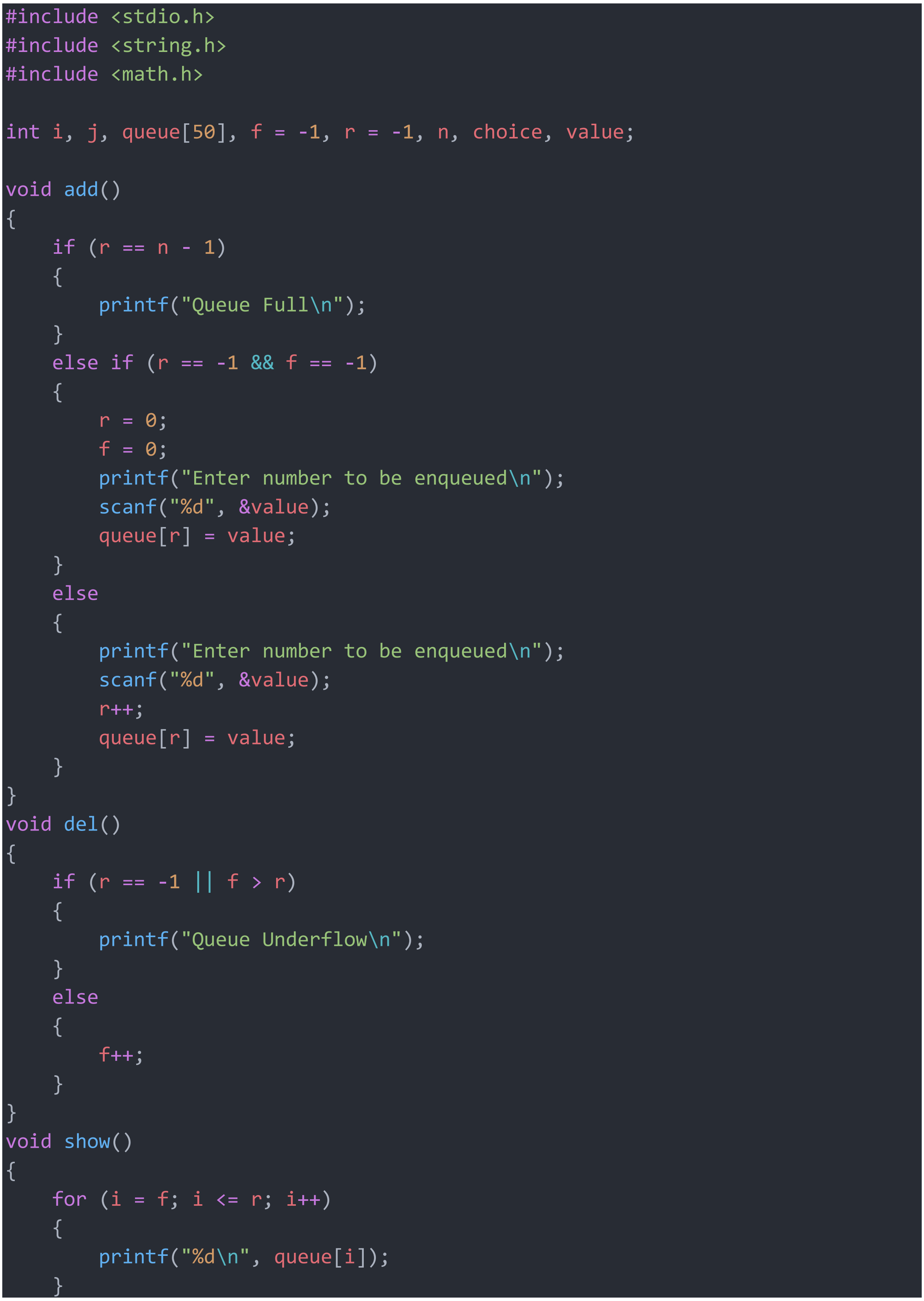


```
#include <stdio.h>
#include <string.h>
#include <math.h>

int i, j, queue[50], f = -1, r = -1, n, choice,
value;

void add()
{
    if (r == n - 1)
    {
        printf("Queue Full\n");
    }
    else if (r == -1 && f == -1)
    {
        r = 0;
        f = 0;
        printf("Enter number to be enqueued\n");
        scanf("%d", &value);
        queue[r] = value;
    }
    else
    {
        printf("Enter number to be enqueued\n");
        scanf("%d", &value);
        r++;
        queue[r] = value;
    }
}
void del()
{
    if (r == -1 || f > r)
    {
        printf("Queue Underflow\n");
    }
    else
    {
        f++;
    }
}
void show()
{
    for (i = f; i <= r; i++)
    {
        printf("%d\n", queue[i]);
    }
}
```

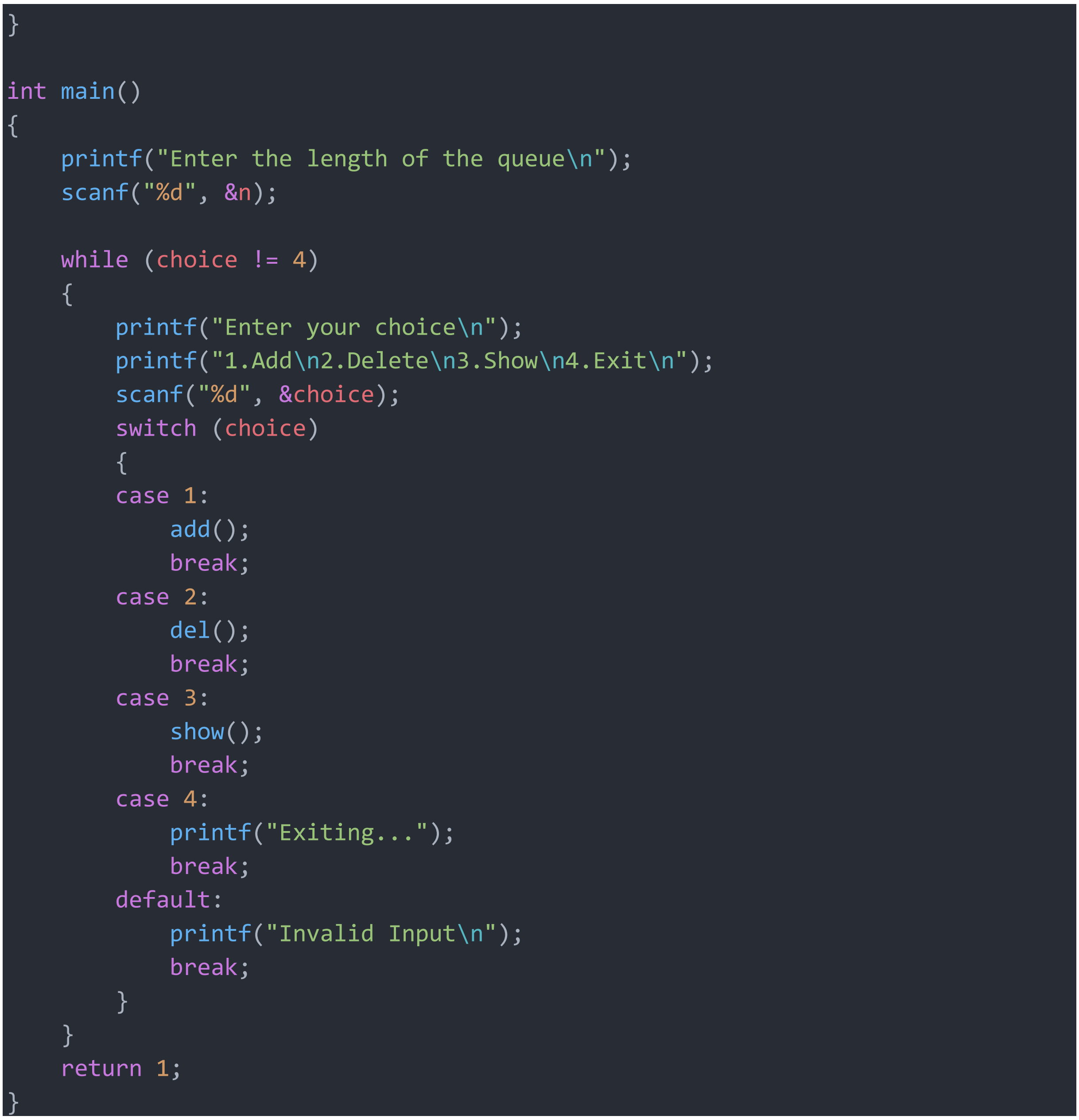


```c
int main()
{
    printf("Enter the length of the queue\n");
    scanf("%d", &n);

    while (choice != 4)
    {
        printf("Enter your choice\n");
        printf("1.Add\n2.Delete\n3.Show\n4.Exit\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            add();
            break;
        case 2:
            del();
            break;
        case 3:
            show();
            break;
        case 4:
            printf("Exiting...");
            break;
        default:
            printf("Invalid Input\n");
            break;
        }
    }
    return 1;
}
```

Complete Code

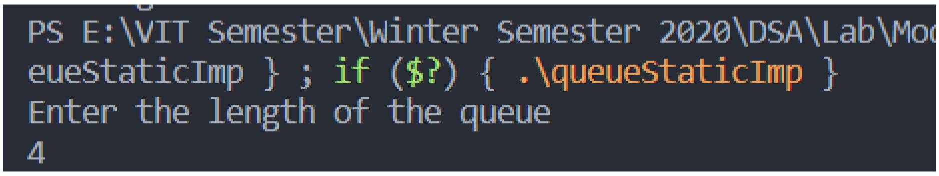


```c
#include <stdio.h>
#include <string.h>
#include <math.h>

int i, j, queue[50], f = -1, r = -1, n, choice, value;

void add()
{
    if (r == n - 1)
    {
        printf("Queue Full\n");
    }
    else if (r == -1 && f == -1)
    {
        r = 0;
        f = 0;
        printf("Enter number to be enqueued\n");
        scanf("%d", &value);
        queue[r] = value;
    }
    else
    {
        printf("Enter number to be enqueued\n");
        scanf("%d", &value);
        r++;
        queue[r] = value;
    }
}
void del()
{
    if (r == -1 || f > r)
    {
        printf("Queue Underflow\n");
    }
    else
    {
        f++;
    }
}
void show()
{
    for (i = f; i <= r; i++)
    {
        printf("%d\n", queue[i]);
    }
```

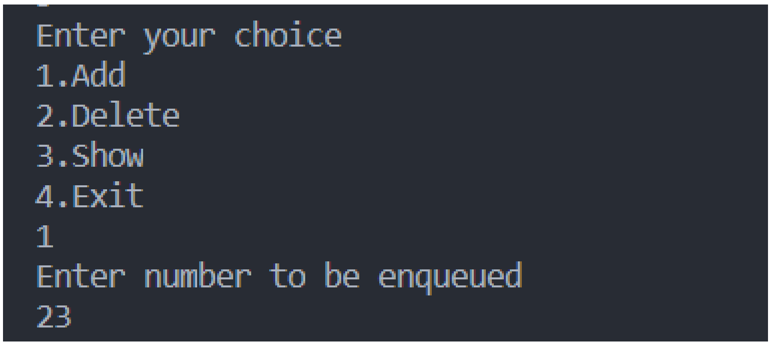


```c
}

int main()
{
    printf("Enter the length of the queue\n");
    scanf("%d", &n);

    while (choice != 4)
    {
        printf("Enter your choice\n");
        printf("1.Add\n2.Delete\n3.Show\n4.Exit\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            add();
            break;
        case 2:
            del();
            break;
        case 3:
            show();
            break;
        case 4:
            printf("Exiting...");
            break;
        default:
            printf("Invalid Input\n");
            break;
        }
    }
    return 1;
}
```

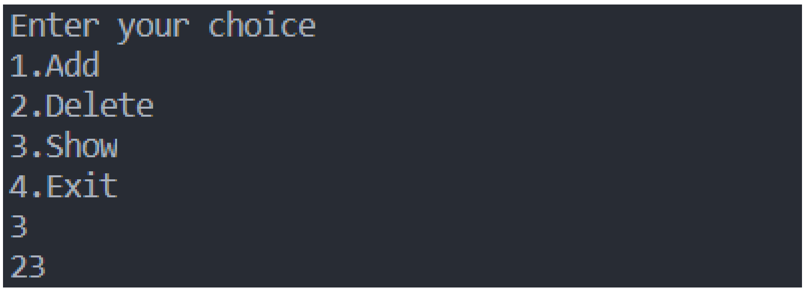
OUTPUT



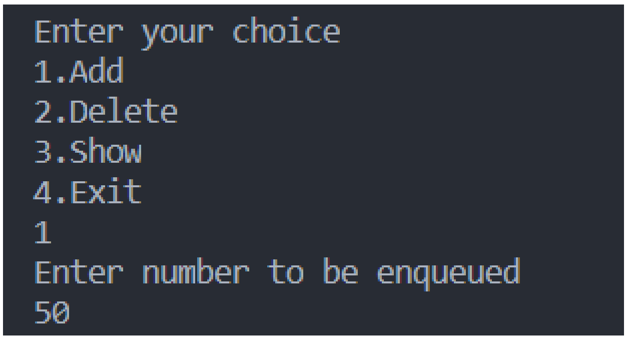
```
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Moc
eueStaticImp } ; if ($?) { .\queueStaticImp }
Enter the length of the queue
4
```



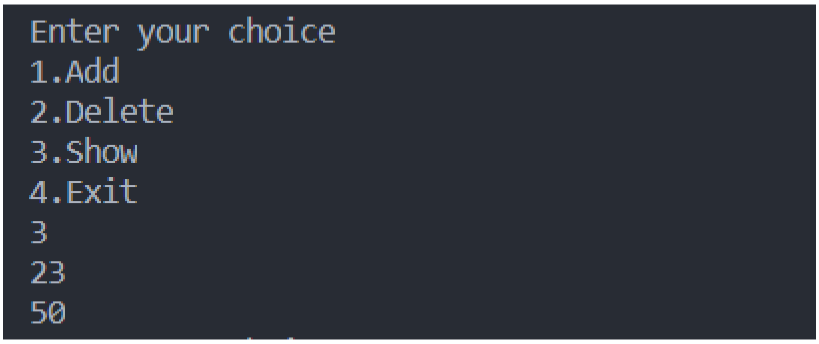
```
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
23
```



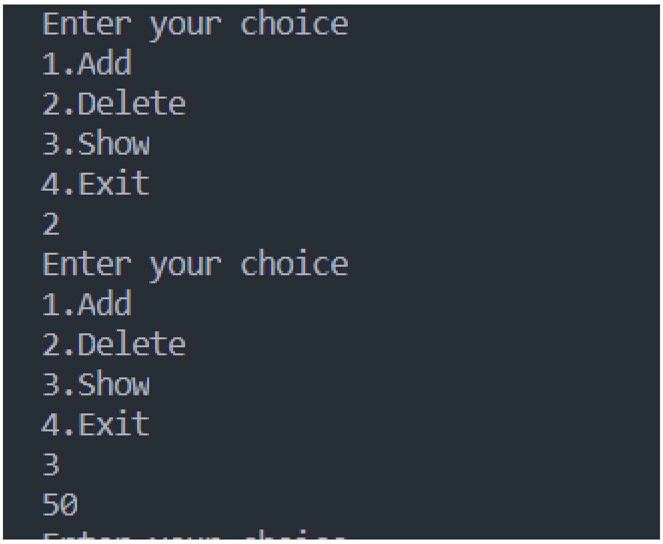
```
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
23
```



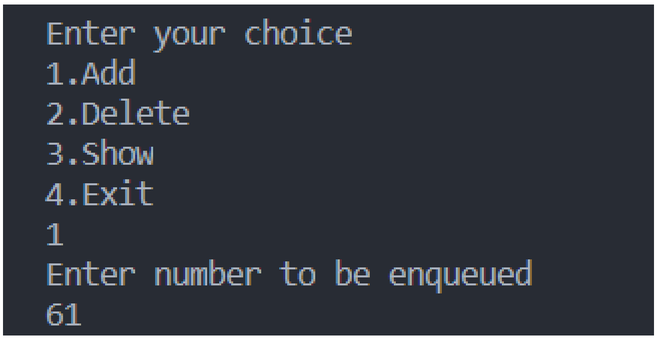
```
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
50
```



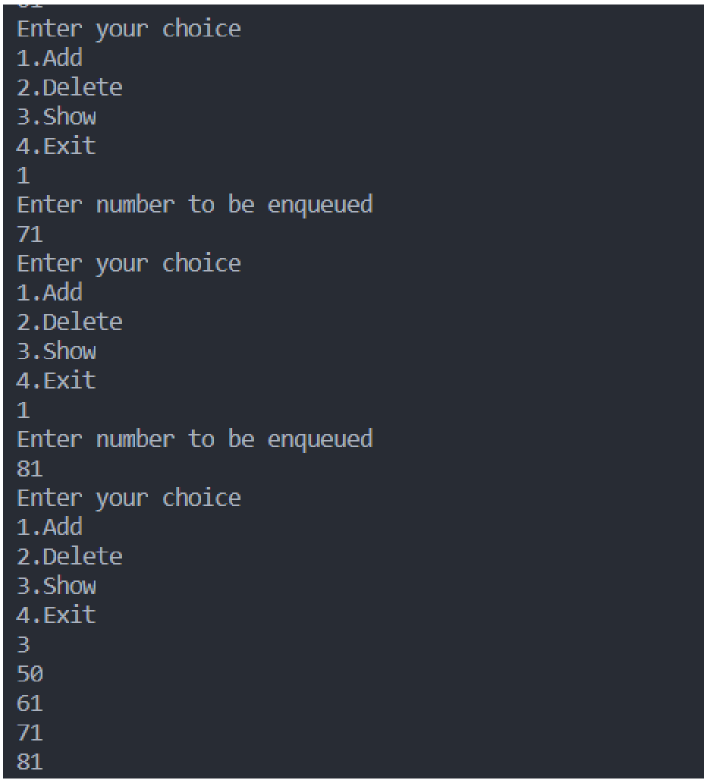
```
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
23
50
```



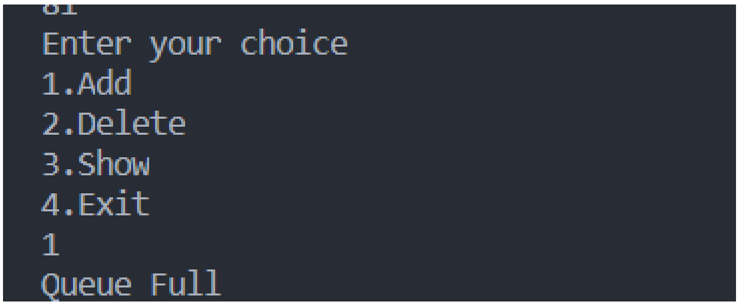
```
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
50
Enter your choice
```



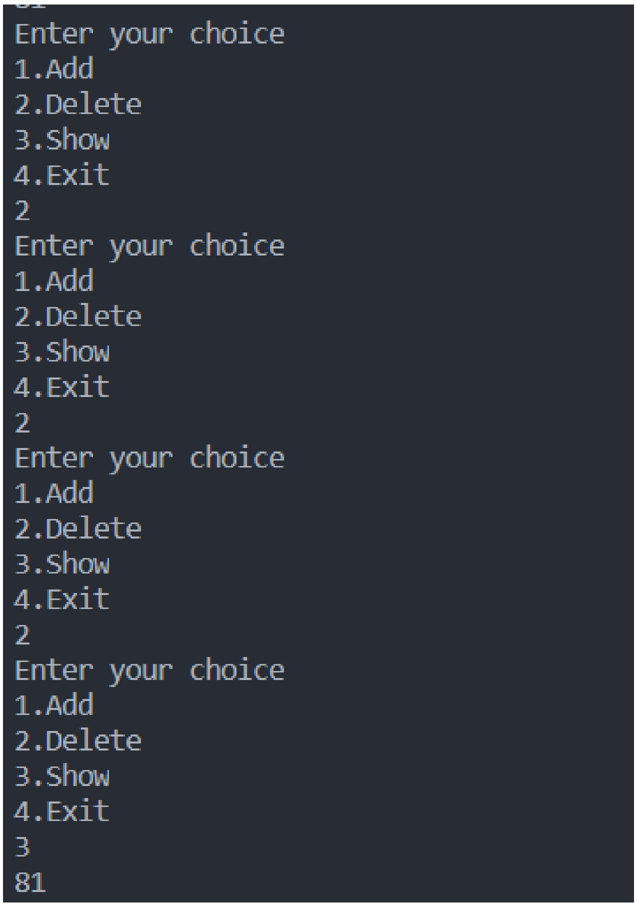
```
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
61
```



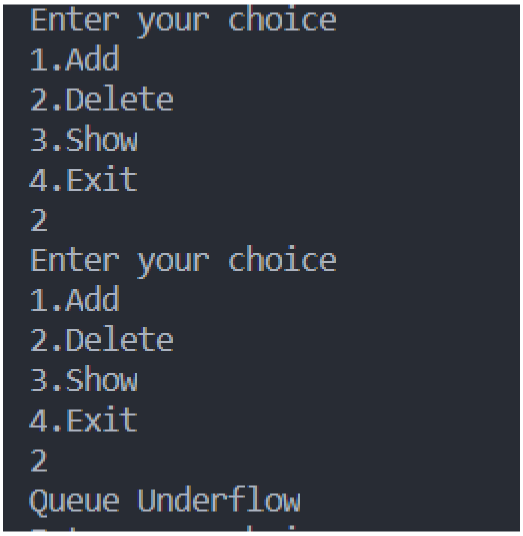
```
61
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
71
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
81
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
50
61
71
81
```



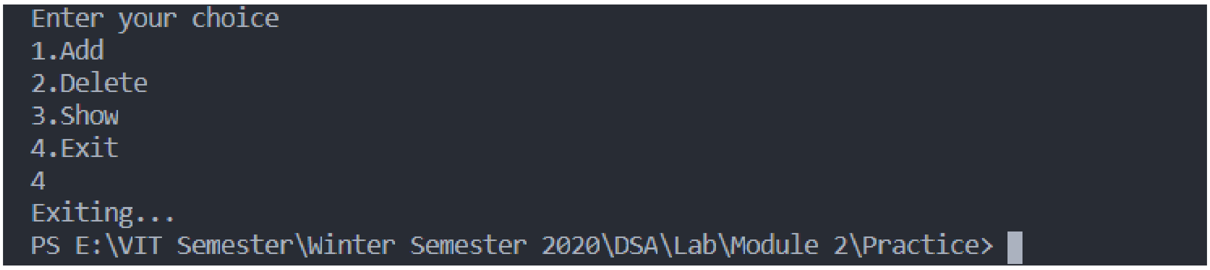
```
 81
 Enter your choice
 1.Add
 2.Delete
 3.Show
 4.Exit
 1
 Queue Full
```



```
81
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
81
```



```
 Enter your choice
 1.Add
 2.Delete
 3.Show
 4.Exit
 2
 Enter your choice
 1.Add
 2.Delete
 3.Show
 4.Exit
 2
 Queue Underflow
 Enter your choice
```



```
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
4
Exiting...
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2\Practice>
```

```
Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.


Try the new cross-platform PowerShell https://aka.ms/pscore6


PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2> cd "e:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2\Practice\" ; if ($?) { gcc queueStaticImp.c -o queueStaticImp } ; if ($?) { .\queueStaticImp }

Enter the length of the queue

4

Enter your choice

1.Add

2.Delete

3.Show

4.Exit

1

Enter number to be enqueued

23

Enter your choice

1.Add

2.Delete

3.Show

4.Exit

3

23

Enter your choice
```

```
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
50
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
23
50
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
50
Enter your choice
1.Add
```

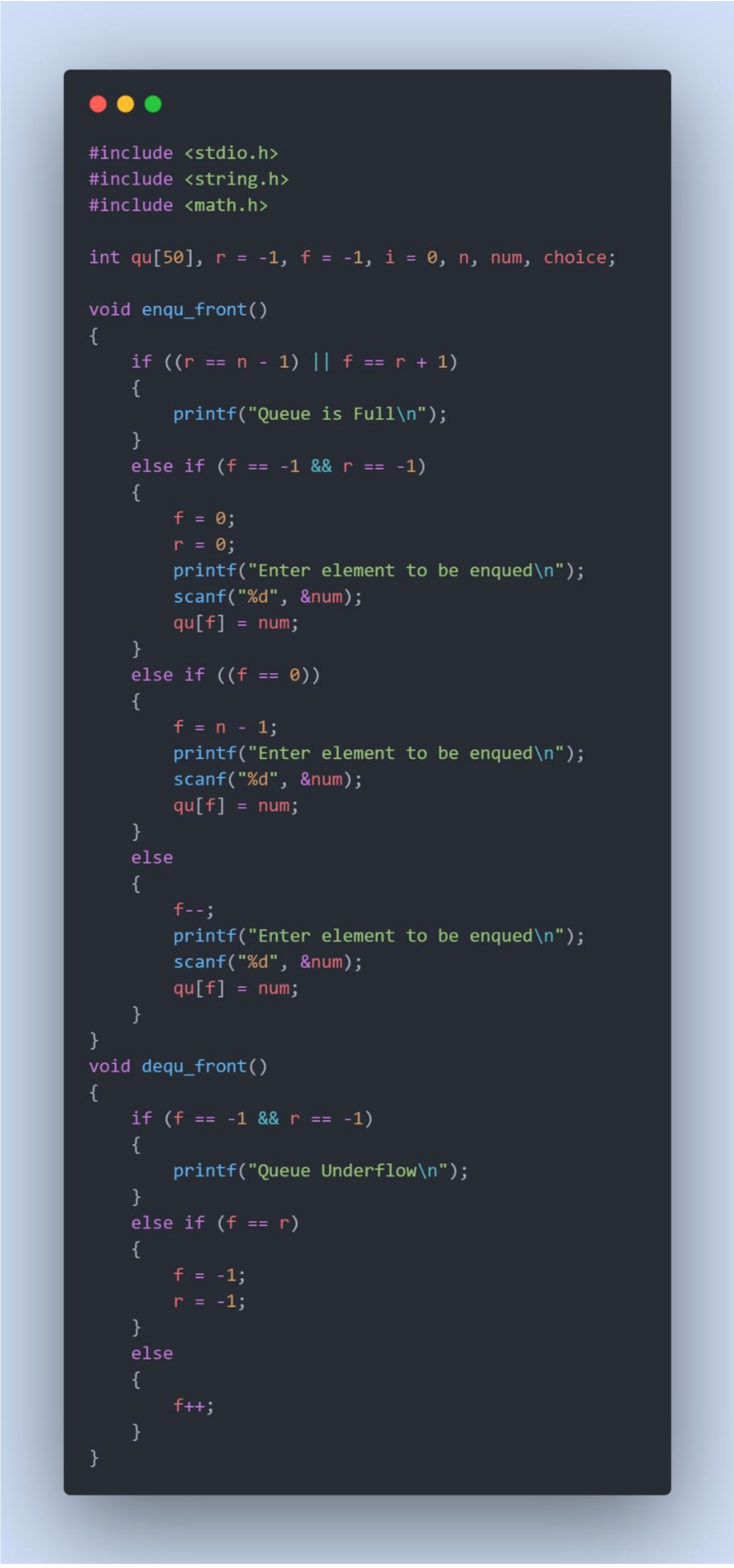
```
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
61
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
71
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Enter number to be enqueued
81
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
50
61
```

```
71
81
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
1
Queue Full
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
50
61
71
81
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
```

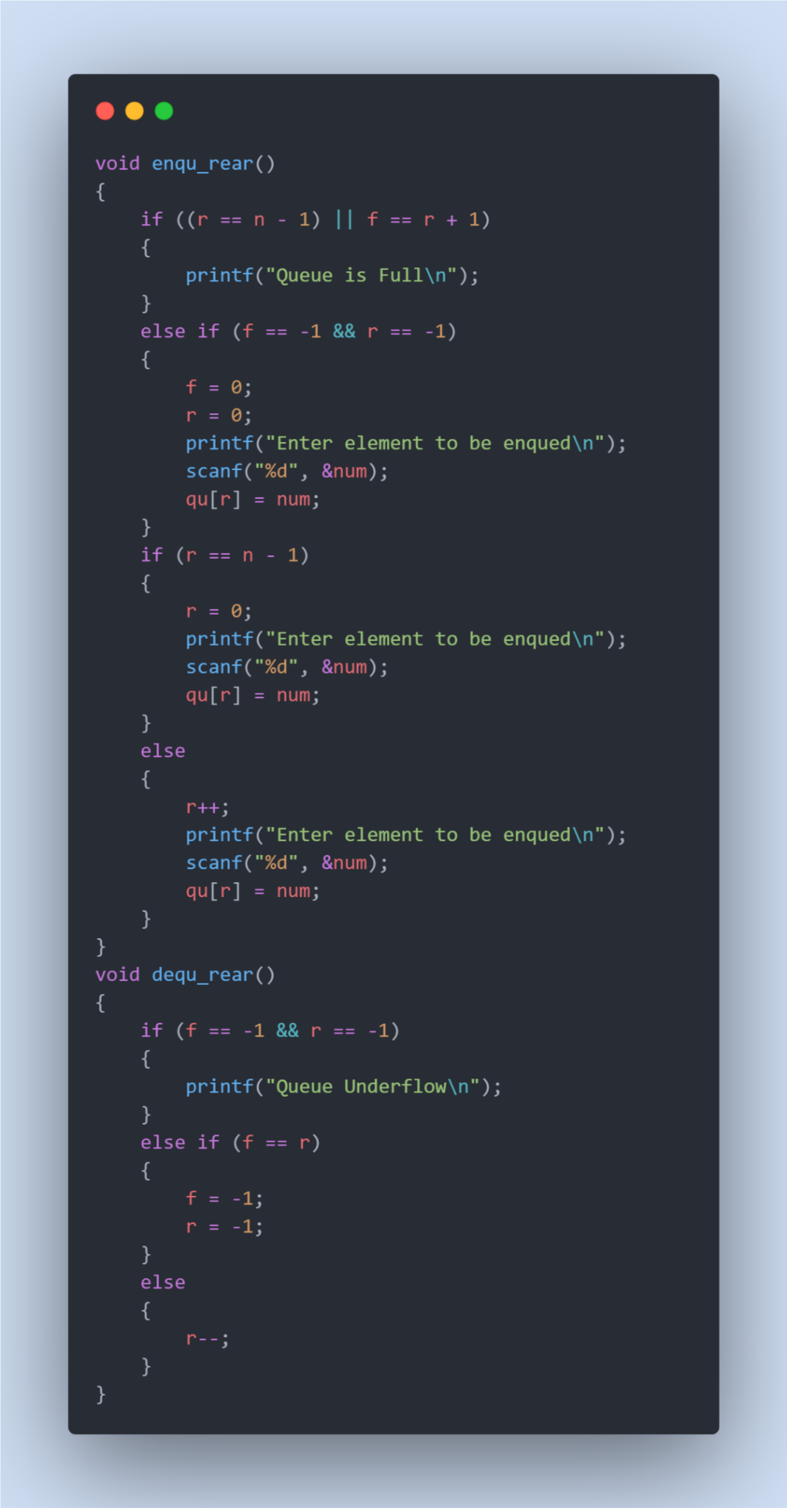
```
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
3
81
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Enter your choice
1.Add
2.Delete
3.Show
4.Exit
2
Queue Underflow
Enter your choice
1.Add
2.Delete
```

```
3.Show
4.Exit
4
Exiting...
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2\Practice>
```

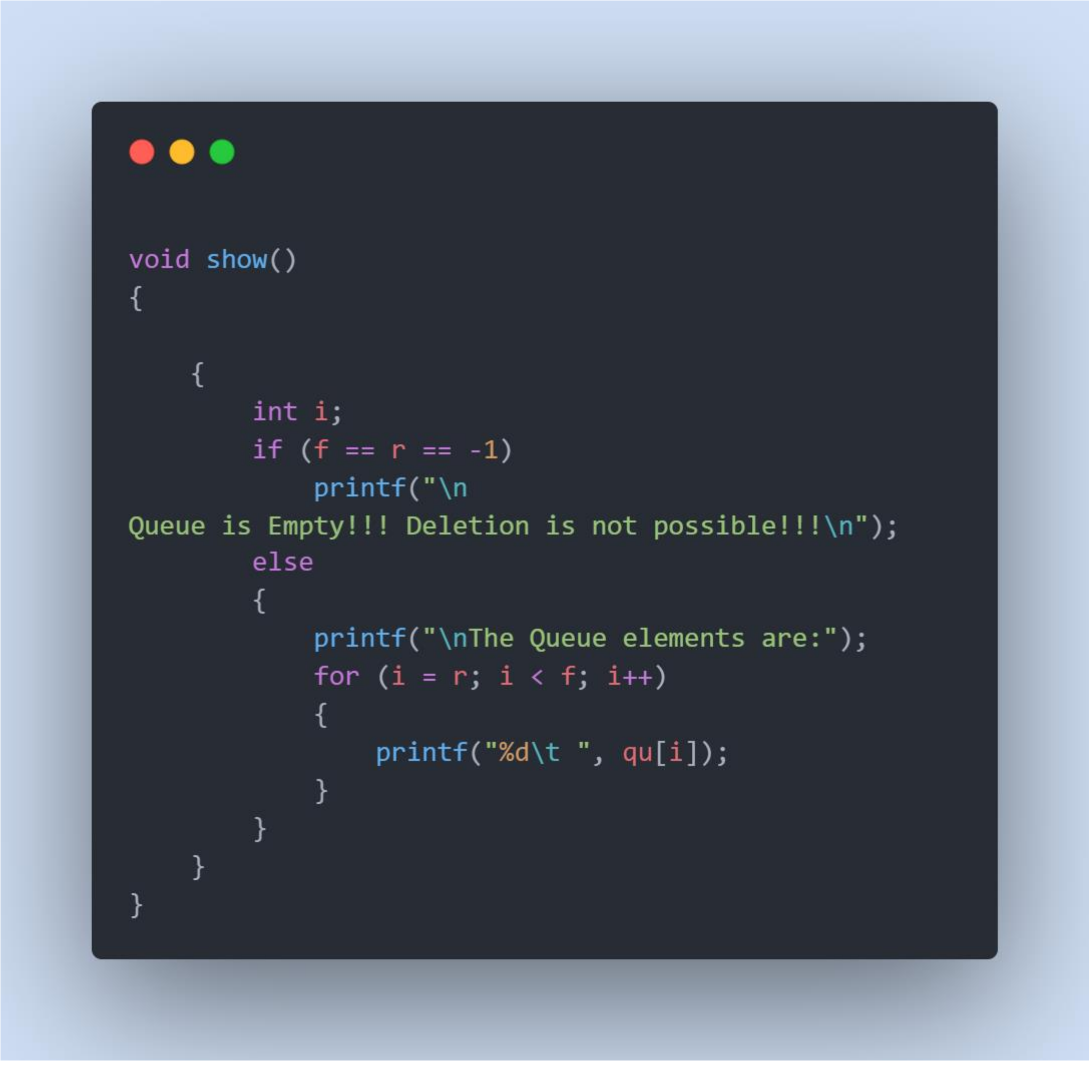
3. Write a program using arrays to perform insertion and deletion from both the ends of a Deque and also display the contents of it based on the choice given by the user.



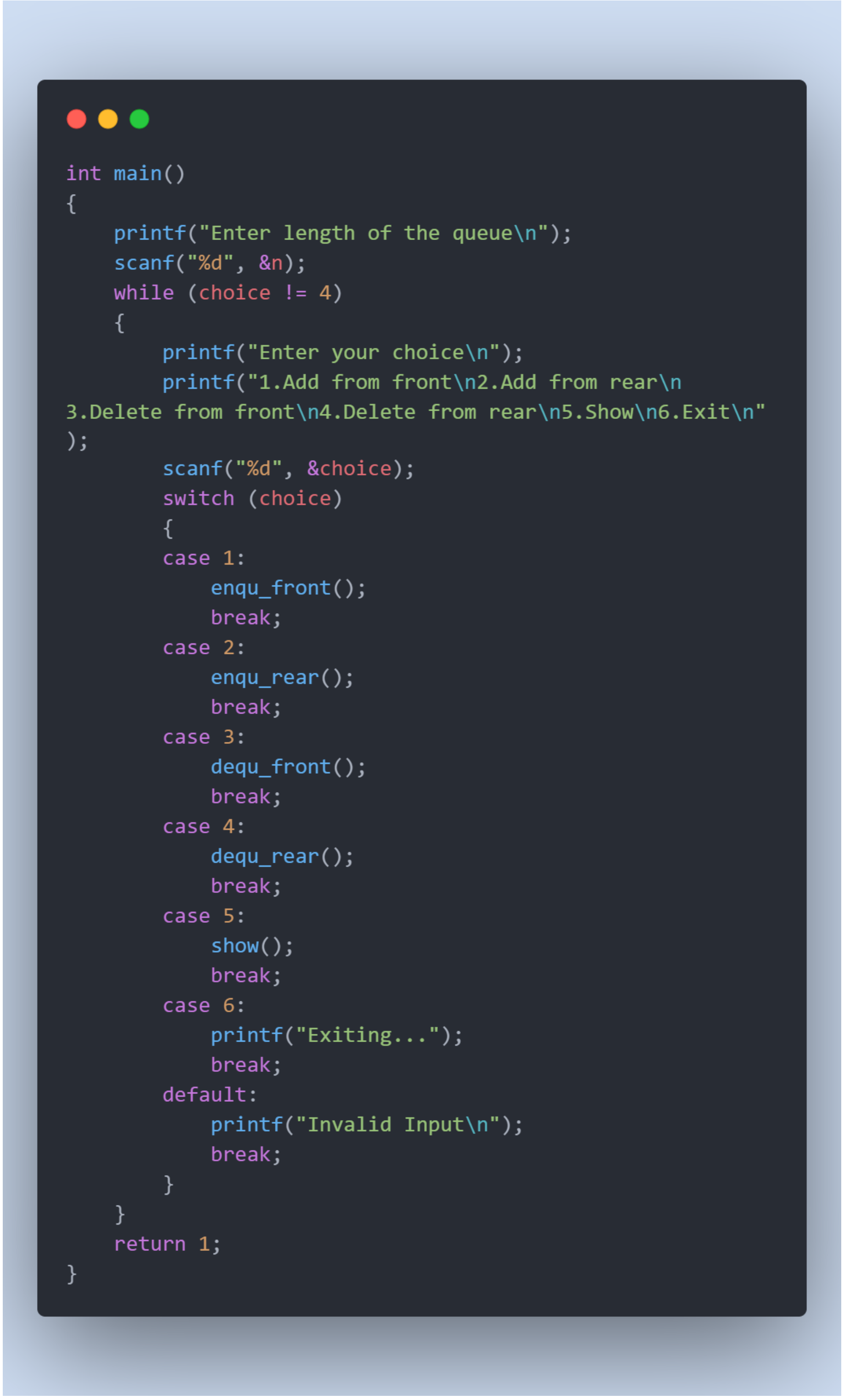
```c
#include <stdio.h>
#include <string.h>
#include <math.h>

int qu[50], r = -1, f = -1, i = 0, n, num, choice;

void enqu_front()
{
    if ((r == n - 1) || f == r + 1)
    {
        printf("Queue is Full\n");
    }
    else if (f == -1 && r == -1)
    {
        f = 0;
        r = 0;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[f] = num;
    }
    else if ((f == 0))
    {
        f = n - 1;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[f] = num;
    }
    else
    {
        f--;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[f] = num;
    }
}
void dequ_front()
{
    if (f == -1 && r == -1)
    {
        printf("Queue Underflow\n");
    }
    else if (f == r)
    {
        f = -1;
        r = -1;
    }
    else
    {
        f++;
    }
}
```



```c
void enqu_rear()
{
    if ((r == n - 1) || f == r + 1)
    {
        printf("Queue is Full\n");
    }
    else if (f == -1 && r == -1)
    {
        f = 0;
        r = 0;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[r] = num;
    }
    if (r == n - 1)
    {
        r = 0;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[r] = num;
    }
    else
    {
        r++;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[r] = num;
    }
}
void dequ_rear()
{
    if (f == -1 && r == -1)
    {
        printf("Queue Underflow\n");
    }
    else if (f == r)
    {
        f = -1;
        r = -1;
    }
    else
    {
        r--;
    }
}
```



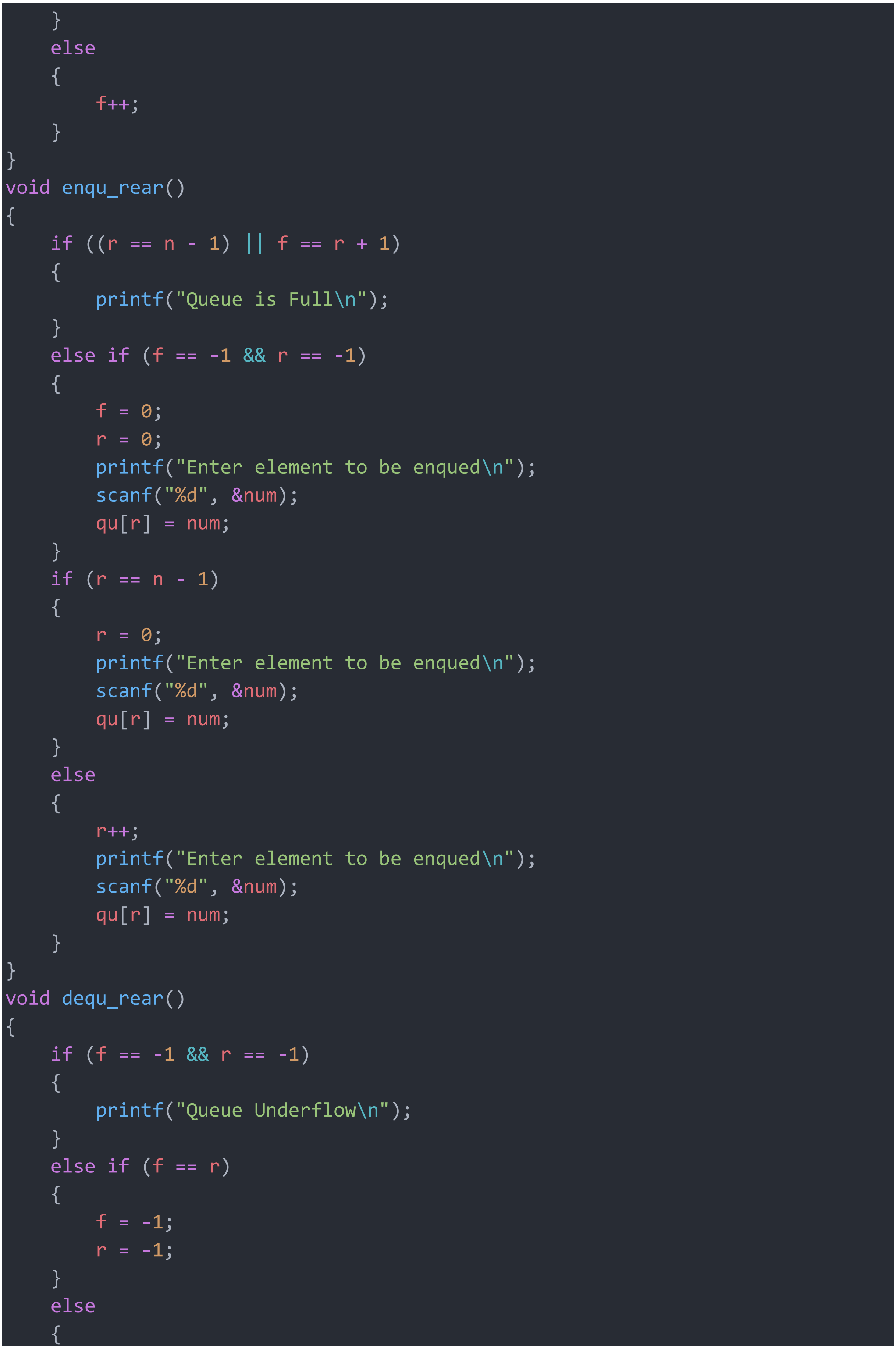
```
void show()
{

    {
        int i;
        if (f == r == -1)
            printf("\n
Queue is Empty!!! Deletion is not possible!!!\n");
        else
        {
            printf("\nThe Queue elements are:");
            for (i = r; i < f; i++)
            {
                printf("%d\t ", qu[i]);
            }
        }
    }
}
```



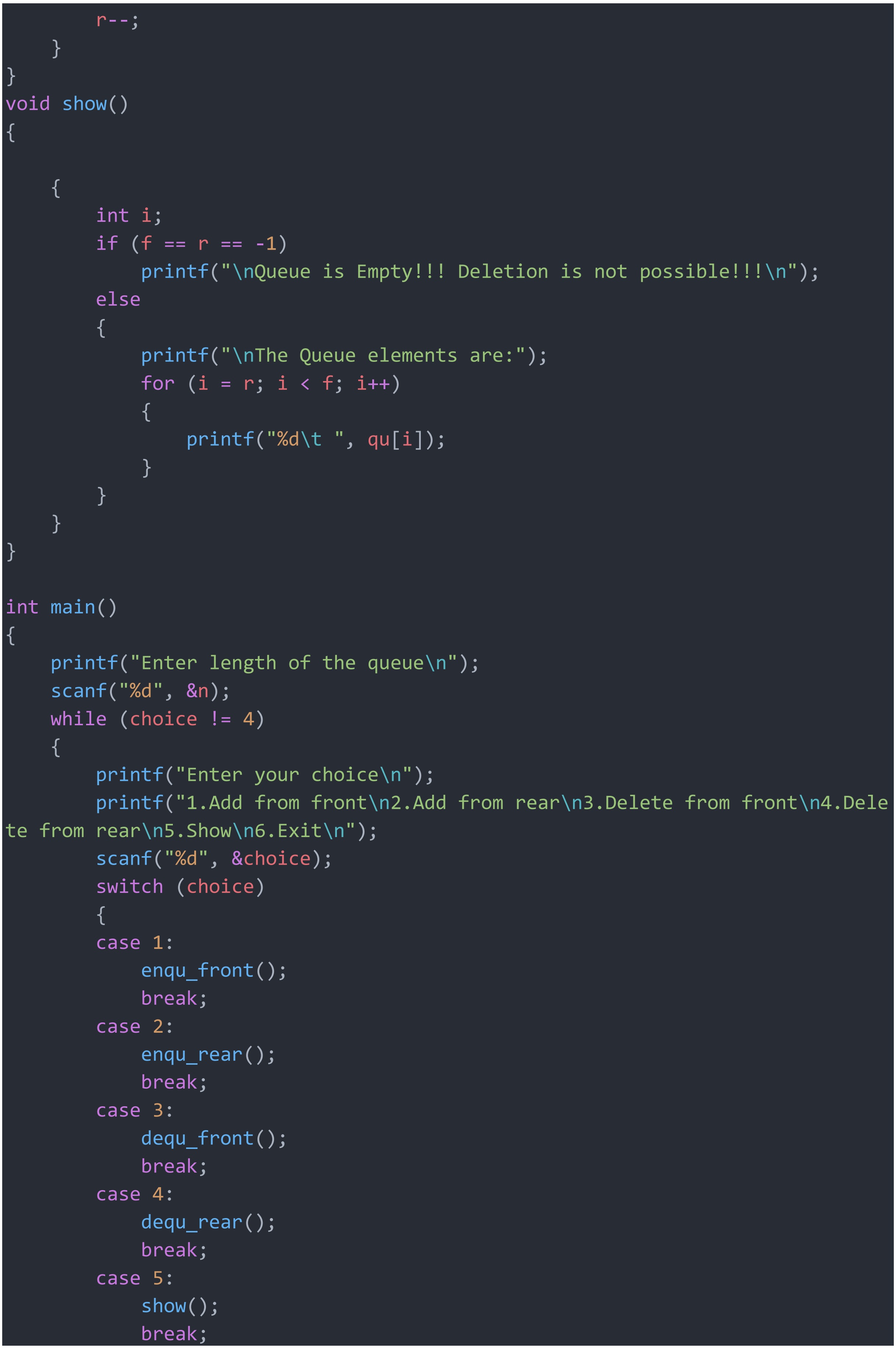
```c
int main()
{
    printf("Enter length of the queue\n");
    scanf("%d", &n);
    while (choice != 4)
    {
        printf("Enter your choice\n");
        printf("1.Add from front\n2.Add from rear\n
3.Delete from front\n4.Delete from rear\n5.Show\n6.Exit\n"
);
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            enqu_front();
            break;
        case 2:
            enqu_rear();
            break;
        case 3:
            dequ_front();
            break;
        case 4:
            dequ_rear();
            break;
        case 5:
            show();
            break;
        case 6:
            printf("Exiting...");
            break;
        default:
            printf("Invalid Input\n");
            break;
        }
    }
    return 1;
}
```

Complete Code

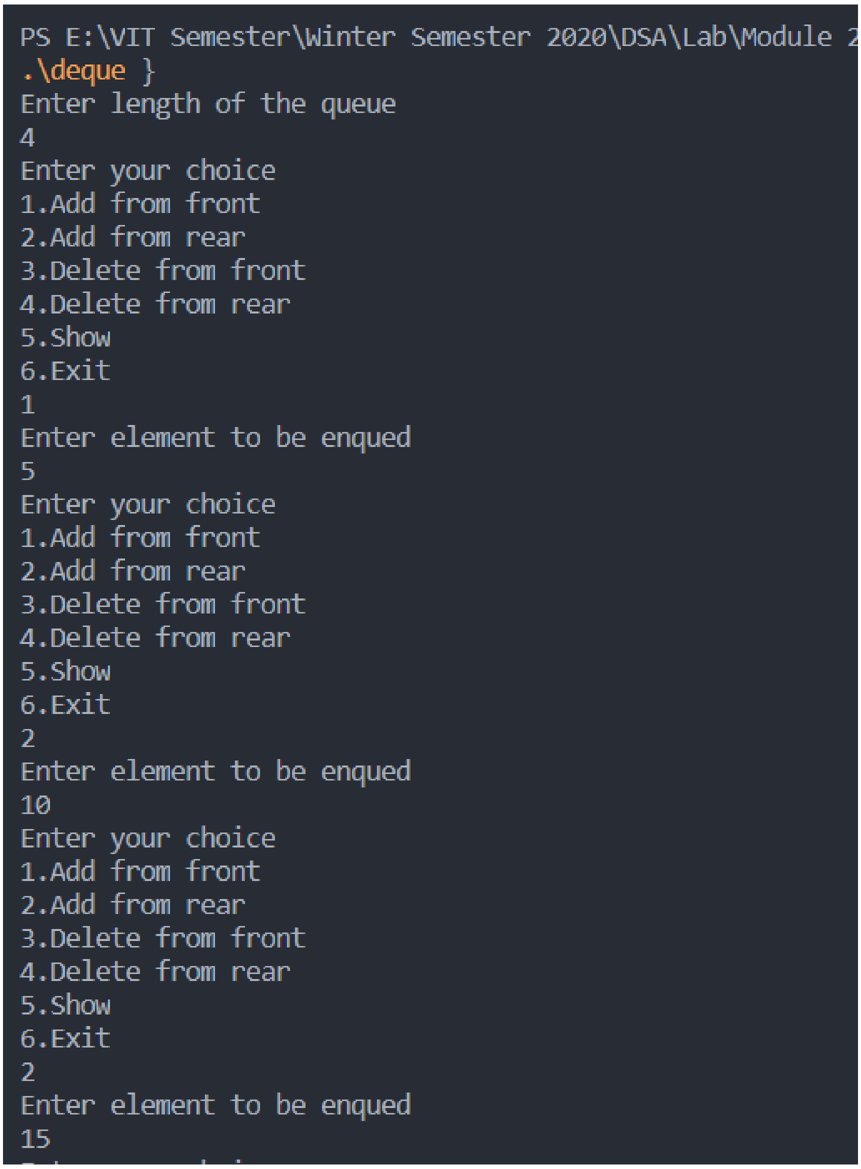
```
#include <stdio.h>
#include <string.h>
#include <math.h>

int qu[50], r = -1, f = -1, i = 0, n, num, choice;

void enqu_front()
{
    if ((r == n - 1) || f == r + 1)
    {
        printf("Queue is Full\n");
    }
    else if (f == -1 && r == -1)
    {
        f = 0;
        r = 0;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[f] = num;
    }
    else if ((f == 0))
    {
        f = n - 1;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[f] = num;
    }
    else
    {
        f--;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[f] = num;
    }
}
void dequ_front()
{
    if (f == -1 && r == -1)
    {
        printf("Queue Underflow\n");
    }
    else if (f == r)
    {
        f = -1;
        r = -1;
```



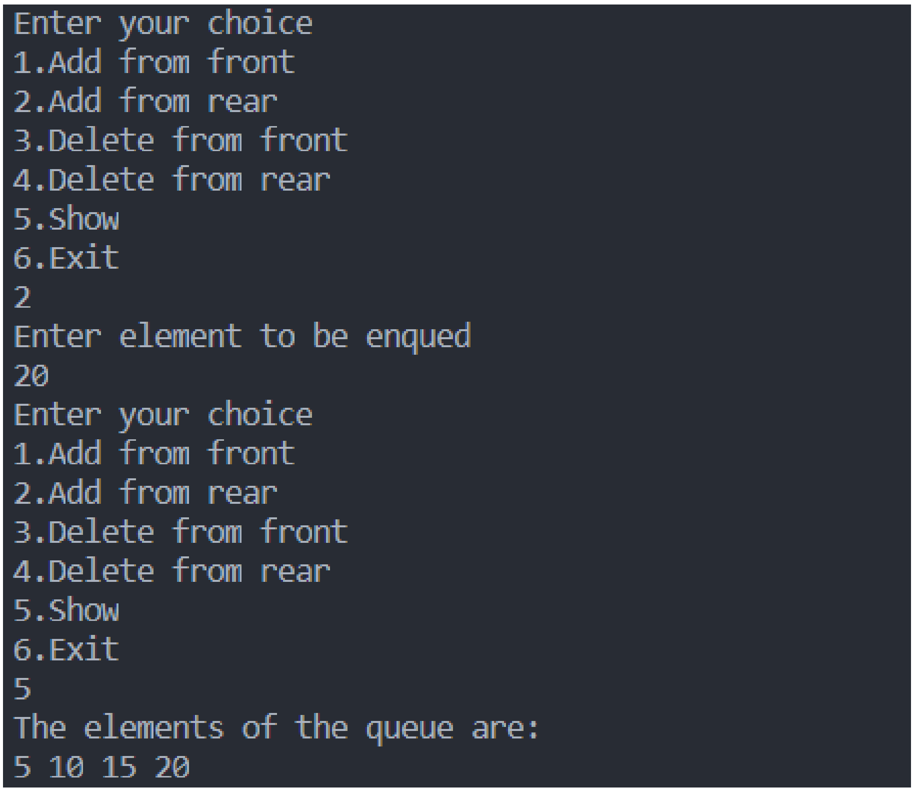
```
    }
    else
    {
        f++;
    }
}
void enqu_rear()
{
    if ((r == n - 1) || f == r + 1)
    {
        printf("Queue is Full\n");
    }
    else if (f == -1 && r == -1)
    {
        f = 0;
        r = 0;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[r] = num;
    }
    if (r == n - 1)
    {
        r = 0;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[r] = num;
    }
    else
    {
        r++;
        printf("Enter element to be enqued\n");
        scanf("%d", &num);
        qu[r] = num;
    }
}
void dequ_rear()
{
    if (f == -1 && r == -1)
    {
        printf("Queue Underflow\n");
    }
    else if (f == r)
    {
        f = -1;
        r = -1;
    }
    else
    {
```



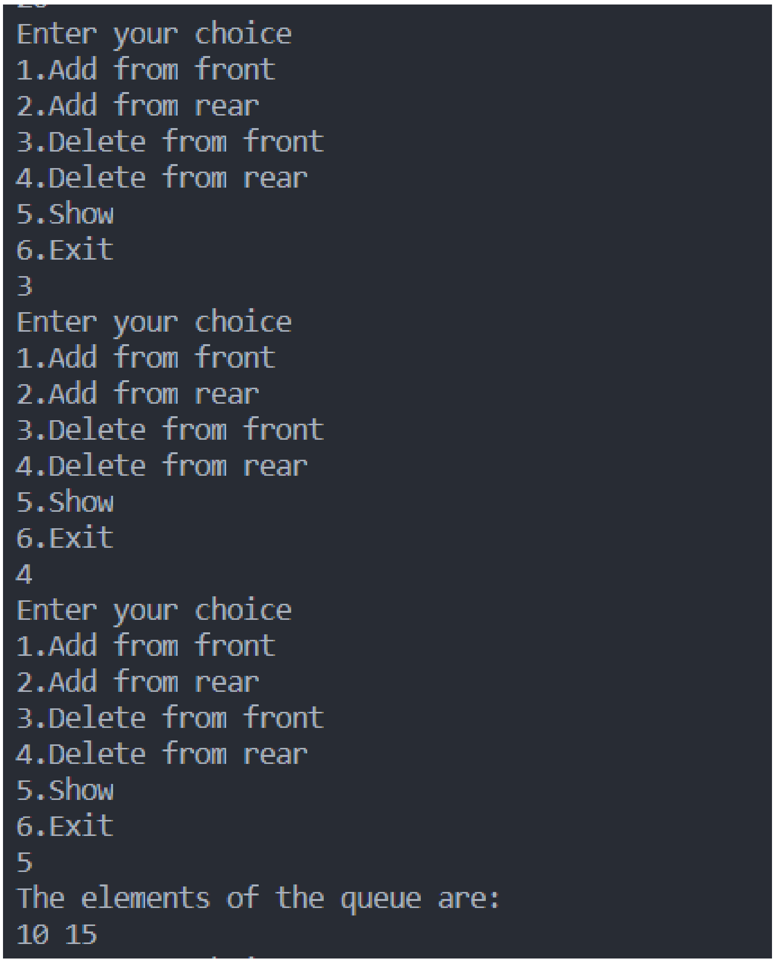
```
        r--;
    }
}
void show()
{

    {
        int i;
        if (f == r == -1)
            printf("\nQueue is Empty!!! Deletion is not possible!!!\n");
        else
        {
            printf("\nThe Queue elements are:");
            for (i = r; i < f; i++)
            {
                printf("%d\t ", qu[i]);
            }
        }
    }
}

int main()
{
    printf("Enter length of the queue\n");
    scanf("%d", &n);
    while (choice != 4)
    {
        printf("Enter your choice\n");
        printf("1.Add from front\n2.Add from rear\n3.Delete from front\n4.Dele
te from rear\n5.Show\n6.Exit\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            enqu_front();
            break;
        case 2:
            enqu_rear();
            break;
        case 3:
            dequ_front();
            break;
        case 4:
            dequ_rear();
            break;
        case 5:
            show();
            break;
        case 6:
            printf("Exiting...");
            break;
        default:
            printf("Invalid Input\n");
            break;
        }
    }
    return 1;
}
```



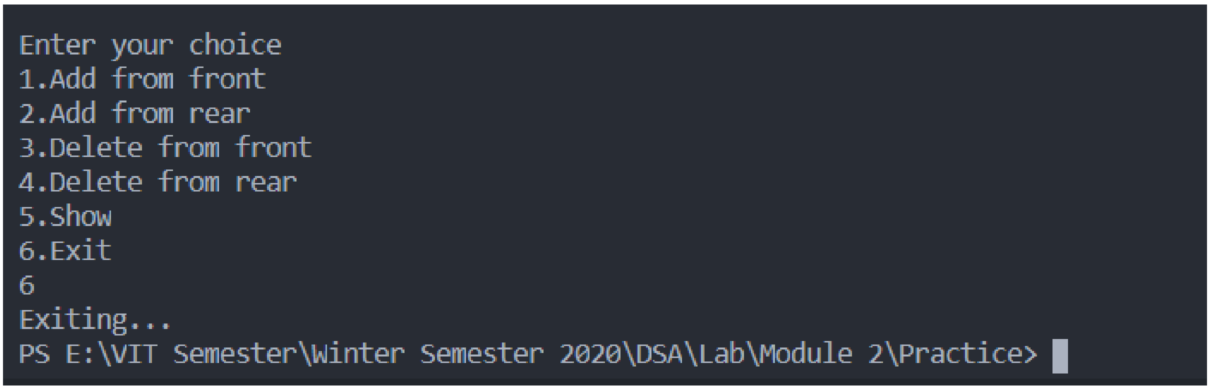
```
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2
.\deque }
Enter length of the queue
4
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
1
Enter element to be enqued
5
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
2
Enter element to be enqued
10
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
2
Enter element to be enqued
15
```



```
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
2
Enter element to be enqued
20
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
5
The elements of the queue are:
5 10 15 20
```

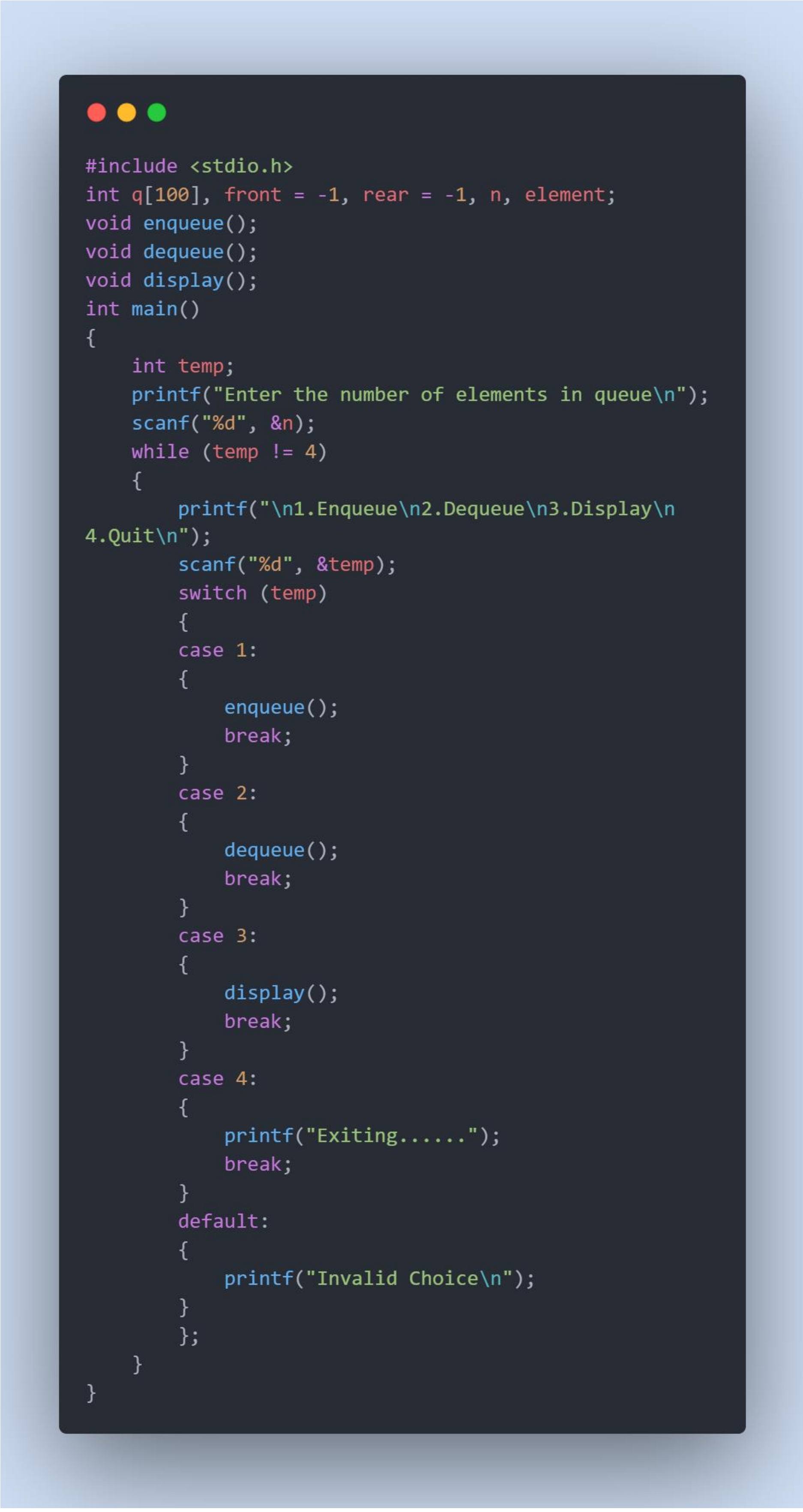


```
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
3
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
4
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
5
The elements of the queue are:
10 15
```

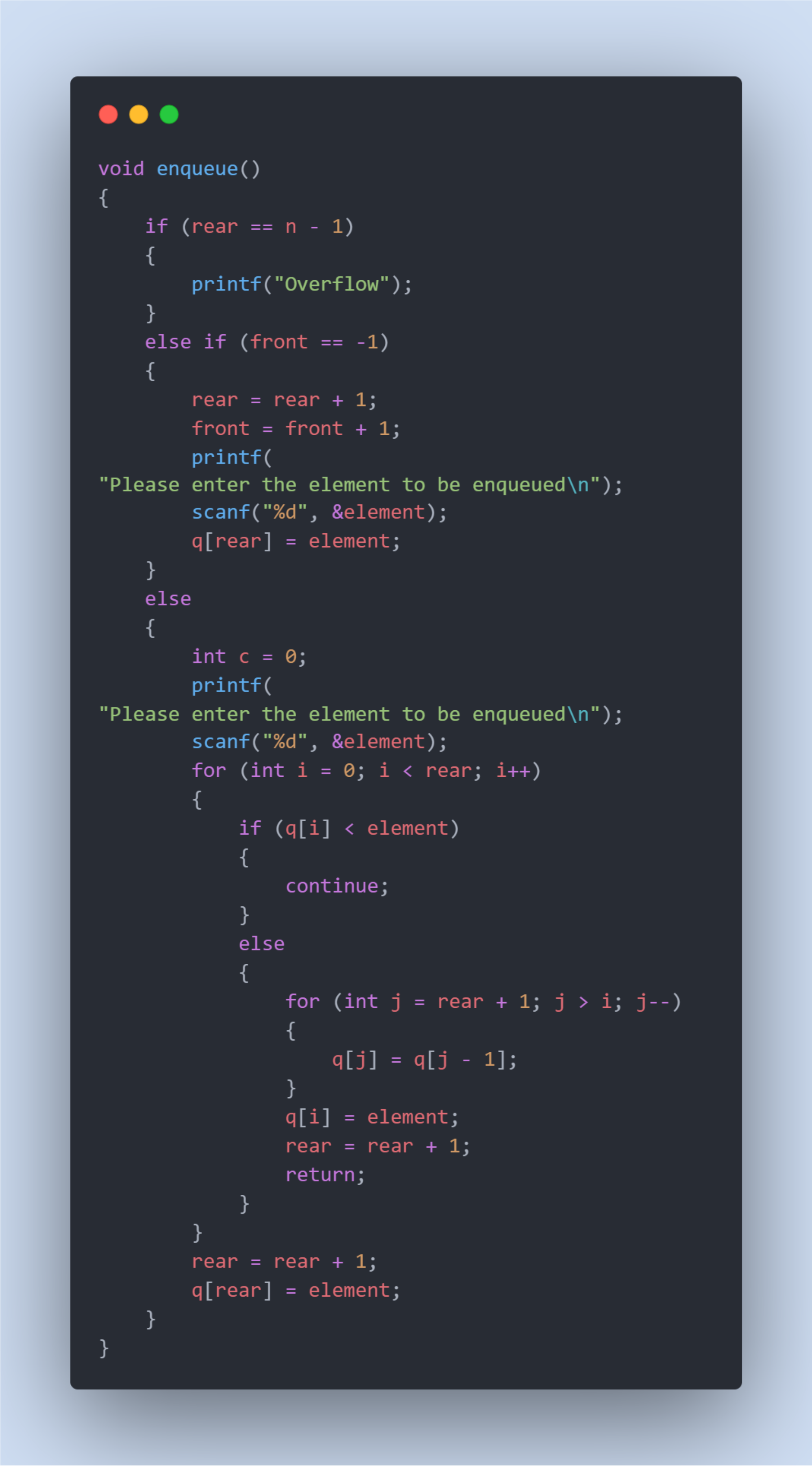


```
Enter your choice
1.Add from front
2.Add from rear
3.Delete from front
4.Delete from rear
5.Show
6.Exit
6
Exiting...
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2\Practice>
```

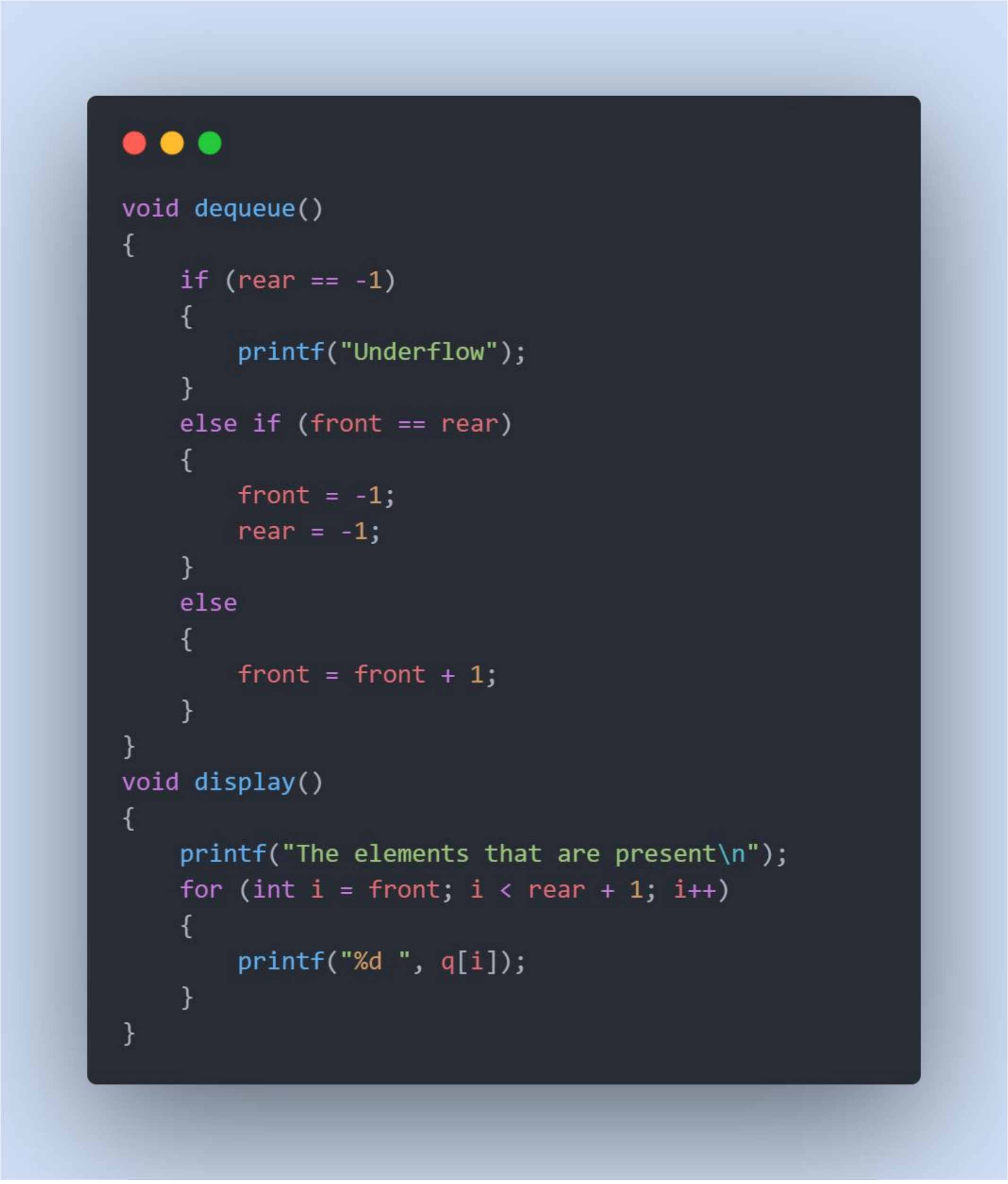
4. Develop a code to implement a priority queue such the least element is processed first.



```c
#include <stdio.h>
int q[100], front = -1, rear = -1, n, element;
void enqueue();
void dequeue();
void display();
int main()
{
    int temp;
    printf("Enter the number of elements in queue\n");
    scanf("%d", &n);
    while (temp != 4)
    {
        printf("\n1.Enqueue\n2.Dequeue\n3.Display\n
4.Quit\n");
        scanf("%d", &temp);
        switch (temp)
        {
        case 1:
        {
            enqueue();
            break;
        }
        case 2:
        {
            dequeue();
            break;
        }
        case 3:
        {
            display();
            break;
        }
        case 4:
        {
            printf("Exiting......");
            break;
        }
        default:
        {
            printf("Invalid Choice\n");
        }
        };
    }
}
```



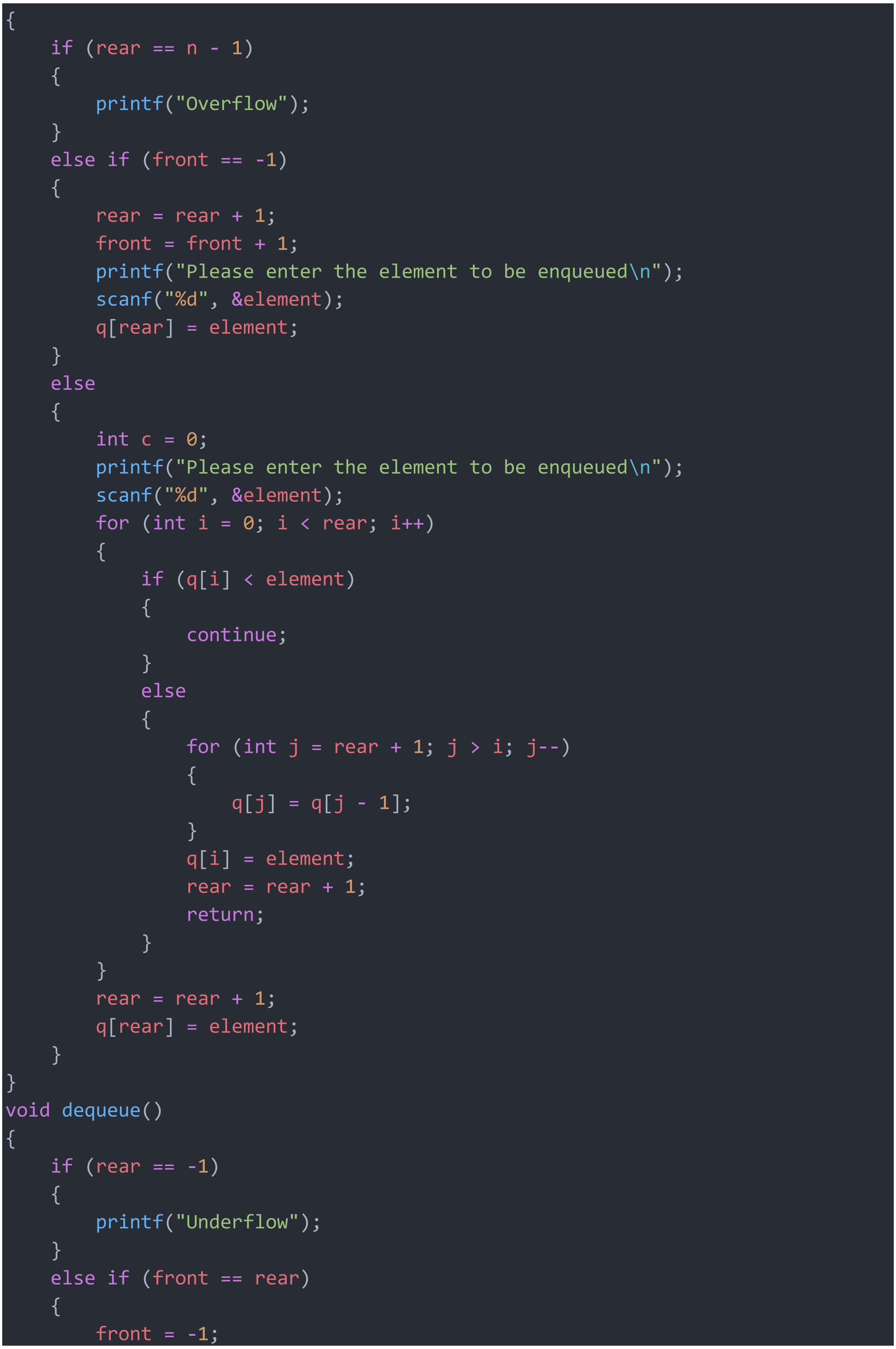
```c
void enqueue()
{
    if (rear == n - 1)
    {
        printf("Overflow");
    }
    else if (front == -1)
    {
        rear = rear + 1;
        front = front + 1;
        printf(
"Please enter the element to be enqueued\n");
        scanf("%d", &element);
        q[rear] = element;
    }
    else
    {
        int c = 0;
        printf(
"Please enter the element to be enqueued\n");
        scanf("%d", &element);
        for (int i = 0; i < rear; i++)
        {
            if (q[i] < element)
            {
                continue;
            }
            else
            {
                for (int j = rear + 1; j > i; j--)
                {
                    q[j] = q[j - 1];
                }
                q[i] = element;
                rear = rear + 1;
                return;
            }
        }
        rear = rear + 1;
        q[rear] = element;
    }
}
```



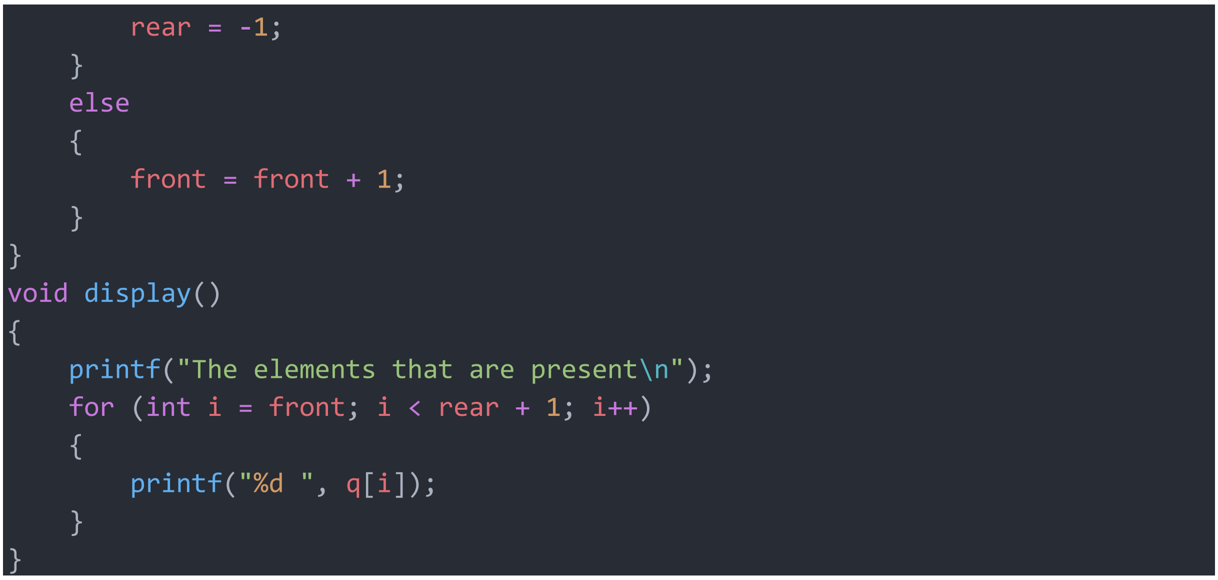
```
void dequeue()
{
    if (rear == -1)
    {
        printf("Underflow");
    }
    else if (front == rear)
    {
        front = -1;
        rear = -1;
    }
    else
    {
        front = front + 1;
    }
}
void display()
{
    printf("The elements that are present\n");
    for (int i = front; i < rear + 1; i++)
    {
        printf("%d ", q[i]);
    }
}
```

Complete Code

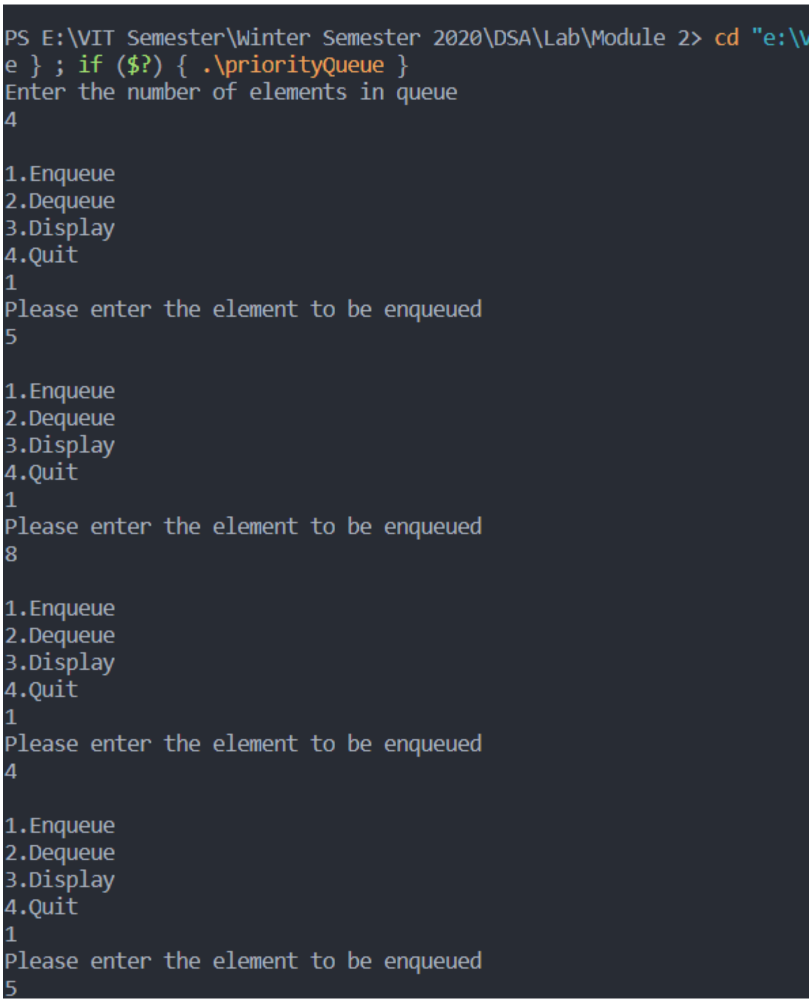
```c
#include <stdio.h>

int q[100], front = -1, rear = -1, n, element;
void enqueue();
void dequeue();
void display();
int main()
{
    int temp;
    printf("Enter the number of elements in queue\n");
    scanf("%d", &n);
    while (temp != 4)
    {
        printf("\n1.Enqueue\n2.Dequeue\n3.Display\n4.Quit\n");
        scanf("%d", &temp);
        switch (temp)
        {
        case 1:
        {
            enqueue();
            break;
        }
        case 2:
        {
            dequeue();
            break;
        }
        case 3:
        {
            display();
            break;
        }
        case 4:
        {
            printf("Exiting......");
            break;
        }
        default:
        {
            printf("Invalid Choice\n");
        }
        };
    }
}
void enqueue()
```



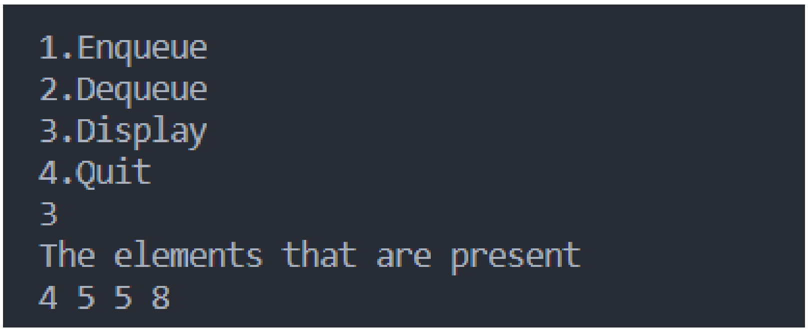
```
{
    if (rear == n - 1)
    {
        printf("Overflow");
    }
    else if (front == -1)
    {
        rear = rear + 1;
        front = front + 1;
        printf("Please enter the element to be enqueued\n");
        scanf("%d", &element);
        q[rear] = element;
    }
    else
    {
        int c = 0;
        printf("Please enter the element to be enqueued\n");
        scanf("%d", &element);
        for (int i = 0; i < rear; i++)
        {
            if (q[i] < element)
            {
                continue;
            }
            else
            {
                for (int j = rear + 1; j > i; j--)
                {
                    q[j] = q[j - 1];
                }
                q[i] = element;
                rear = rear + 1;
                return;
            }
        }
        rear = rear + 1;
        q[rear] = element;
    }
}
void dequeue()
{
    if (rear == -1)
    {
        printf("Underflow");
    }
    else if (front == rear)
    {
        front = -1;
```



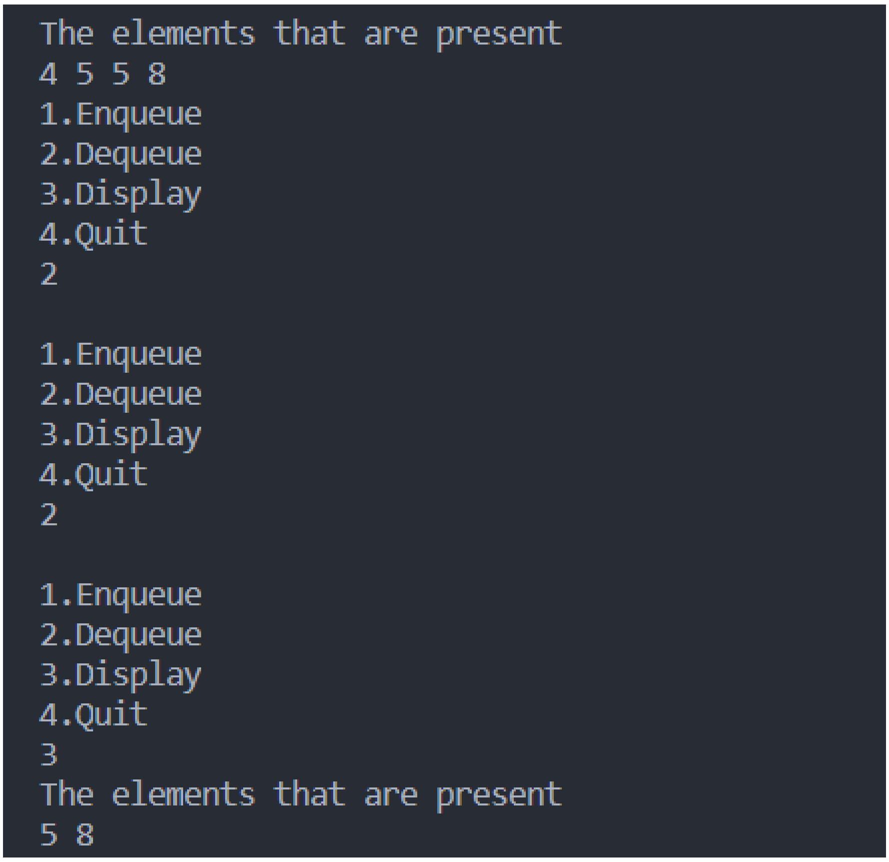
```
        rear = -1;
    }
    else
    {
        front = front + 1;
    }
}
void display()
{
    printf("The elements that are present\n");
    for (int i = front; i < rear + 1; i++)
    {
        printf("%d ", q[i]);
    }
}
```



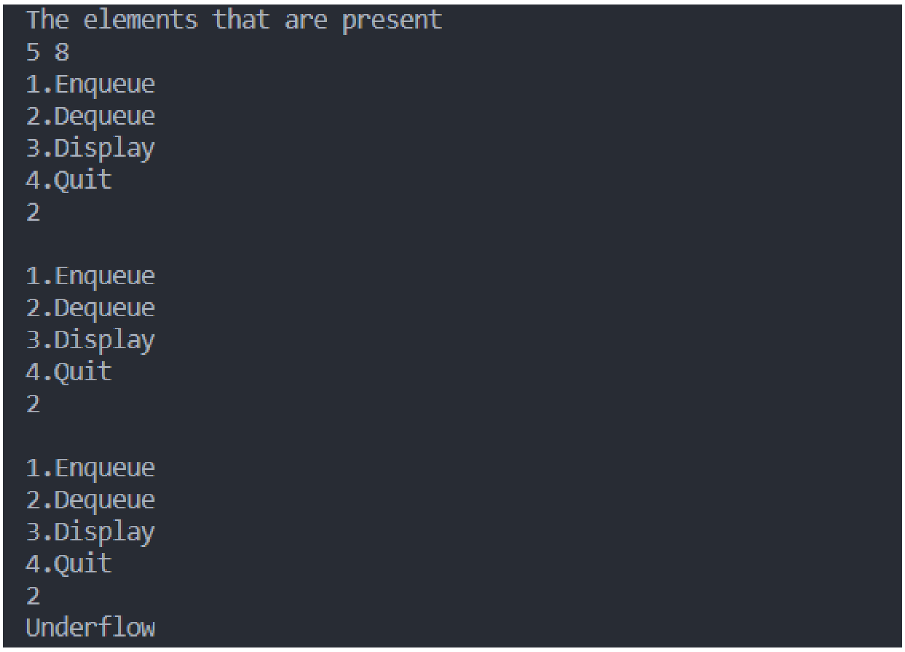
```
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2> cd "e:\V
e } ; if ($?) { .\priorityQueue }
Enter the number of elements in queue
4

1.Enqueue
2.Dequeue
3.Display
4.Quit
1
Please enter the element to be enqueued
5

1.Enqueue
2.Dequeue
3.Display
4.Quit
1
Please enter the element to be enqueued
8

1.Enqueue
2.Dequeue
3.Display
4.Quit
1
Please enter the element to be enqueued
4

1.Enqueue
2.Dequeue
3.Display
4.Quit
1
Please enter the element to be enqueued
5
```



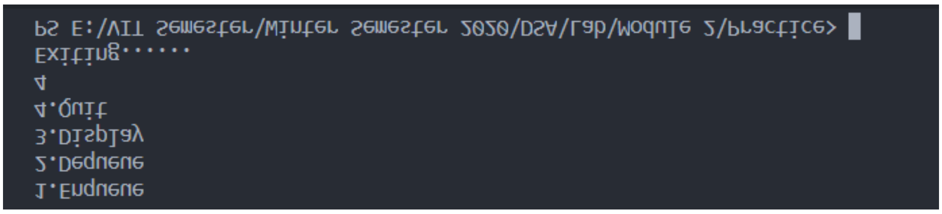
```
1.Enqueue
2.Dequeue
3.Display
4.Quit
3
The elements that are present
4 5 5 8
```



```
The elements that are present
4 5 5 8
1.Enqueue
2.Dequeue
3.Display
4.Quit
2

1.Enqueue
2.Dequeue
3.Display
4.Quit
2

1.Enqueue
2.Dequeue
3.Display
4.Quit
3
The elements that are present
5 8
```



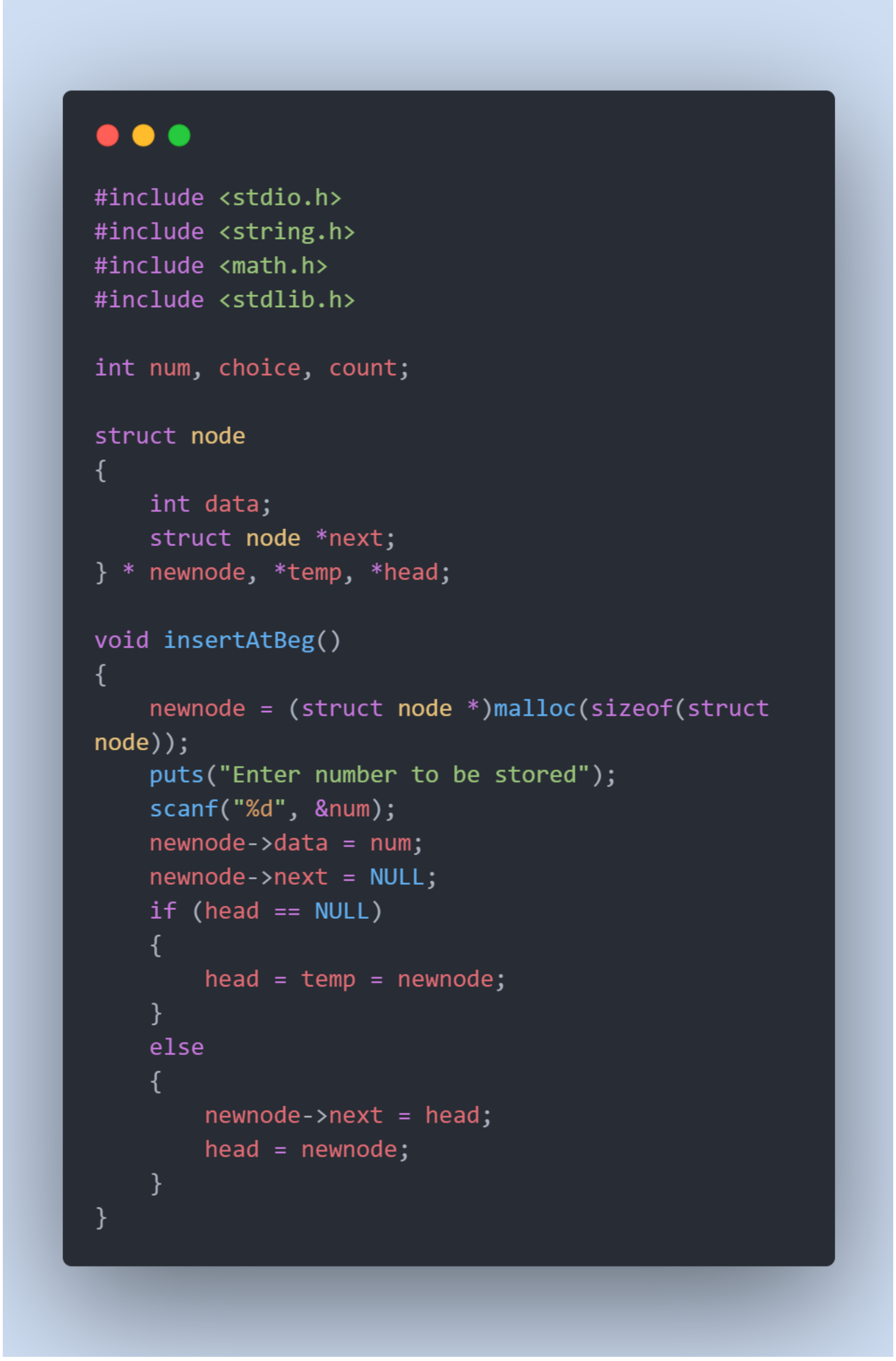
```
The elements that are present
5 8
1.Enqueue
2.Dequeue
3.Display
4.Quit
2

1.Enqueue
2.Dequeue
3.Display
4.Quit
2

1.Enqueue
2.Dequeue
3.Display
4.Quit
2
Underflow
```



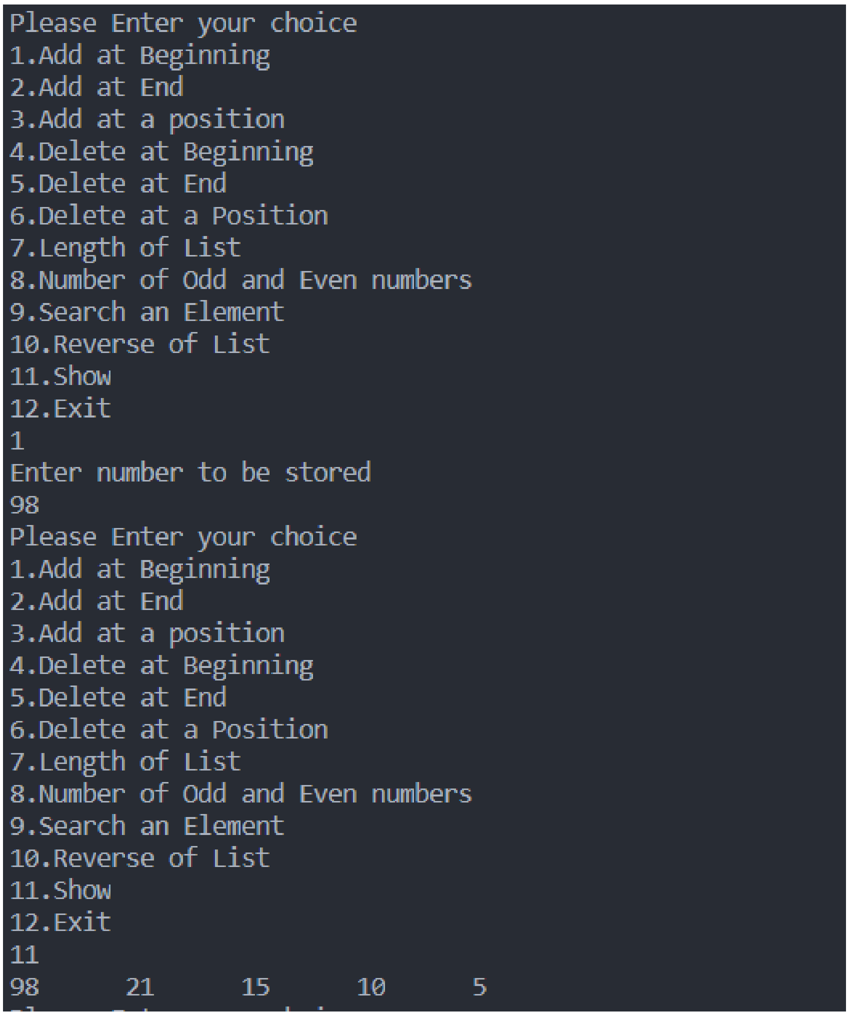
```
1.Enqueue
2.Dequeue
3.Display
4.Quit
4
Exiting......
PS E:/VII Semester/Winter Semester 2020/DSA/Lab/Module 2/Practice> █
```

5. Write a menu driven program to perform following functions in a singly linked list.

i) Insertion in the beginning of the list

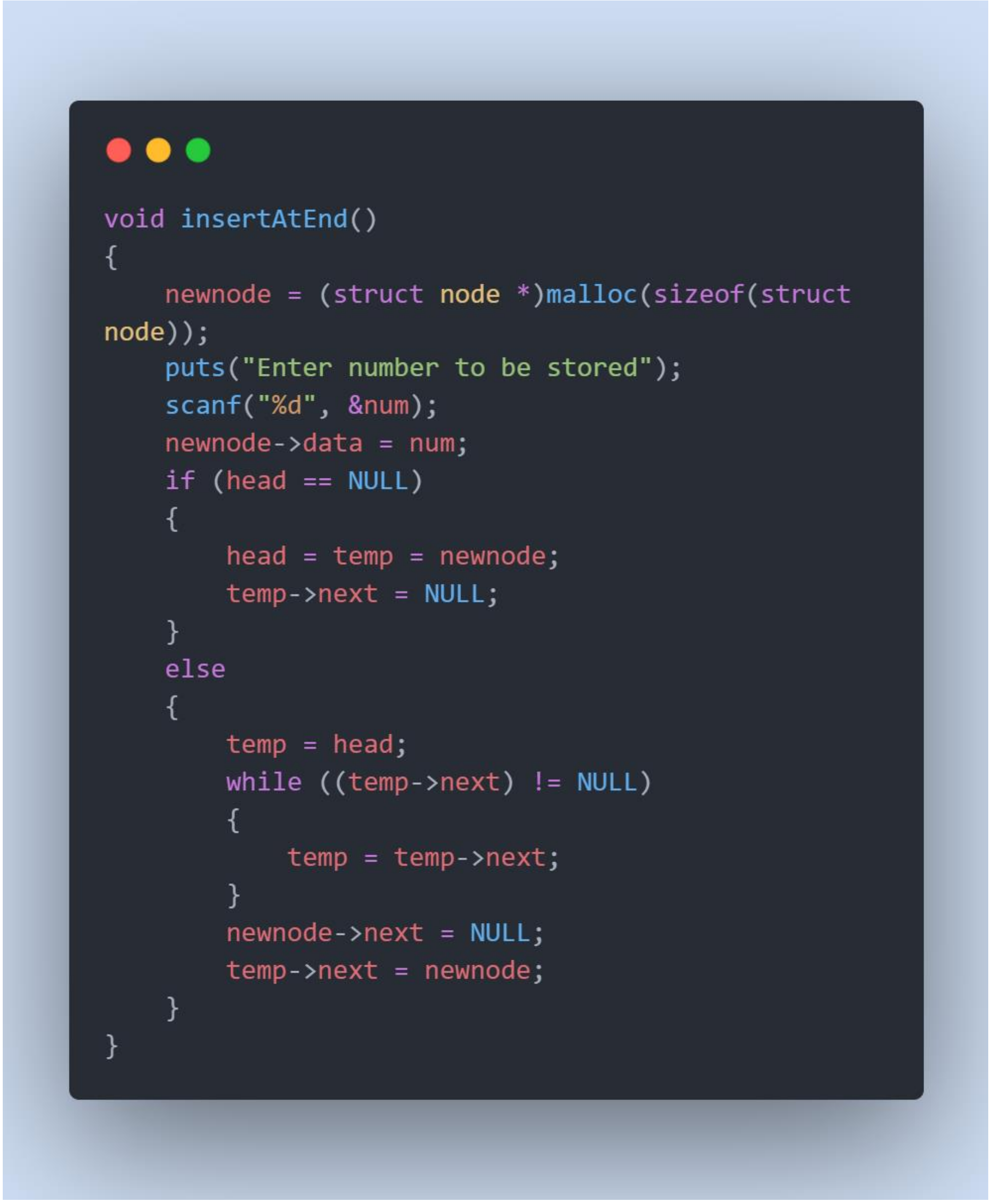


```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

int num, choice, count;

struct node
{
    int data;
    struct node *next;
} * newnode, *temp, *head;

void insertAtBeg()
{
    newnode = (struct node *)malloc(sizeof(struct
node));
    puts("Enter number to be stored");
    scanf("%d", &num);
    newnode->data = num;
    newnode->next = NULL;
    if (head == NULL)
    {
        head = temp = newnode;
    }
    else
    {
        newnode->next = head;
        head = newnode;
    }
}
```

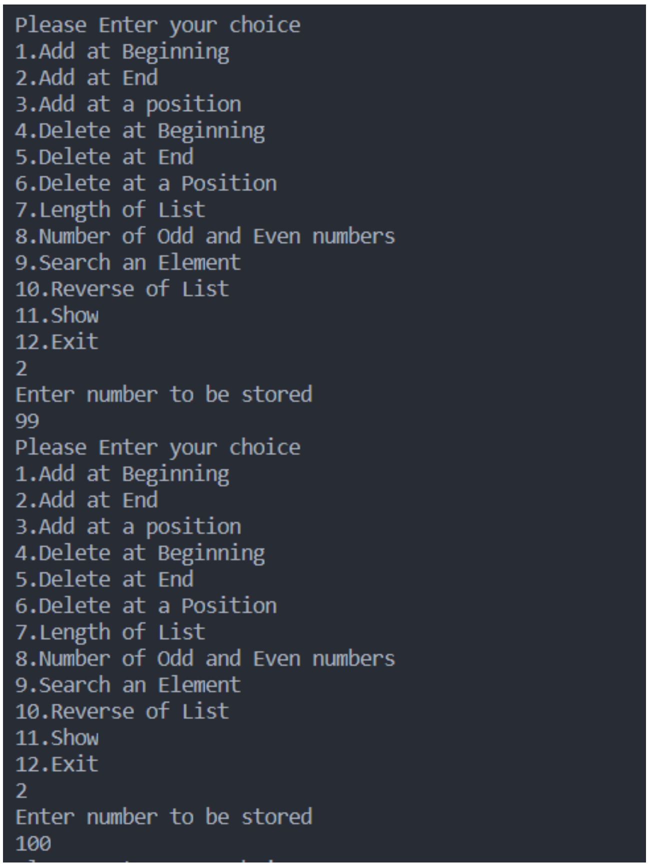


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
1
Enter number to be stored
98
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
98      21      15      10      5
```

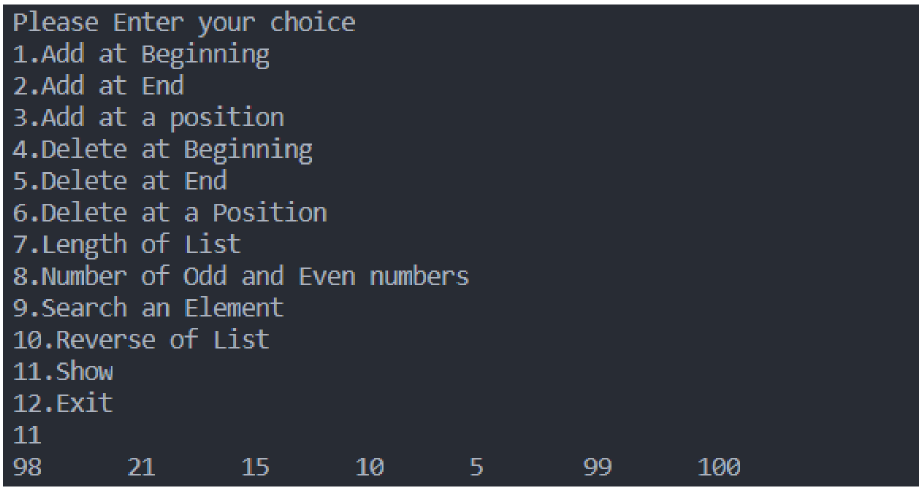
ii) Insertion at the end of the list



```
void insertAtEnd()
{
    newnode = (struct node *)malloc(sizeof(struct
node));
    puts("Enter number to be stored");
    scanf("%d", &num);
    newnode->data = num;
    if (head == NULL)
    {
        head = temp = newnode;
        temp->next = NULL;
    }
    else
    {
        temp = head;
        while ((temp->next) != NULL)
        {
            temp = temp->next;
        }
        newnode->next = NULL;
        temp->next = newnode;
    }
}
```

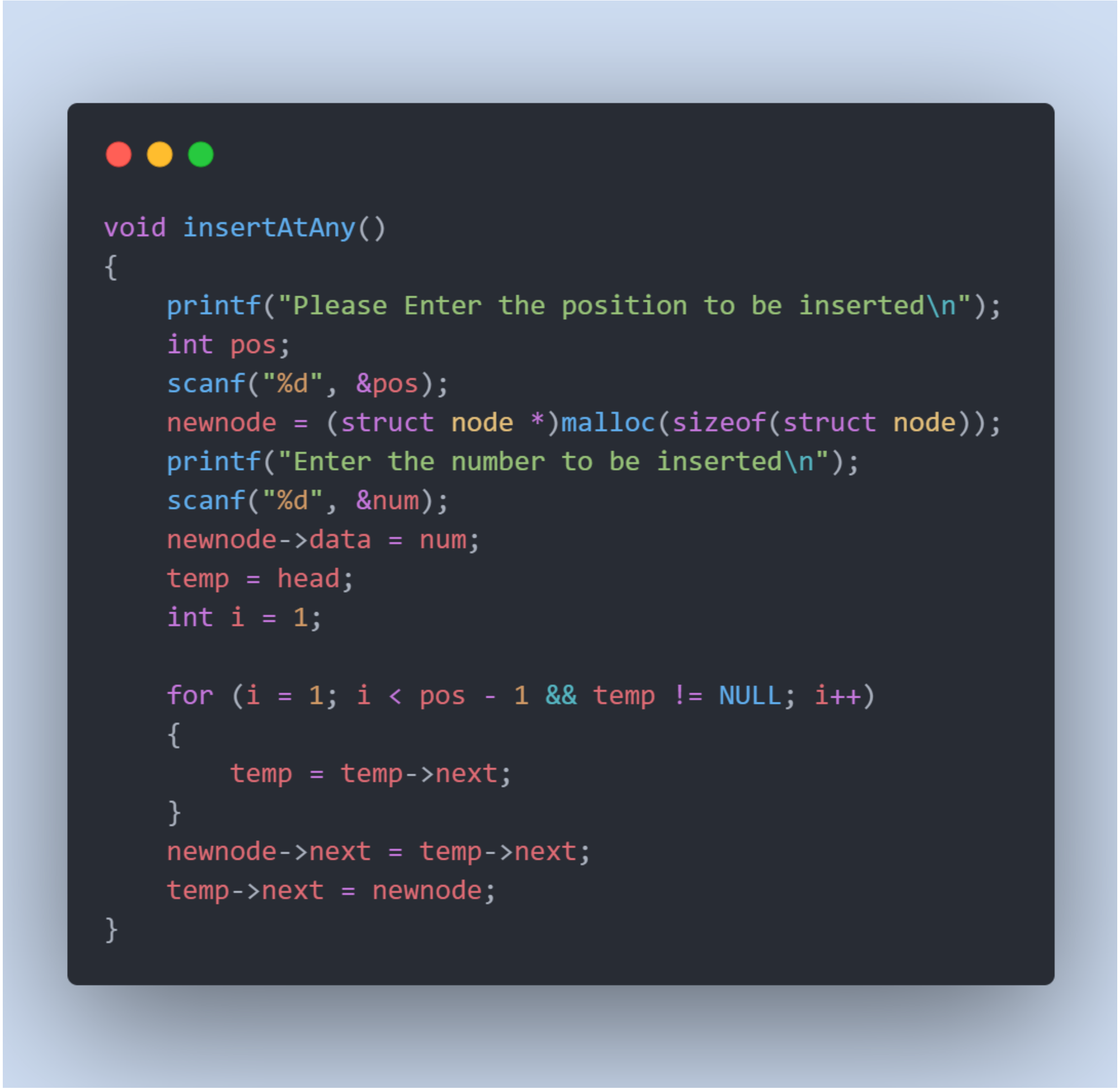


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
2
Enter number to be stored
99
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
2
Enter number to be stored
100
```

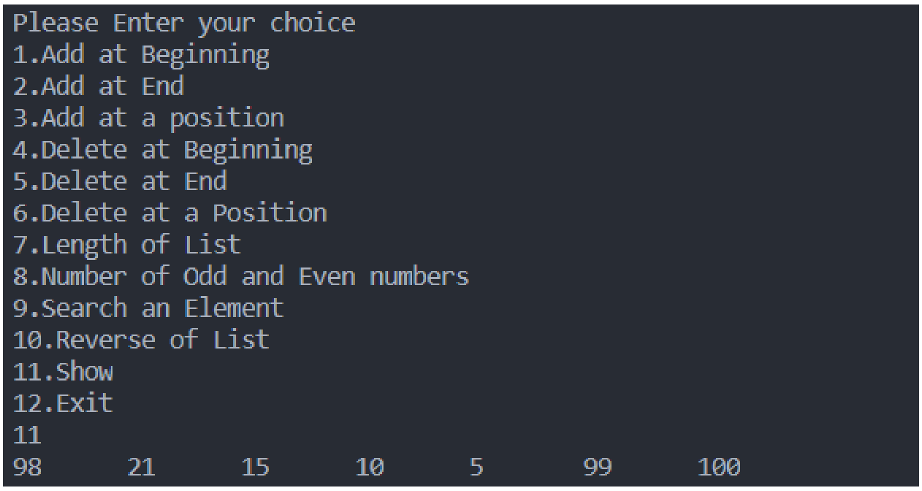


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
98      21      15      10      5       99      100
```

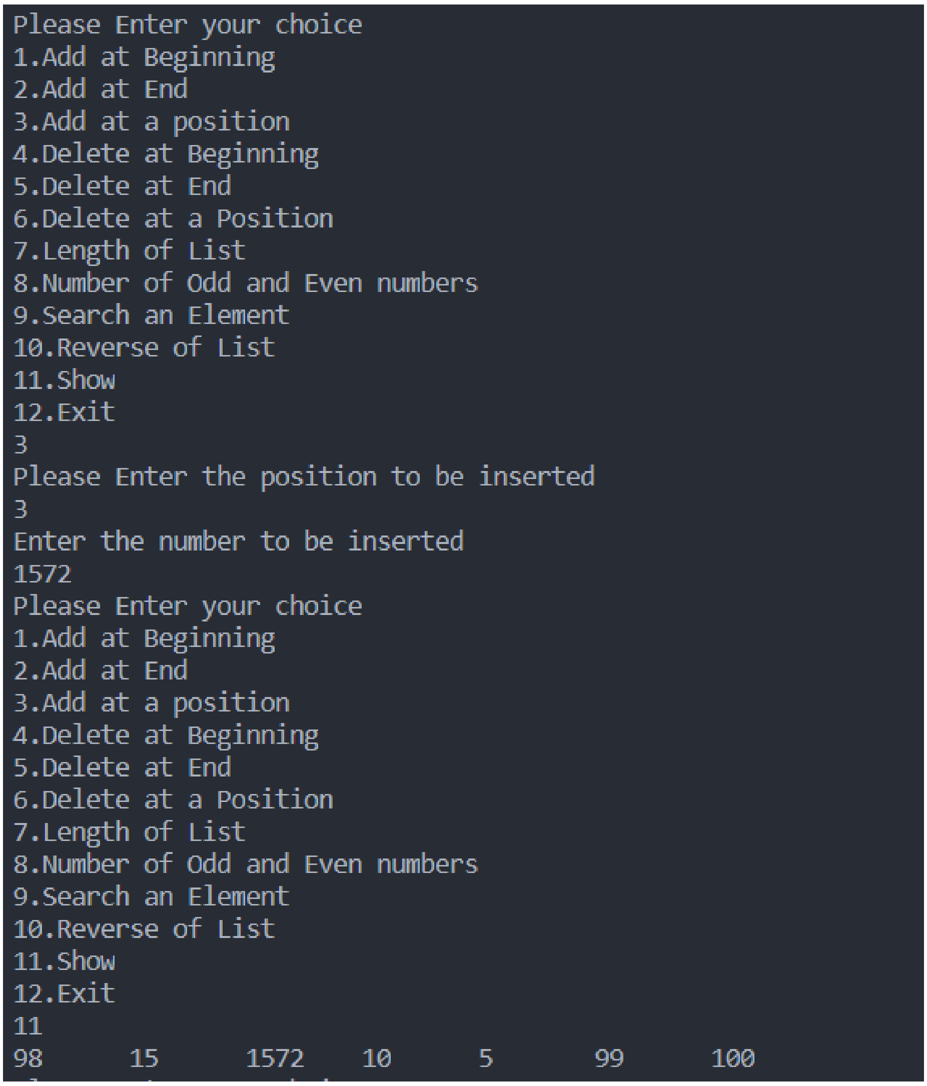
iii) Insertion in a particular location of the list



```
void insertAtAny()
{
    printf("Please Enter the position to be inserted\n");
    int pos;
    scanf("%d", &pos);
    newnode = (struct node *)malloc(sizeof(struct node));
    printf("Enter the number to be inserted\n");
    scanf("%d", &num);
    newnode->data = num;
    temp = head;
    int i = 1;

    for (i = 1; i < pos - 1 && temp != NULL; i++)
    {
        temp = temp->next;
    }
    newnode->next = temp->next;
    temp->next = newnode;
}
```

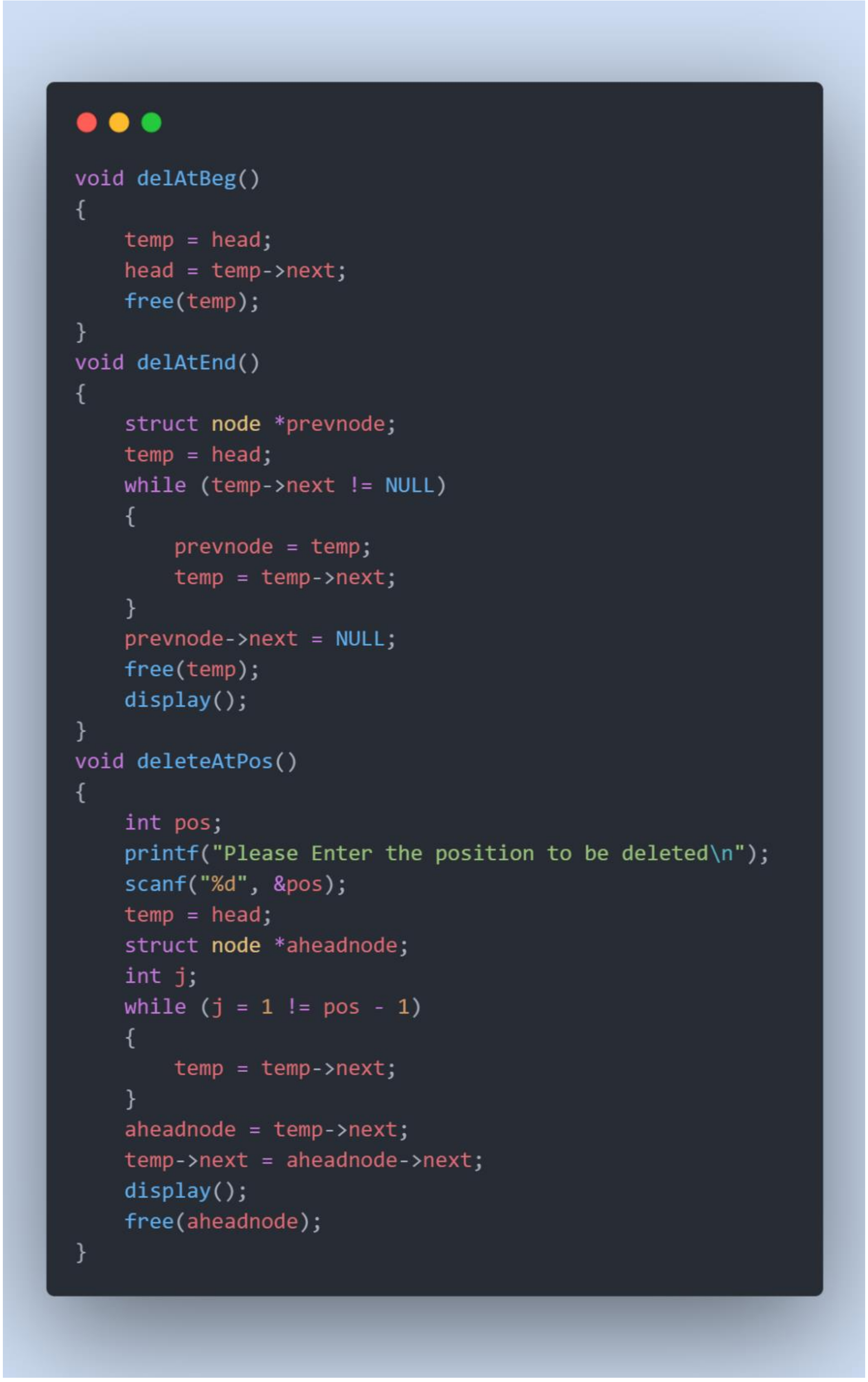


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
98      21      15      10      5       99      100
```

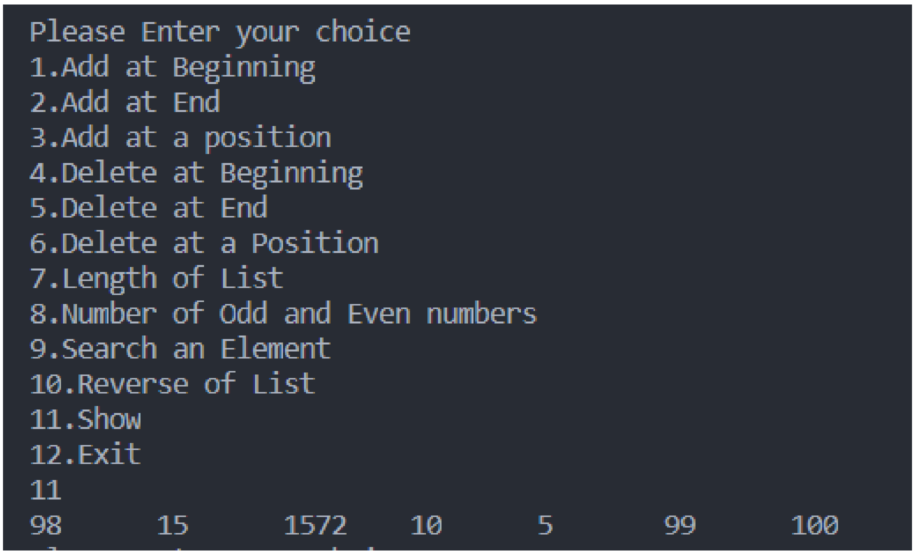


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
3
Please Enter the position to be inserted
3
Enter the number to be inserted
1572
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
98      15      1572    10      5       99      100
```

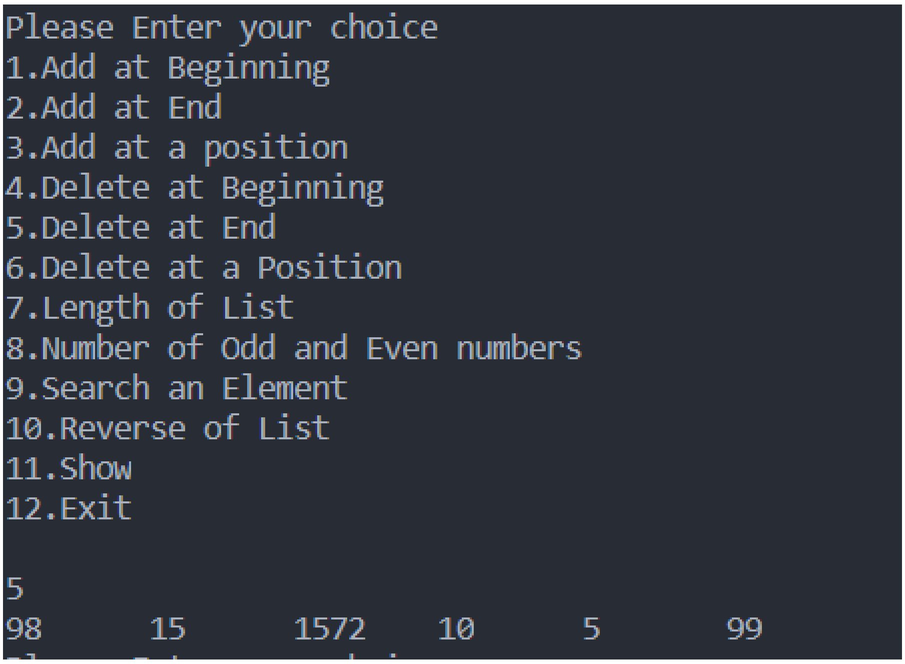
iv) Deletion based on a particular value and location



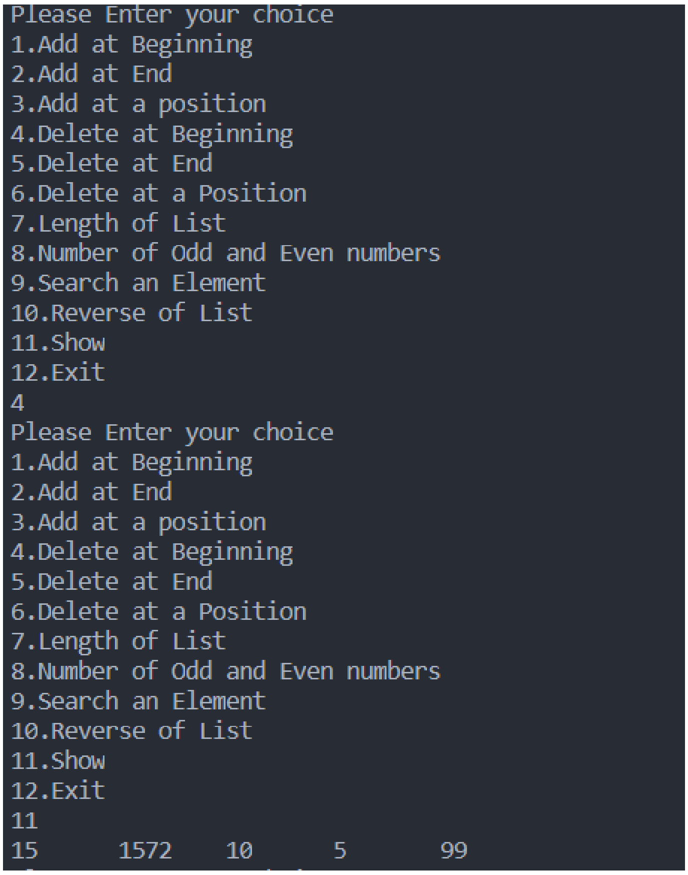
```
void delAtBeg()
{
    temp = head;
    head = temp->next;
    free(temp);
}
void delAtEnd()
{
    struct node *prevnode;
    temp = head;
    while (temp->next != NULL)
    {
        prevnode = temp;
        temp = temp->next;
    }
    prevnode->next = NULL;
    free(temp);
    display();
}
void deleteAtPos()
{
    int pos;
    printf("Please Enter the position to be deleted\n");
    scanf("%d", &pos);
    temp = head;
    struct node *aheadnode;
    int j;
    while (j = 1 != pos - 1)
    {
        temp = temp->next;
    }
    aheadnode = temp->next;
    temp->next = aheadnode->next;
    display();
    free(aheadnode);
}
```



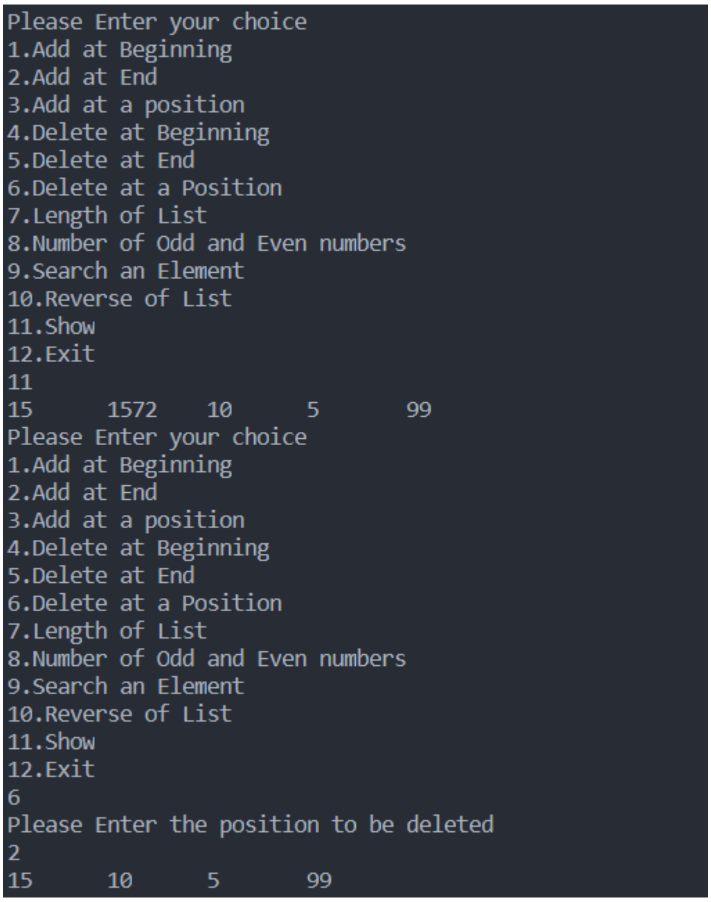
```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
98      15      1572    10      5       99      100
```



```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit

5
98      15      1572    10      5       99
```

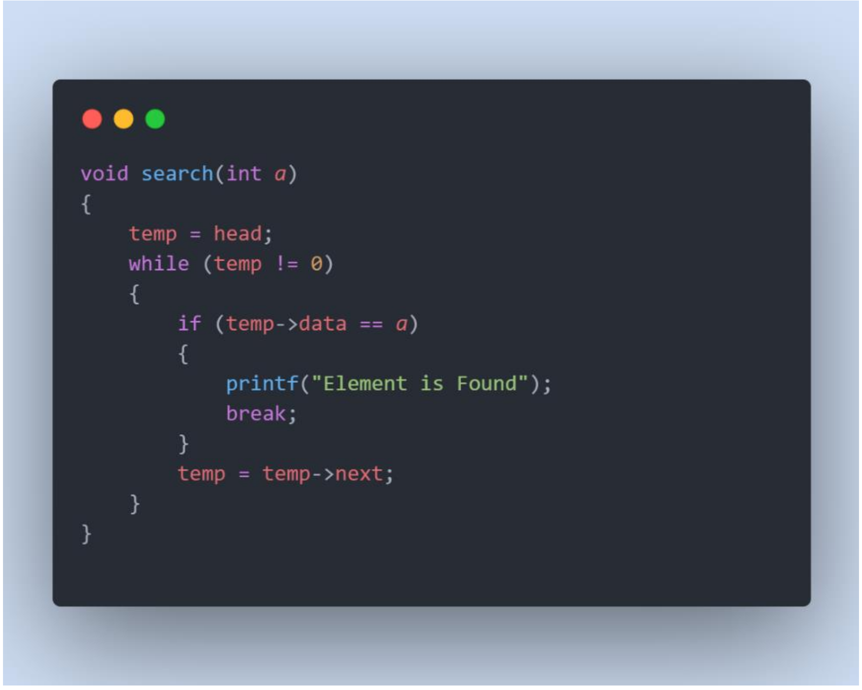


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
4
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
15      1572    10      5       99
```

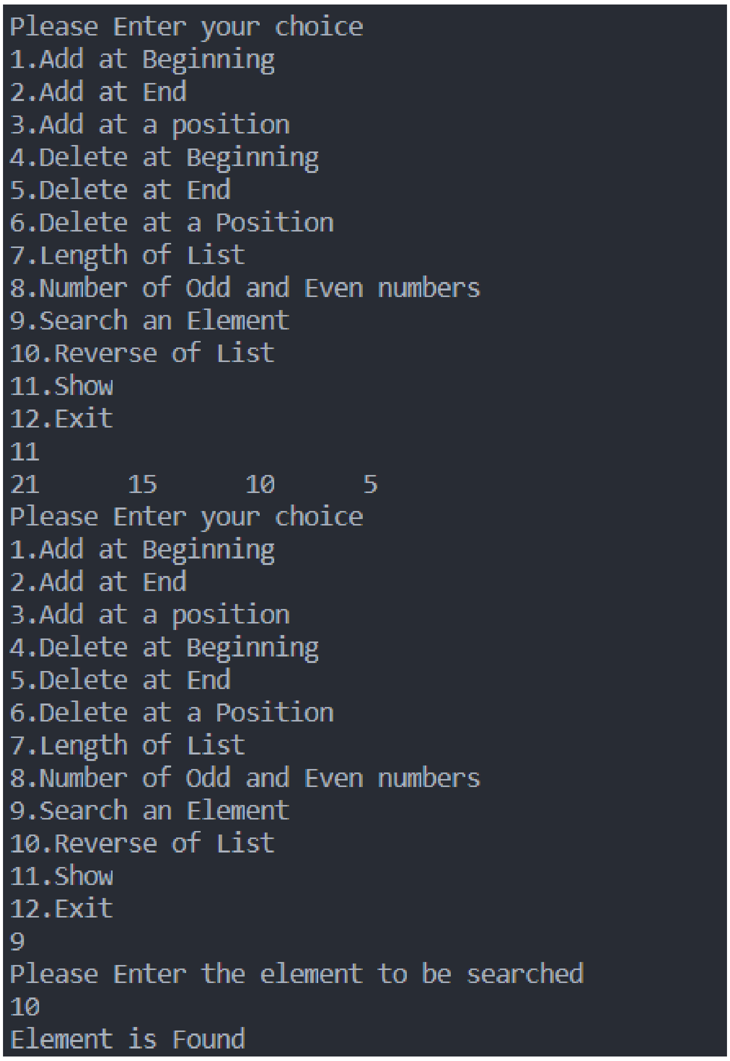


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
15      1572    10      5       99
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
6
Please Enter the position to be deleted
2
15      10      5       99
```

v) Search an element

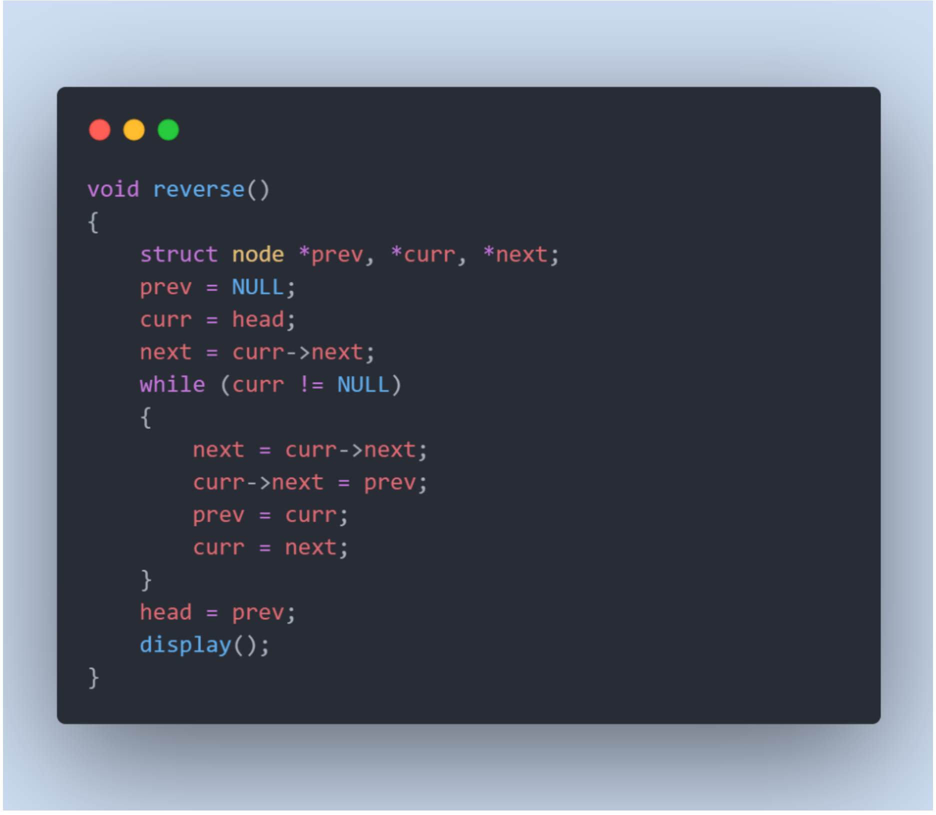


```c
void search(int a)
{
    temp = head;
    while (temp != 0)
    {
        if (temp->data == a)
        {
            printf("Element is Found");
            break;
        }
        temp = temp->next;
    }
}
```

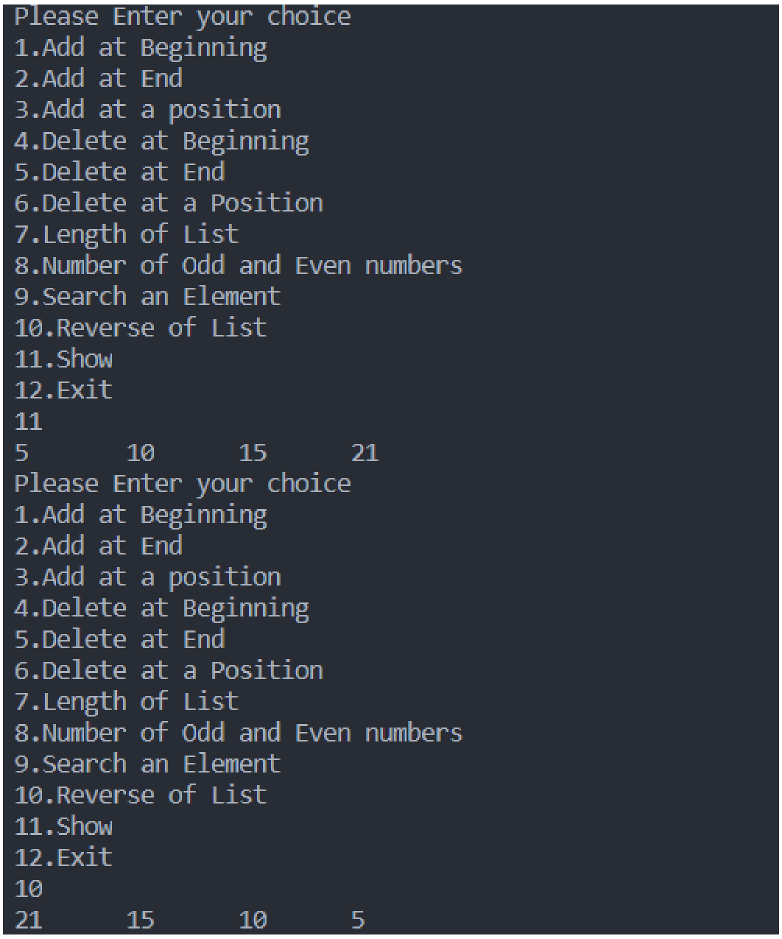


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
21      15      10      5
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
9
Please Enter the element to be searched
10
Element is Found
```

vi) Reverse the list

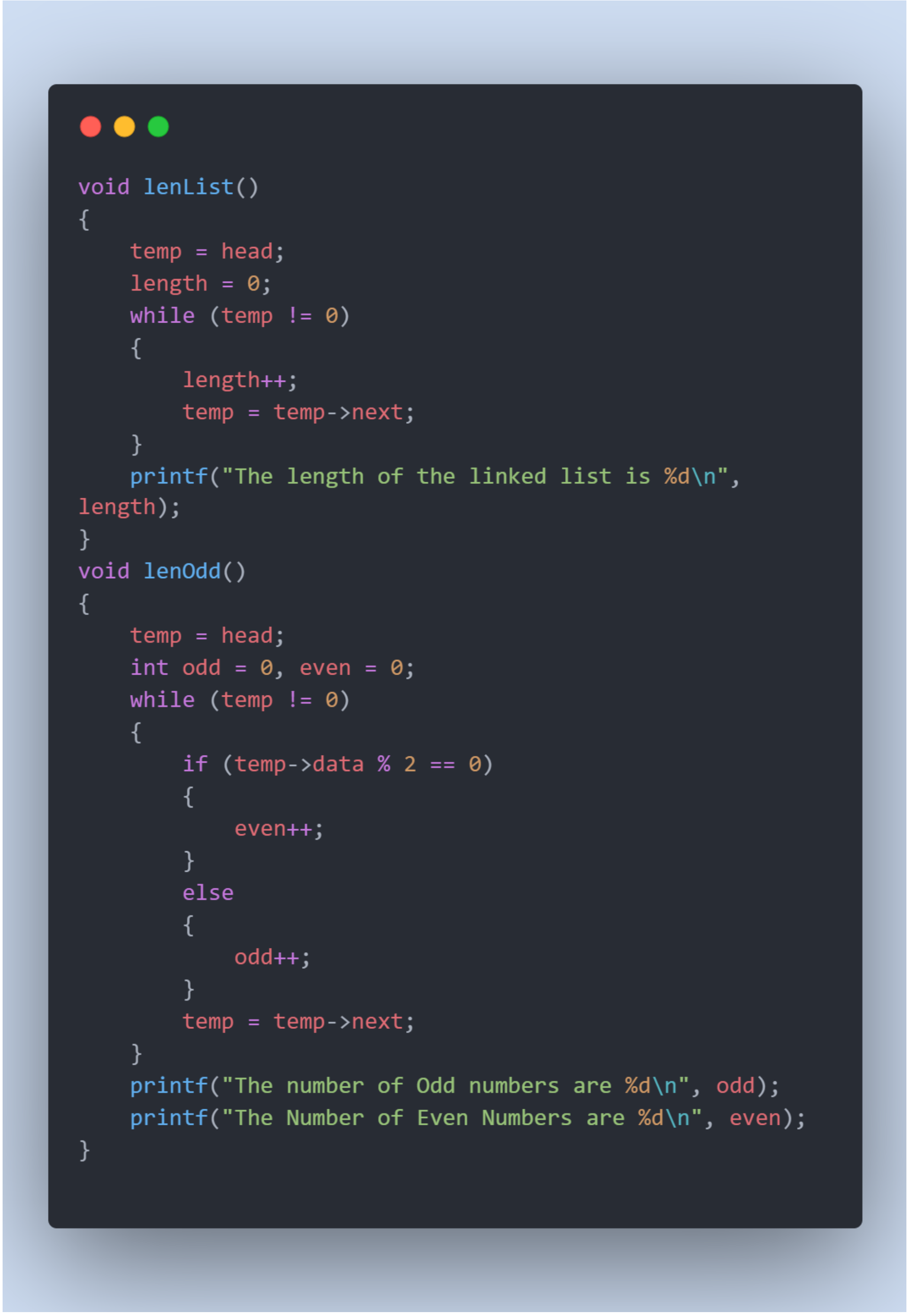


```
void reverse()
{
    struct node *prev, *curr, *next;
    prev = NULL;
    curr = head;
    next = curr->next;
    while (curr != NULL)
    {
        next = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next;
    }
    head = prev;
    display();
}
```

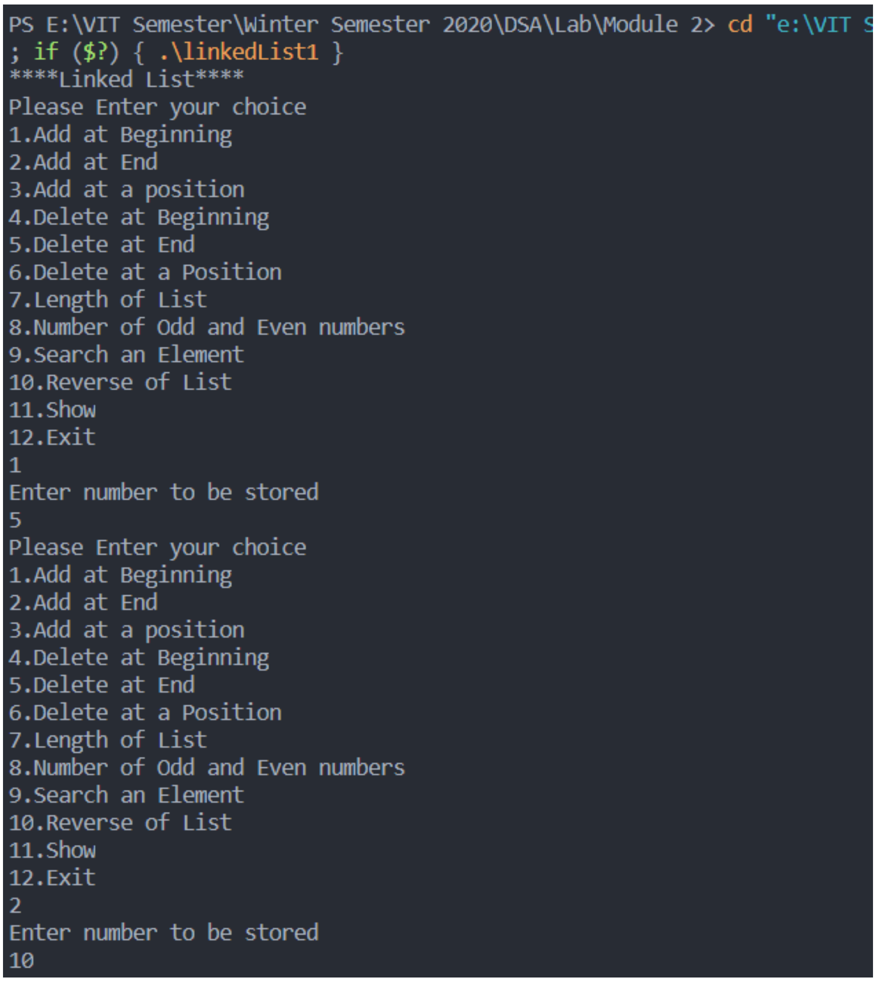


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
5       10      15      21
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
10
21      15      10      5
```

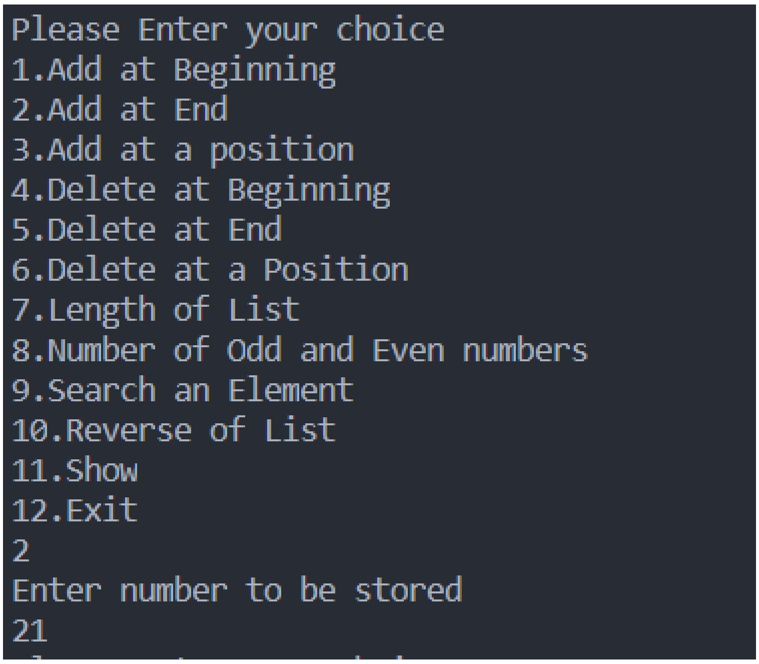
vii) Count the number of even and odd numbers in the list



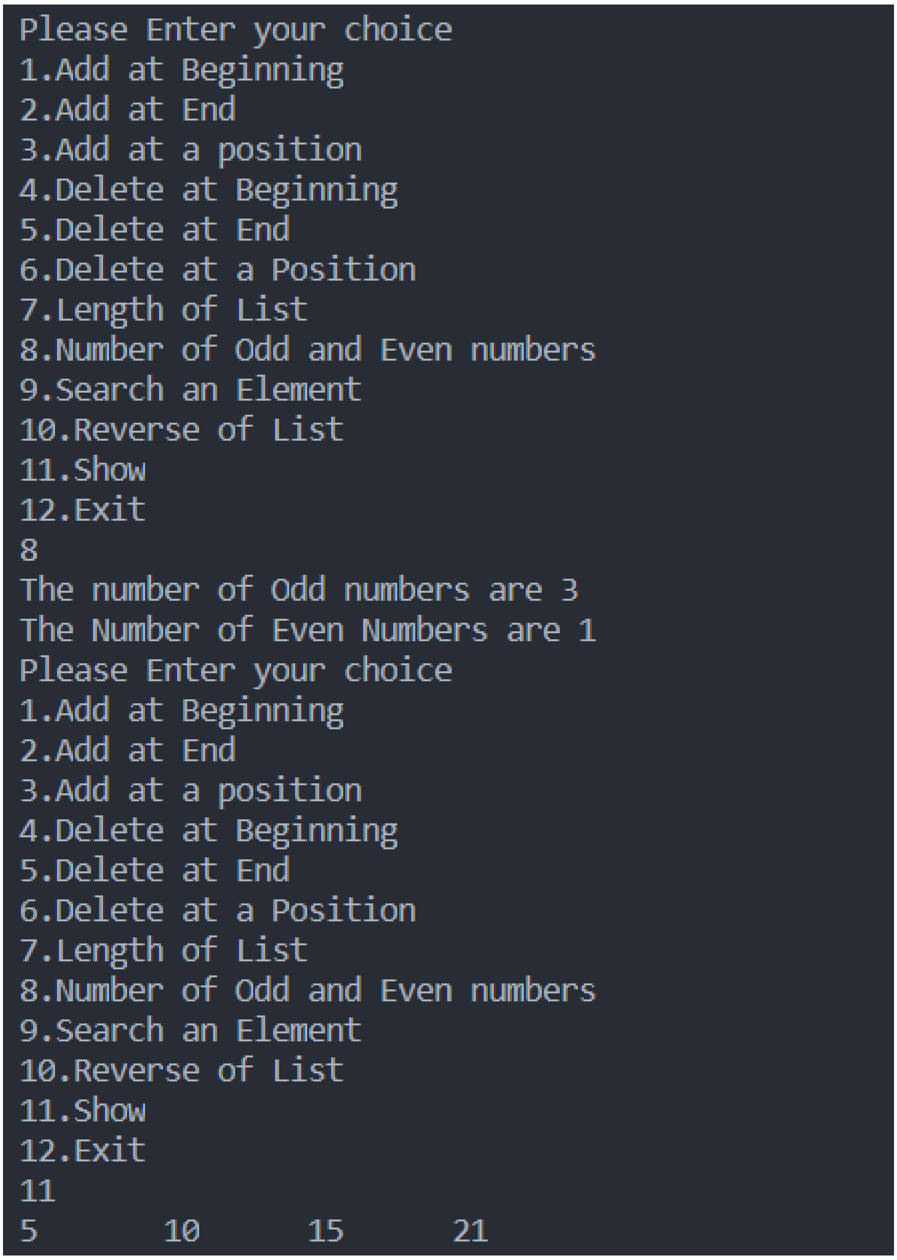
```
void lenList()
{
    temp = head;
    length = 0;
    while (temp != 0)
    {
        length++;
        temp = temp->next;
    }
    printf("The length of the linked list is %d\n",
length);
}
void lenOdd()
{
    temp = head;
    int odd = 0, even = 0;
    while (temp != 0)
    {
        if (temp->data % 2 == 0)
        {
            even++;
        }
        else
        {
            odd++;
        }
        temp = temp->next;
    }
    printf("The number of Odd numbers are %d\n", odd);
    printf("The Number of Even Numbers are %d\n", even);
}
```



```
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2> cd "e:\VIT
; if ($?) { .\linkedList1 }
****Linked List****
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
1
Enter number to be stored
5
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
2
Enter number to be stored
10
```

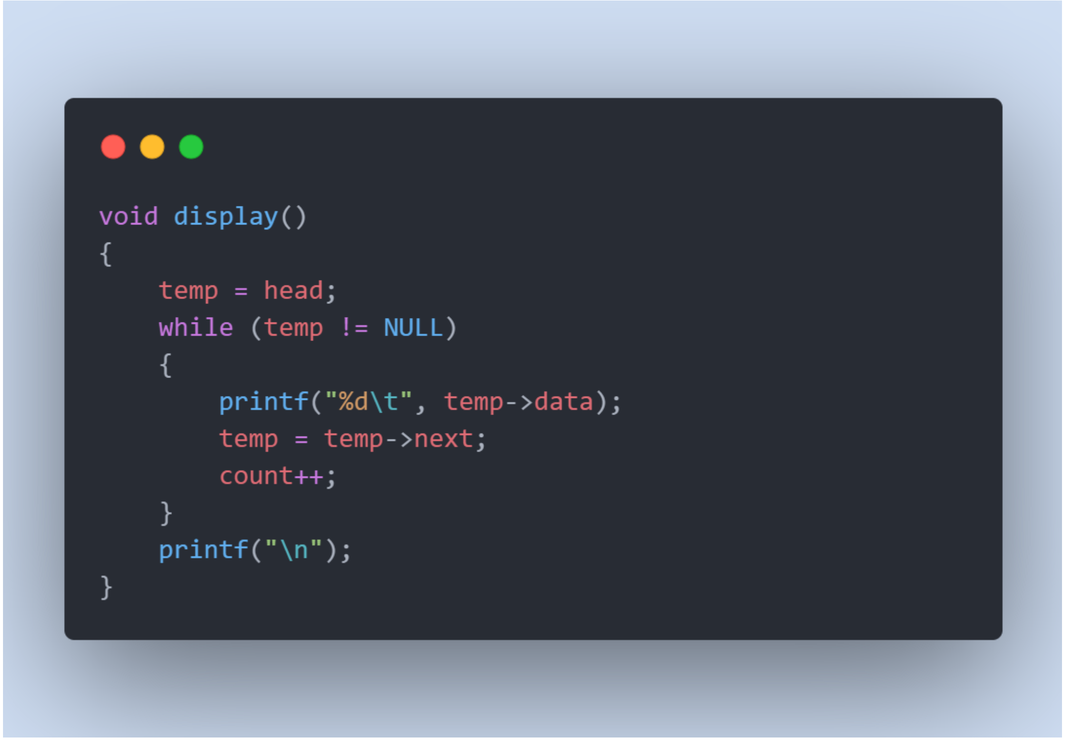


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
2
Enter number to be stored
21
```

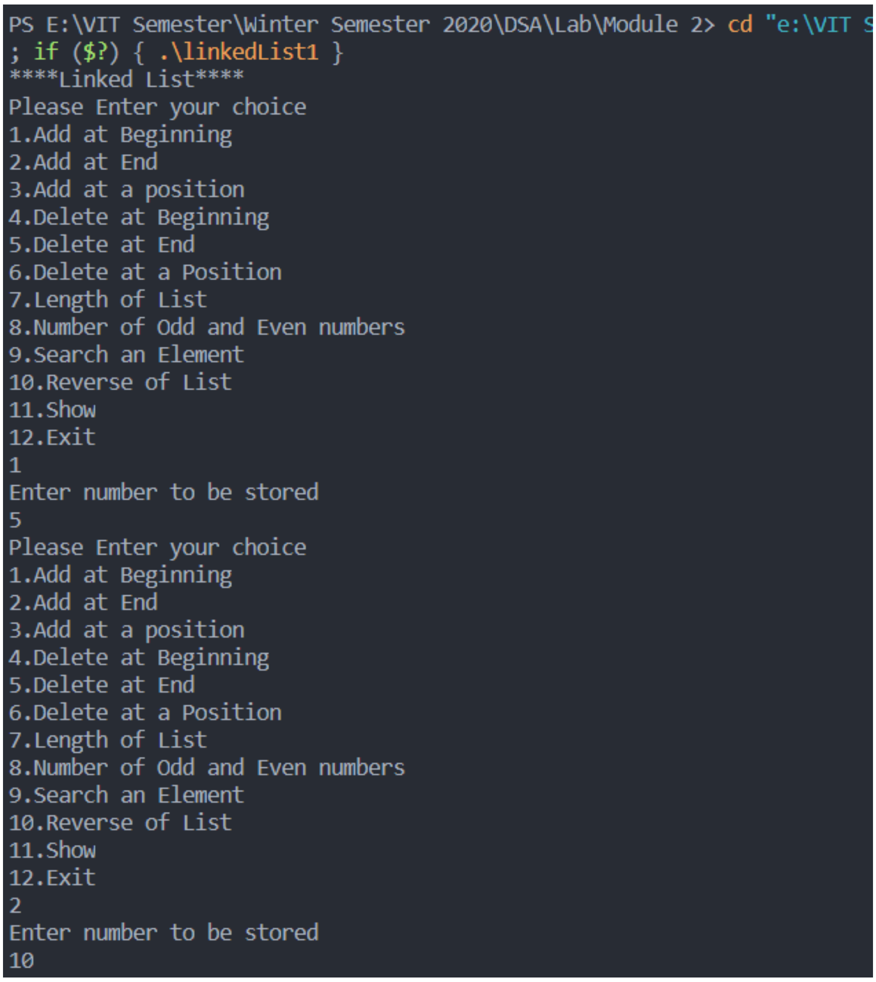


```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
8
The number of Odd numbers are 3
The Number of Even Numbers are 1
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
5       10      15      21
```

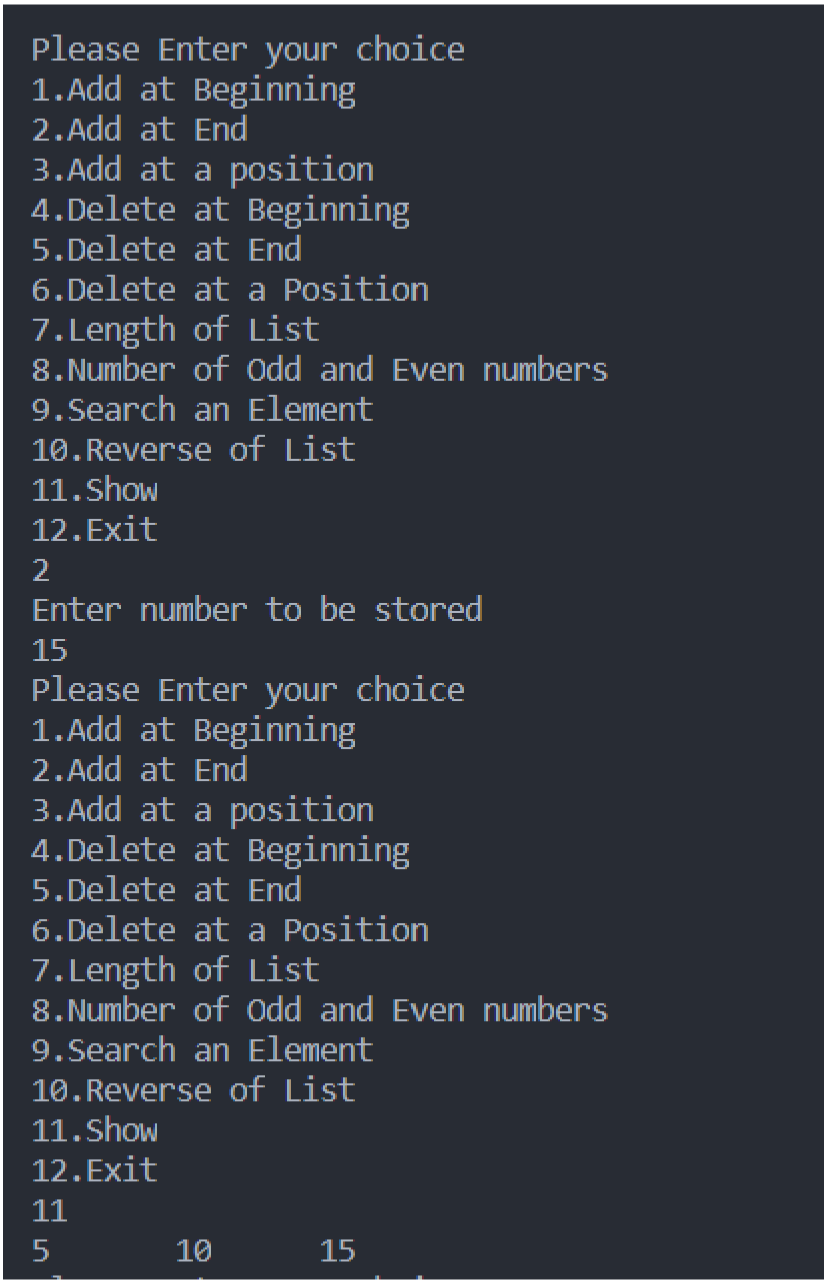
viii)Display Function



```
void display()
{
    temp = head;
    while (temp != NULL)
    {
        printf("%d\t", temp->data);
        temp = temp->next;
        count++;
    }
    printf("\n");
}
```



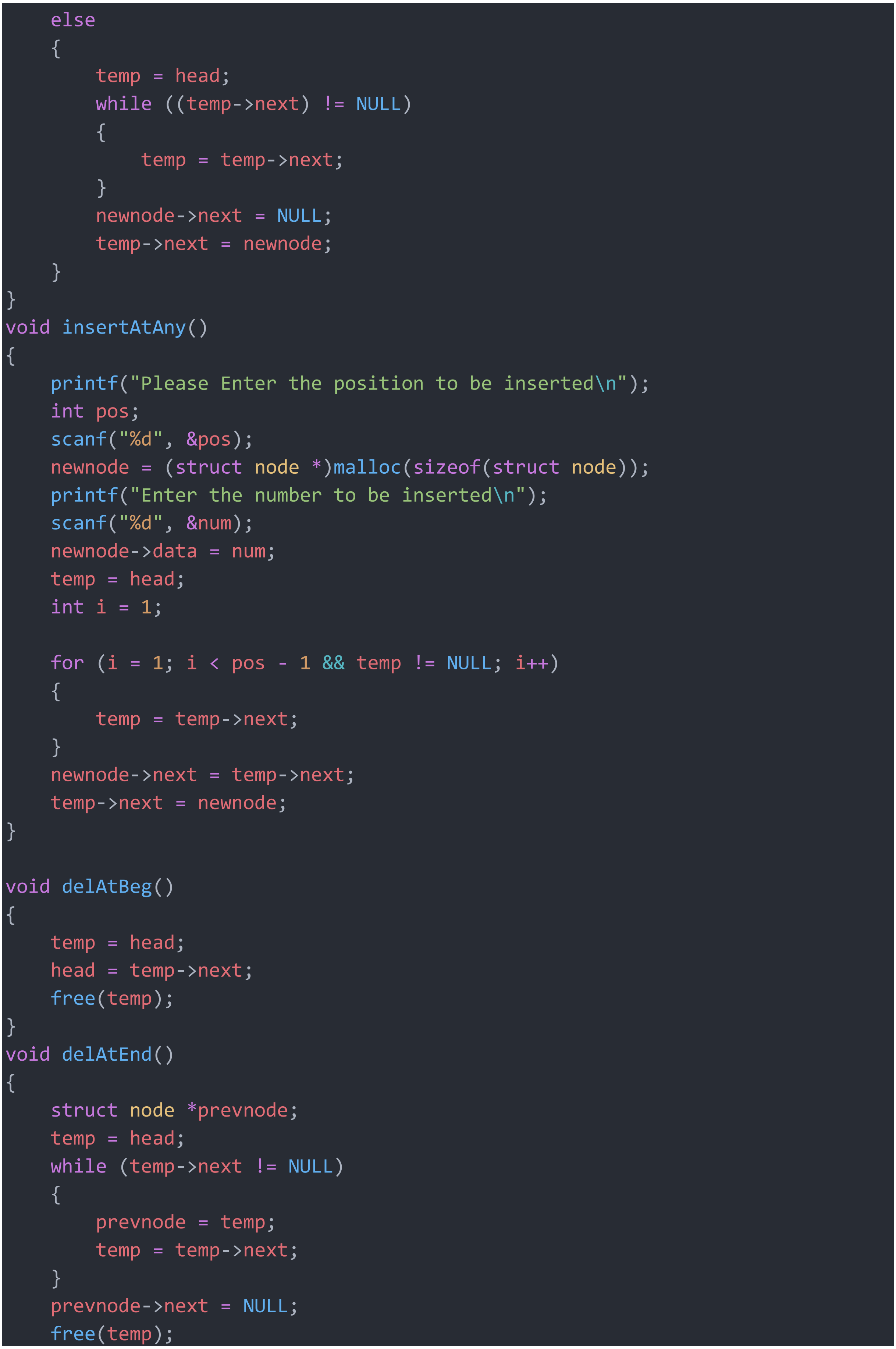
```
PS E:\VIT Semester\Winter Semester 2020\DSA\Lab\Module 2> cd "e:\VIT S
; if ($?) { .\linkedList1 }
****Linked List****
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
1
Enter number to be stored
5
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
2
Enter number to be stored
10
```



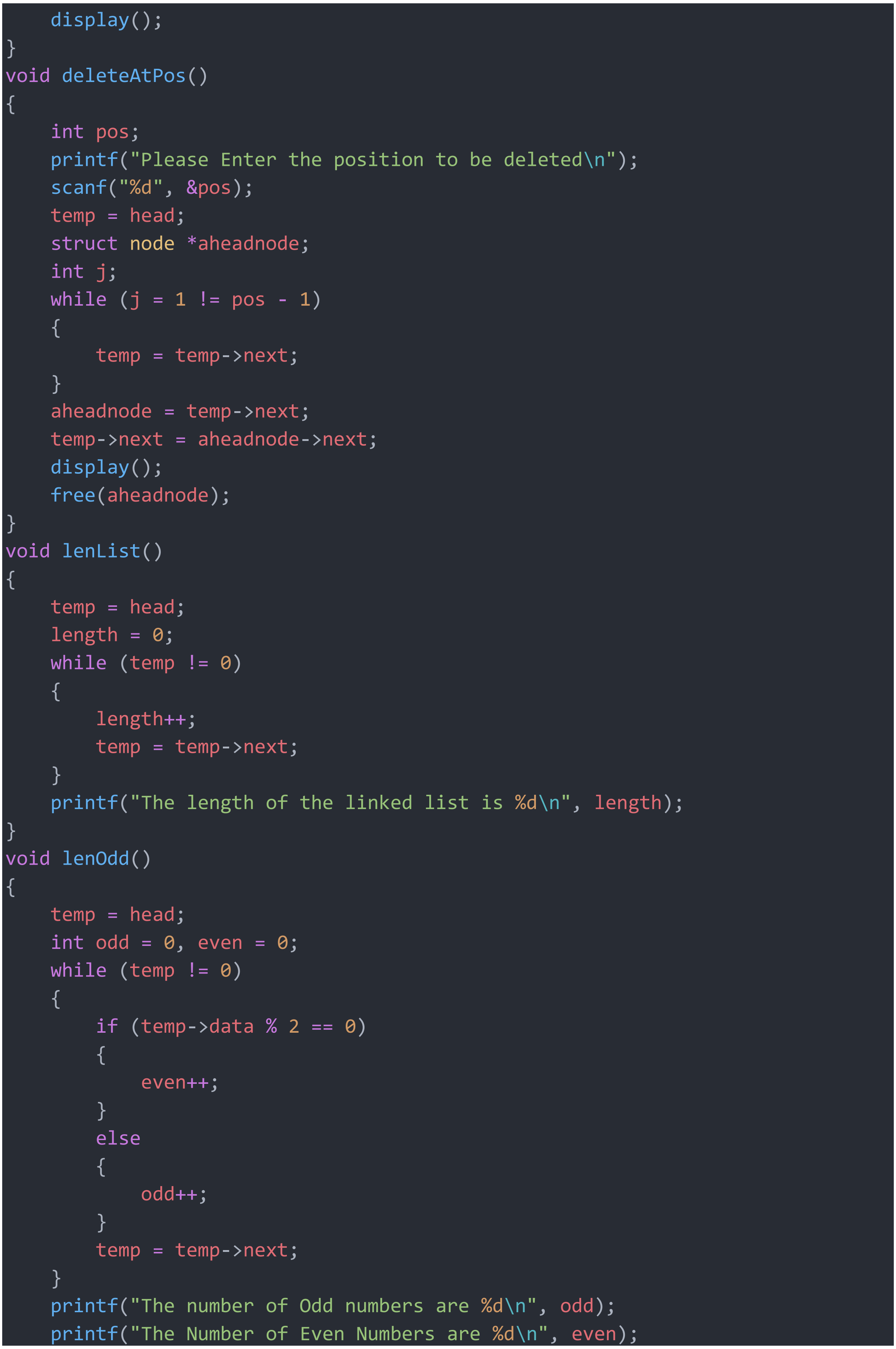
```
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
2
Enter number to be stored
15
Please Enter your choice
1.Add at Beginning
2.Add at End
3.Add at a position
4.Delete at Beginning
5.Delete at End
6.Delete at a Position
7.Length of List
8.Number of Odd and Even numbers
9.Search an Element
10.Reverse of List
11.Show
12.Exit
11
5       10      15
```

Complete Code

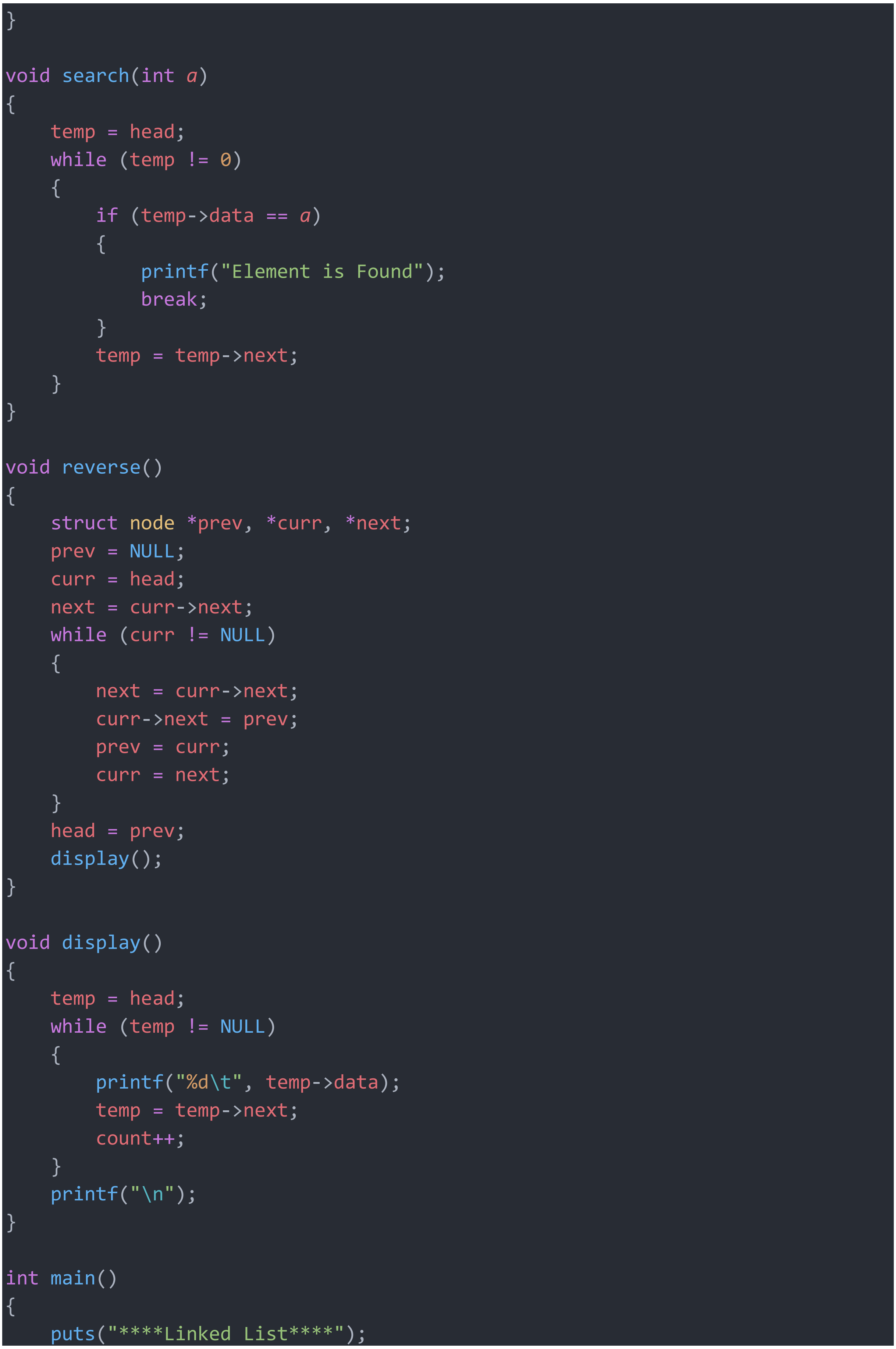
```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>

int num, choice, count, length;

void deleteAtEnd();
void deleteAtPos();
void display();

struct node
{
    int data;
    struct node *next;
} * newnode, *temp, *head;

void insertAtBeg()
{
    newnode = (struct node *)malloc(sizeof(struct node));
    puts("Enter number to be stored");
    scanf("%d", &num);
    newnode->data = num;
    newnode->next = NULL;
    if (head == NULL)
    {
        head = temp = newnode;
    }
    else
    {
        newnode->next = head;
        head = newnode;
    }
}
void insertAtEnd()
{
    newnode = (struct node *)malloc(sizeof(struct node));
    puts("Enter number to be stored");
    scanf("%d", &num);
    newnode->data = num;
    if (head == NULL)
    {
        head = temp = newnode;
        temp->next = NULL;
    }
```



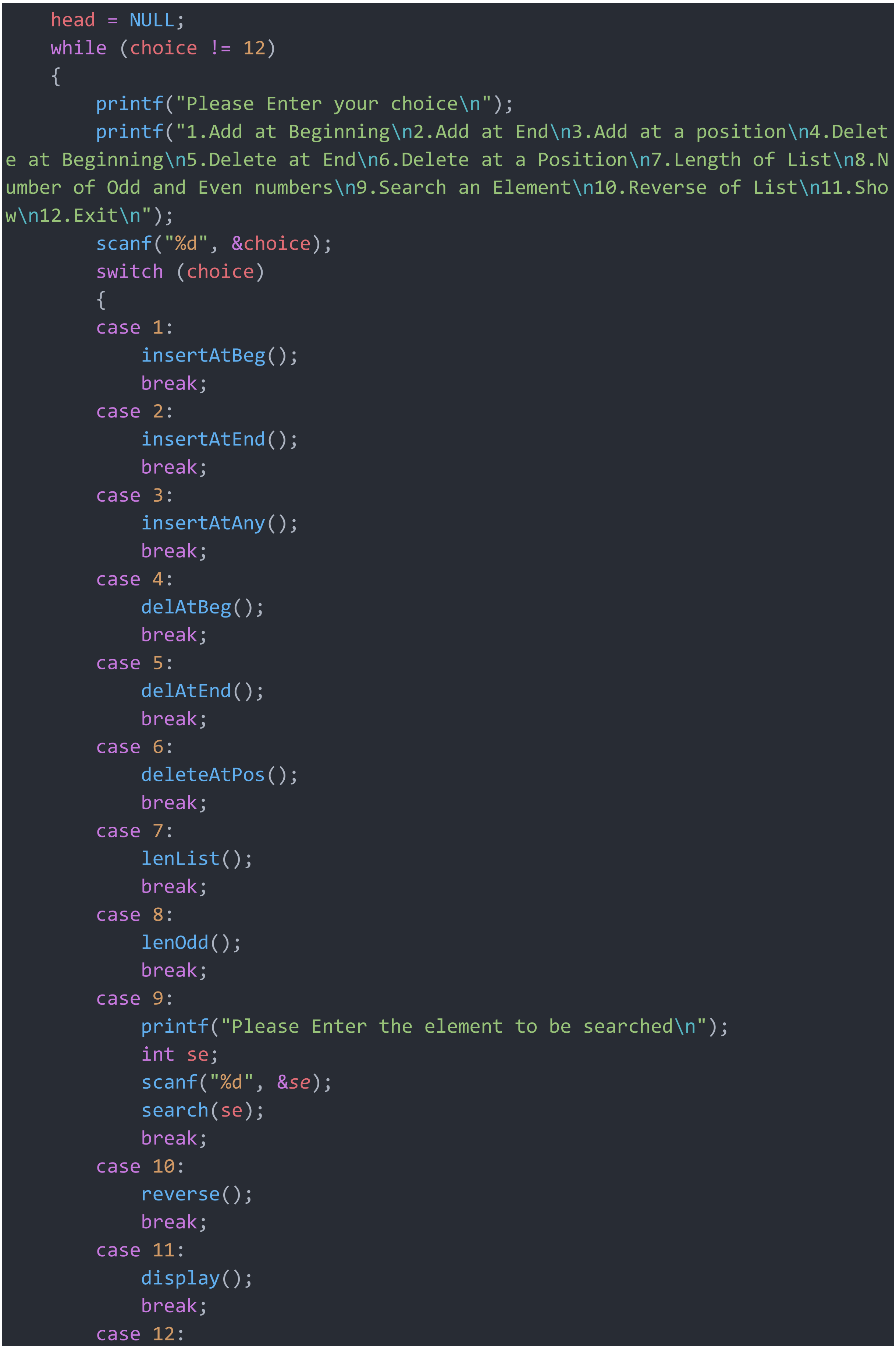
```
    else
    {
        temp = head;
        while ((temp->next) != NULL)
        {
            temp = temp->next;
        }
        newnode->next = NULL;
        temp->next = newnode;
    }
}
void insertAtAny()
{
    printf("Please Enter the position to be inserted\n");
    int pos;
    scanf("%d", &pos);
    newnode = (struct node *)malloc(sizeof(struct node));
    printf("Enter the number to be inserted\n");
    scanf("%d", &num);
    newnode->data = num;
    temp = head;
    int i = 1;

    for (i = 1; i < pos - 1 && temp != NULL; i++)
    {
        temp = temp->next;
    }
    newnode->next = temp->next;
    temp->next = newnode;
}

void delAtBeg()
{
    temp = head;
    head = temp->next;
    free(temp);
}
void delAtEnd()
{
    struct node *prevnode;
    temp = head;
    while (temp->next != NULL)
    {
        prevnode = temp;
        temp = temp->next;
    }
    prevnode->next = NULL;
    free(temp);
```



```
    display();
}
void deleteAtPos()
{
    int pos;
    printf("Please Enter the position to be deleted\n");
    scanf("%d", &pos);
    temp = head;
    struct node *aheadnode;
    int j;
    while (j = 1 != pos - 1)
    {
        temp = temp->next;
    }
    aheadnode = temp->next;
    temp->next = aheadnode->next;
    display();
    free(aheadnode);
}
void lenList()
{
    temp = head;
    length = 0;
    while (temp != 0)
    {
        length++;
        temp = temp->next;
    }
    printf("The length of the linked list is %d\n", length);
}
void lenOdd()
{
    temp = head;
    int odd = 0, even = 0;
    while (temp != 0)
    {
        if (temp->data % 2 == 0)
        {
            even++;
        }
        else
        {
            odd++;
        }
        temp = temp->next;
    }
    printf("The number of Odd numbers are %d\n", odd);
    printf("The Number of Even Numbers are %d\n", even);
```



```
}

void search(int a)
{
    temp = head;
    while (temp != 0)
    {
        if (temp->data == a)
        {
            printf("Element is Found");
            break;
        }
        temp = temp->next;
    }
}

void reverse()
{
    struct node *prev, *curr, *next;
    prev = NULL;
    curr = head;
    next = curr->next;
    while (curr != NULL)
    {
        next = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next;
    }
    head = prev;
    display();
}

void display()
{
    temp = head;
    while (temp != NULL)
    {
        printf("%d\t", temp->data);
        temp = temp->next;
        count++;
    }
    printf("\n");
}

int main()
{
    puts("****Linked List****");
```



```
    head = NULL;
    while (choice != 12)
    {
        printf("Please Enter your choice\n");
        printf("1.Add at Beginning\n2.Add at End\n3.Add at a position\n4.Delete at Beginning\n5.Delete at End\n6.Delete at a Position\n7.Length of List\n8.Number of Odd and Even numbers\n9.Search an Element\n10.Reverse of List\n11.Show\n12.Exit\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            insertAtBeg();
            break;
        case 2:
            insertAtEnd();
            break;
        case 3:
            insertAtAny();
            break;
        case 4:
            delAtBeg();
            break;
        case 5:
            delAtEnd();
            break;
        case 6:
            deleteAtPos();
            break;
        case 7:
            lenList();
            break;
        case 8:
            lenOdd();
            break;
        case 9:
            printf("Please Enter the element to be searched\n");
            int se;
            scanf("%d", &se);
            search(se);
            break;
        case 10:
            reverse();
            break;
        case 11:
            display();
            break;
        case 12:
```

```c
            printf("Exiting.....");
            break;
        default:
            printf("Invalid Input");
            break;
        }
    }
    return 1;
}
```