SISTER NIVEDITA UNIVERSITY	SNU SISTER NIVEDITA UNIVERSITY

School of Engineering

Subject Name:	DATA STRUCTURES AND ALGORITHMS
Subject Code:	
Department:	B.TECH IN COMPUTER SCIENCE AND ENGINEERING

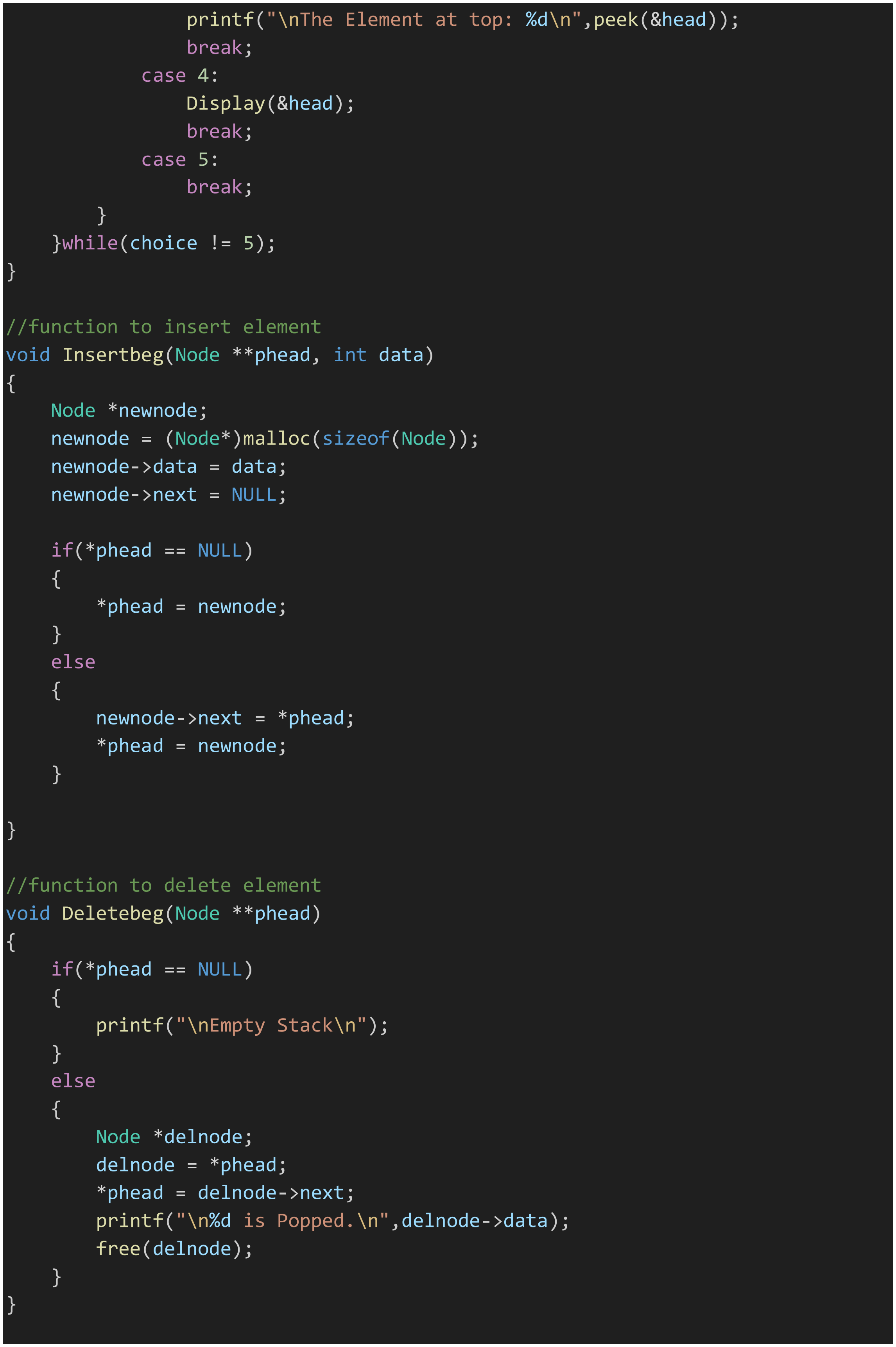
Name:	CHANDREYEE SHOME
Enrolment Number:	2212200001166
Registration Number:	220010415539
Semester:	2nd
Academic Year:	1st (2022 - 2023)

ASSIGNMENT 6

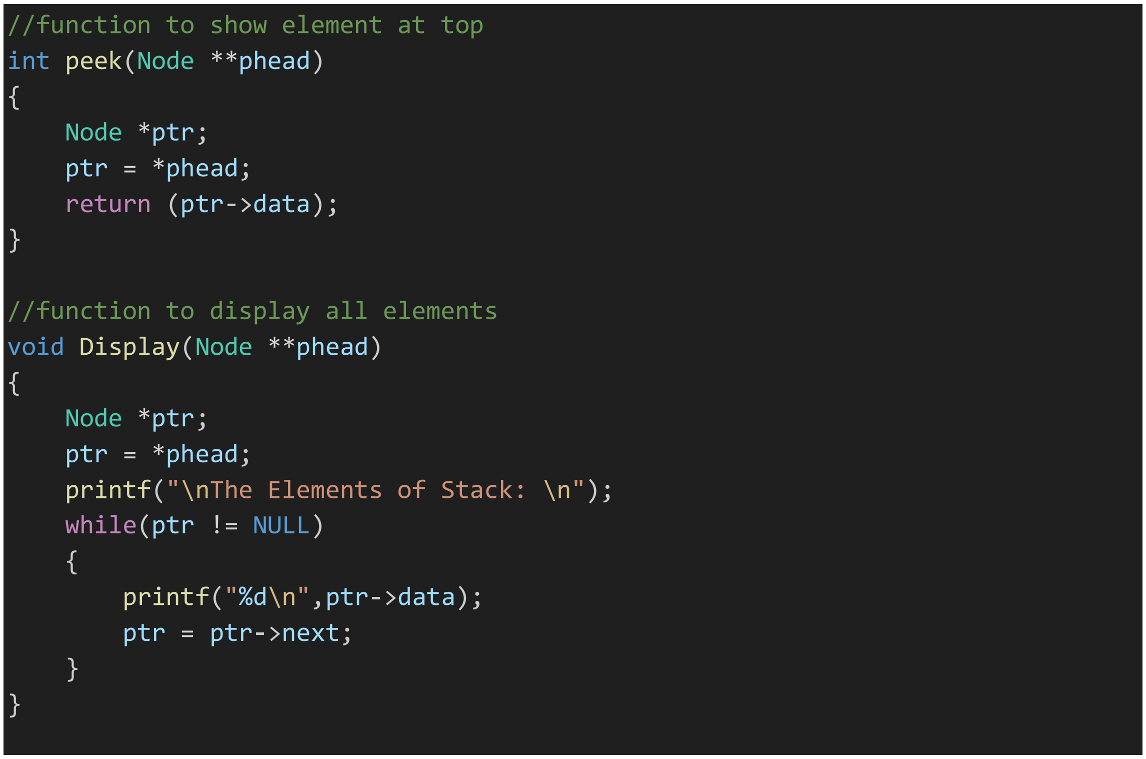
Q 1. Implement a stack using linked lists.

CODE:

```
//STACK IMPLEMENTATION USING LINKED LIST
#include<stdio.h>
#include<stdlib.h>

//user defined datatype Node
typedef struct Node
{
        int data;
    struct Node *next;
}Node;

//function declaration
void Insertbeg(Node **phead, int data);
void Deletebeg(Node **phead);

int peek(Node **phead);

void Display(Node **phead);

int main()
{
    int choice;
    system("cls");

    Node *head;
    head = NULL;
    int data;

    do
    {
        printf("\n1. PUSH\n2. POP\n3. DISPLAY TOP\n4. DISPLAY STACK\n5. STOP\n");
        printf("\nEnter your choice: ");
        scanf("%d",&choice);

        switch(choice)
        {
            case 1:
                printf("\nEnter the element:" );
                scanf("%d", &data);
                Insertbeg(&head, data);
                break;
            case 2:
                Deletebeg(&head);
                break;
            case 3:
```

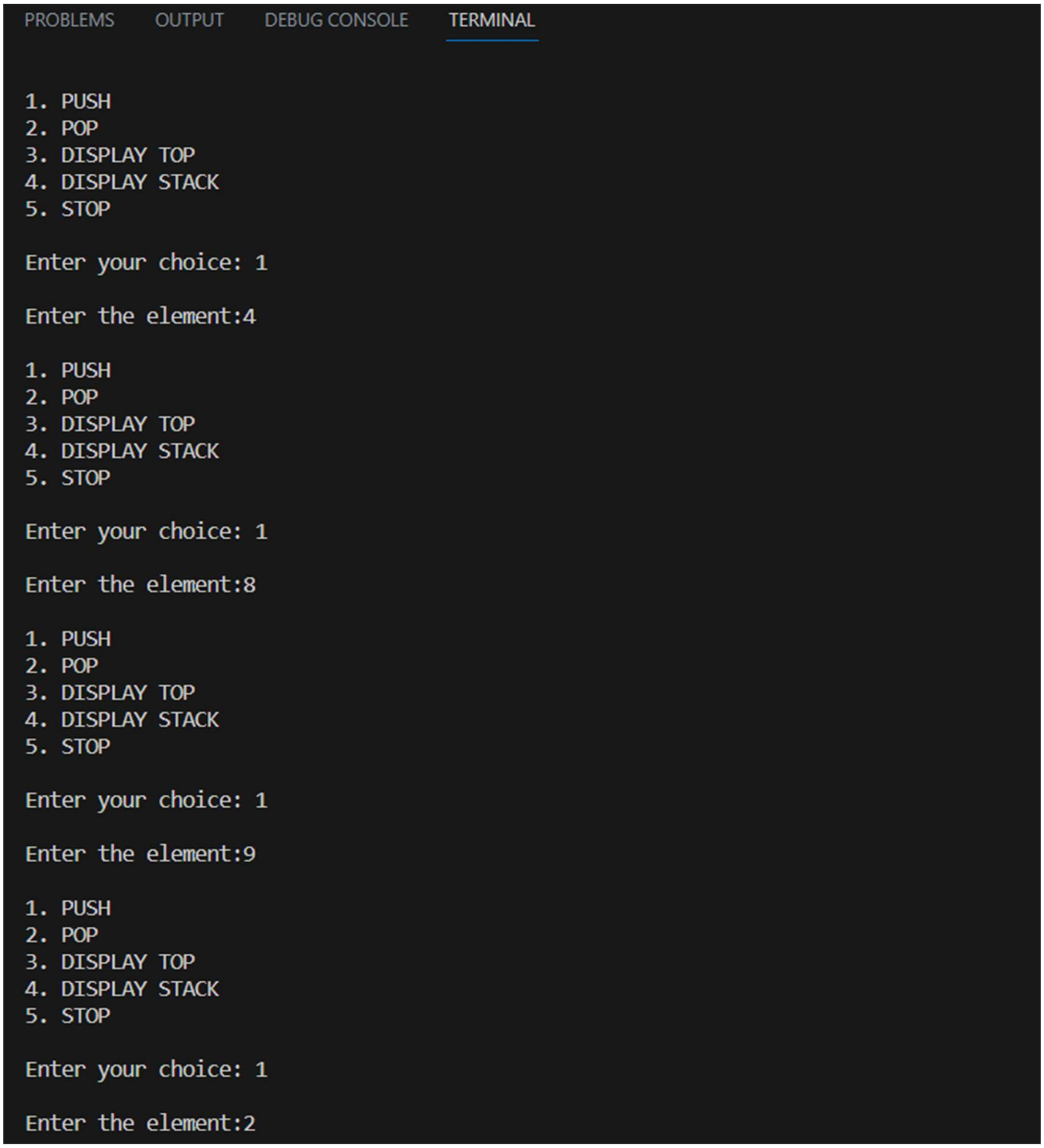


```
                printf("\nThe Element at top: %d\n",peek(&head));
                break;
            case 4:
                Display(&head);
                break;
            case 5:
                break;
        }
    }while(choice != 5);
}

//function to insert element
void Insertbeg(Node **phead, int data)
{
    Node *newnode;
    newnode = (Node*)malloc(sizeof(Node));
    newnode->data = data;
    newnode->next = NULL;

    if(*phead == NULL)
    {
        *phead = newnode;
    }
    else
    {
        newnode->next = *phead;
        *phead = newnode;
    }

}

//function to delete element
void Deletebeg(Node **phead)
{
    if(*phead == NULL)
    {
        printf("\nEmpty Stack\n");
    }
    else
    {
        Node *delnode;
        delnode = *phead;
        *phead = delnode->next;
        printf("\n%d is Popped.\n",delnode->data);
        free(delnode);
    }
}
```

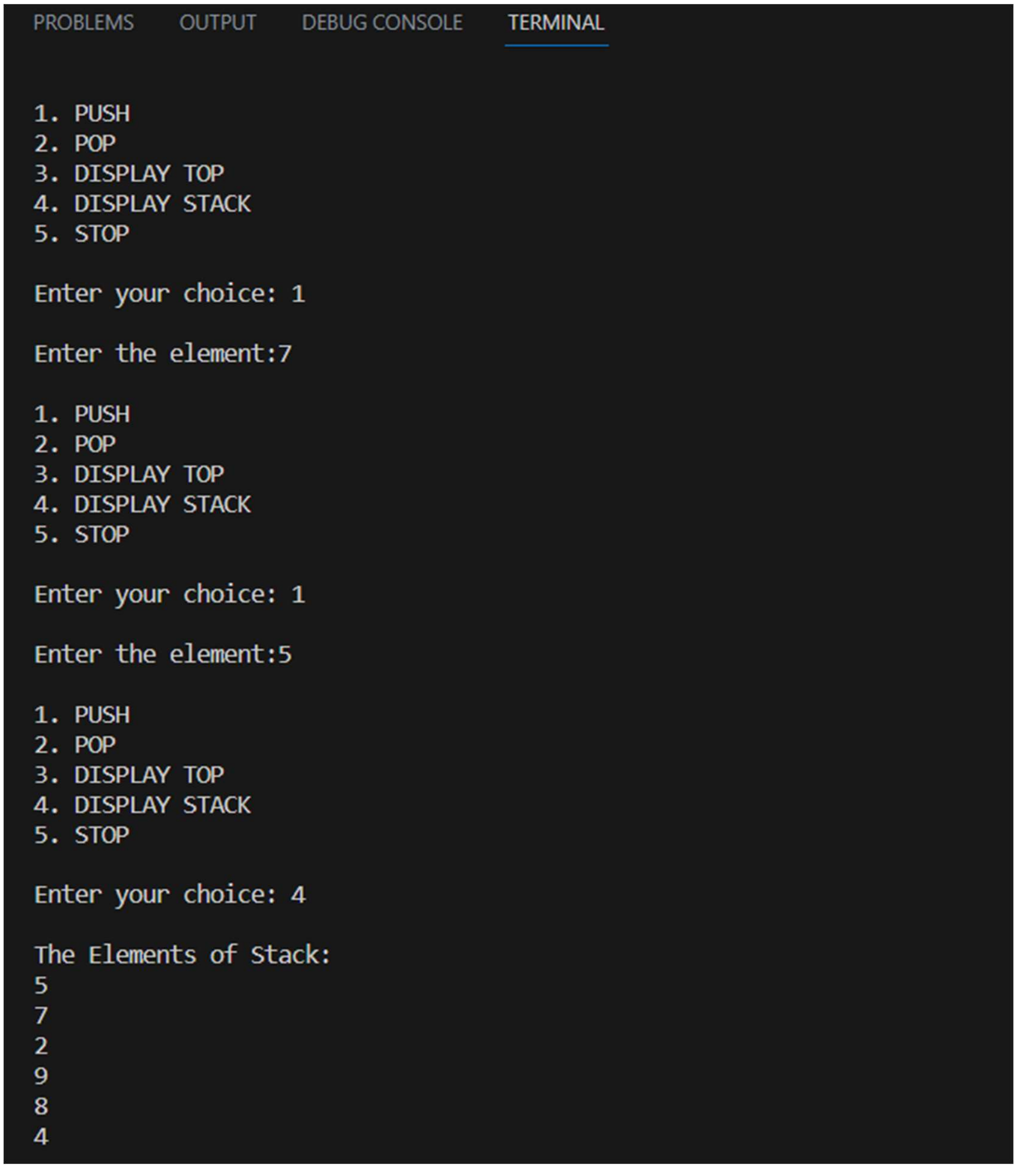


```c
//function to show element at top
int peek(Node **phead)
{
    Node *ptr;
    ptr = *phead;
    return (ptr->data);
}

//function to display all elements
void Display(Node **phead)
{
    Node *ptr;
    ptr = *phead;
    printf("\nThe Elements of Stack: \n");
    while(ptr != NULL)
    {
        printf("%d\n",ptr->data);
        ptr = ptr->next;
    }
}
```

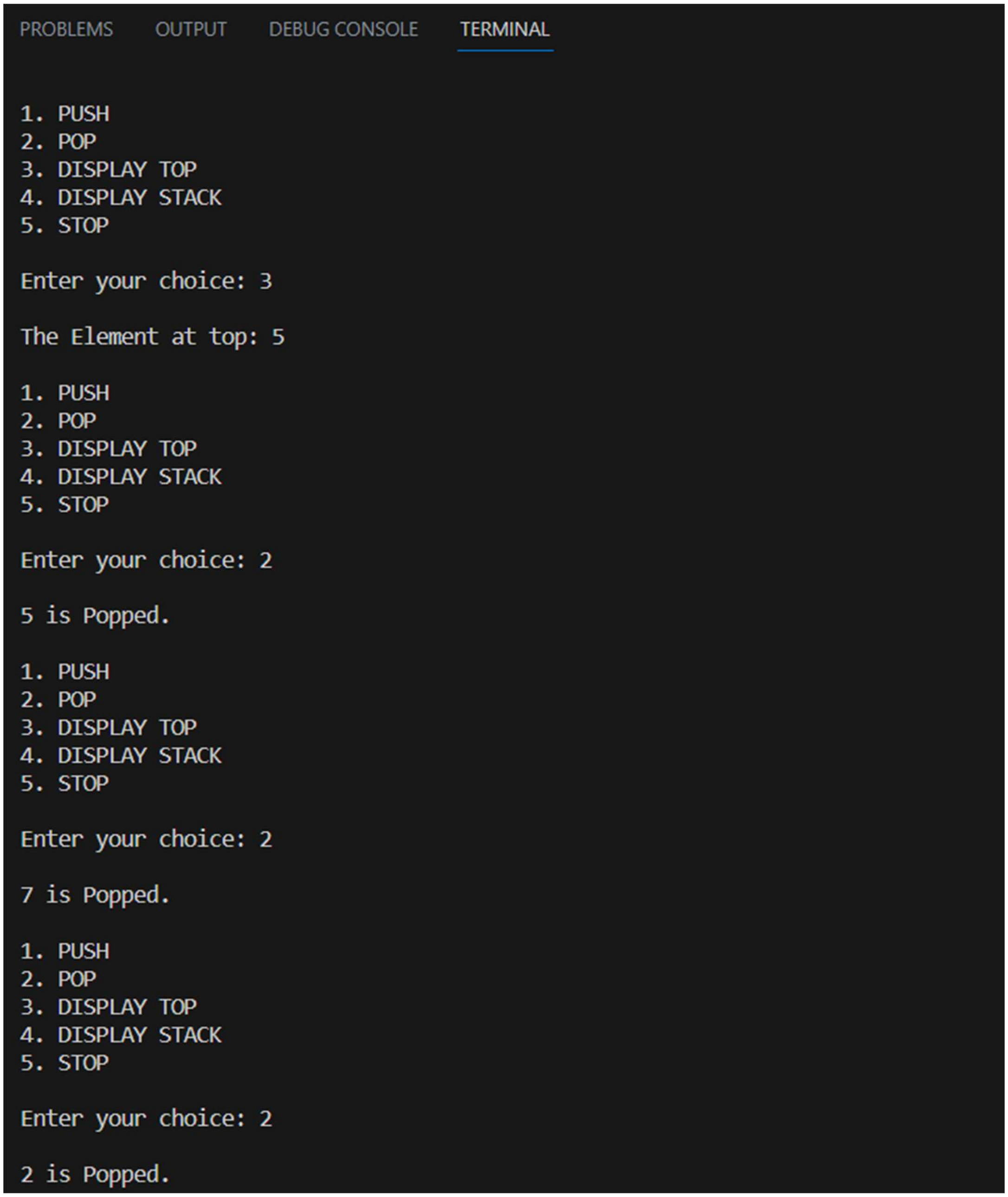
OUTPUT:



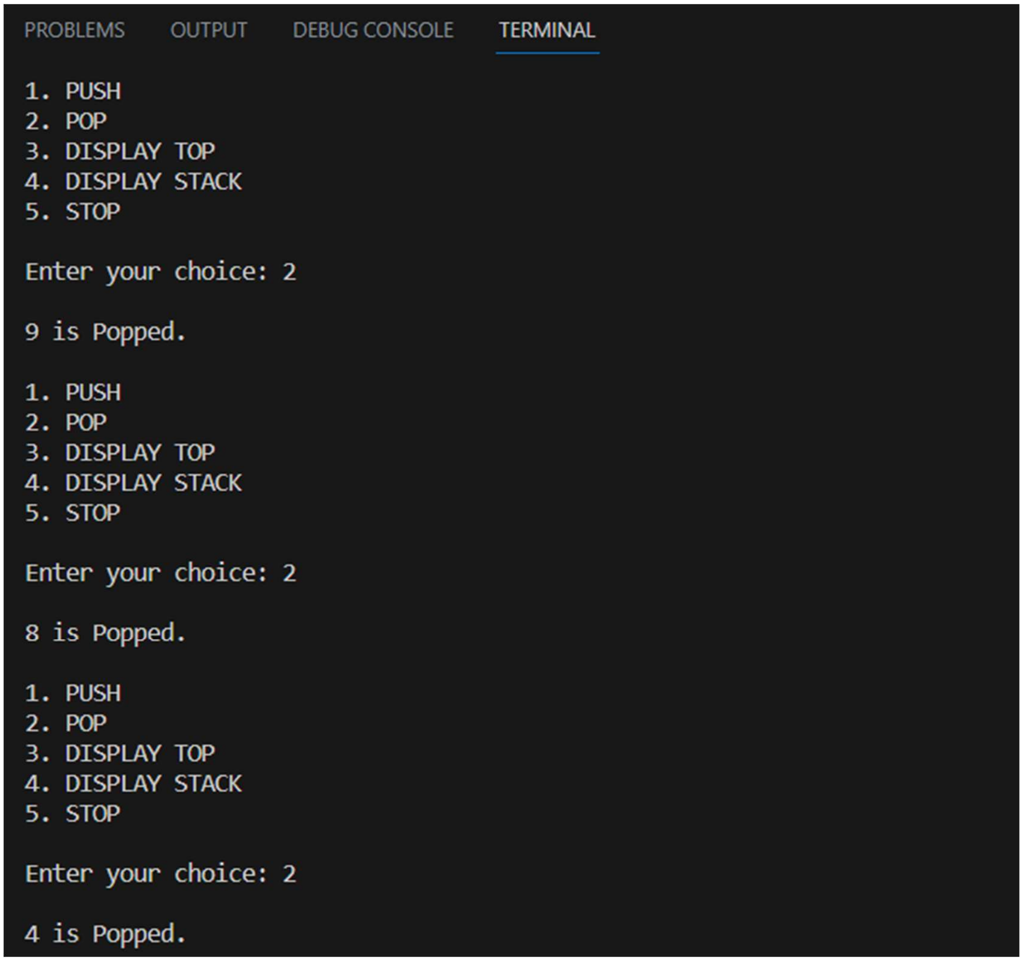
```
1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 1

Enter the element:4

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 1

Enter the element:8

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 1

Enter the element:9

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 1

Enter the element:2
```



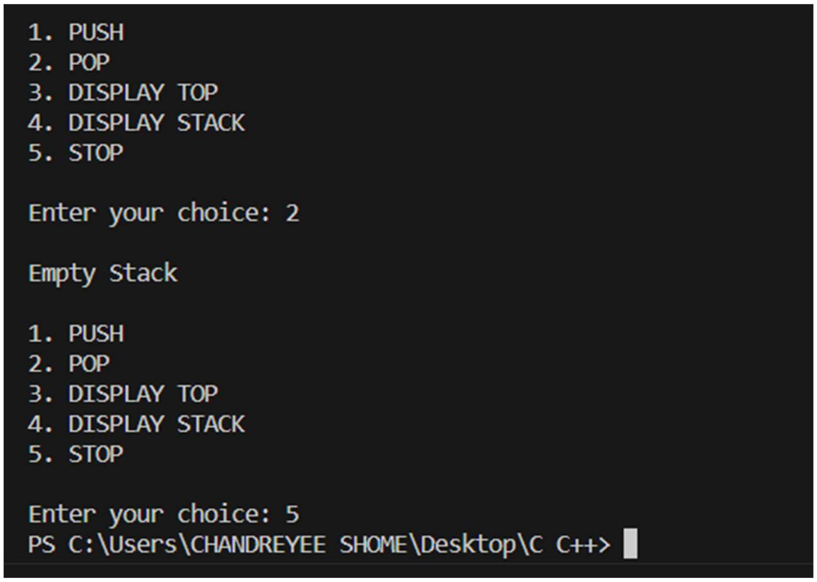
```
1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 1

Enter the element:7

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 1

Enter the element:5

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 4

The Elements of Stack:
5
7
2
9
8
4
```



```
1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 3

The Element at top: 5

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 2

5 is Popped.

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 2

7 is Popped.

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 2

2 is Popped.
```



```
1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 2

9 is Popped.

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 2

8 is Popped.

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 2

4 is Popped.
```

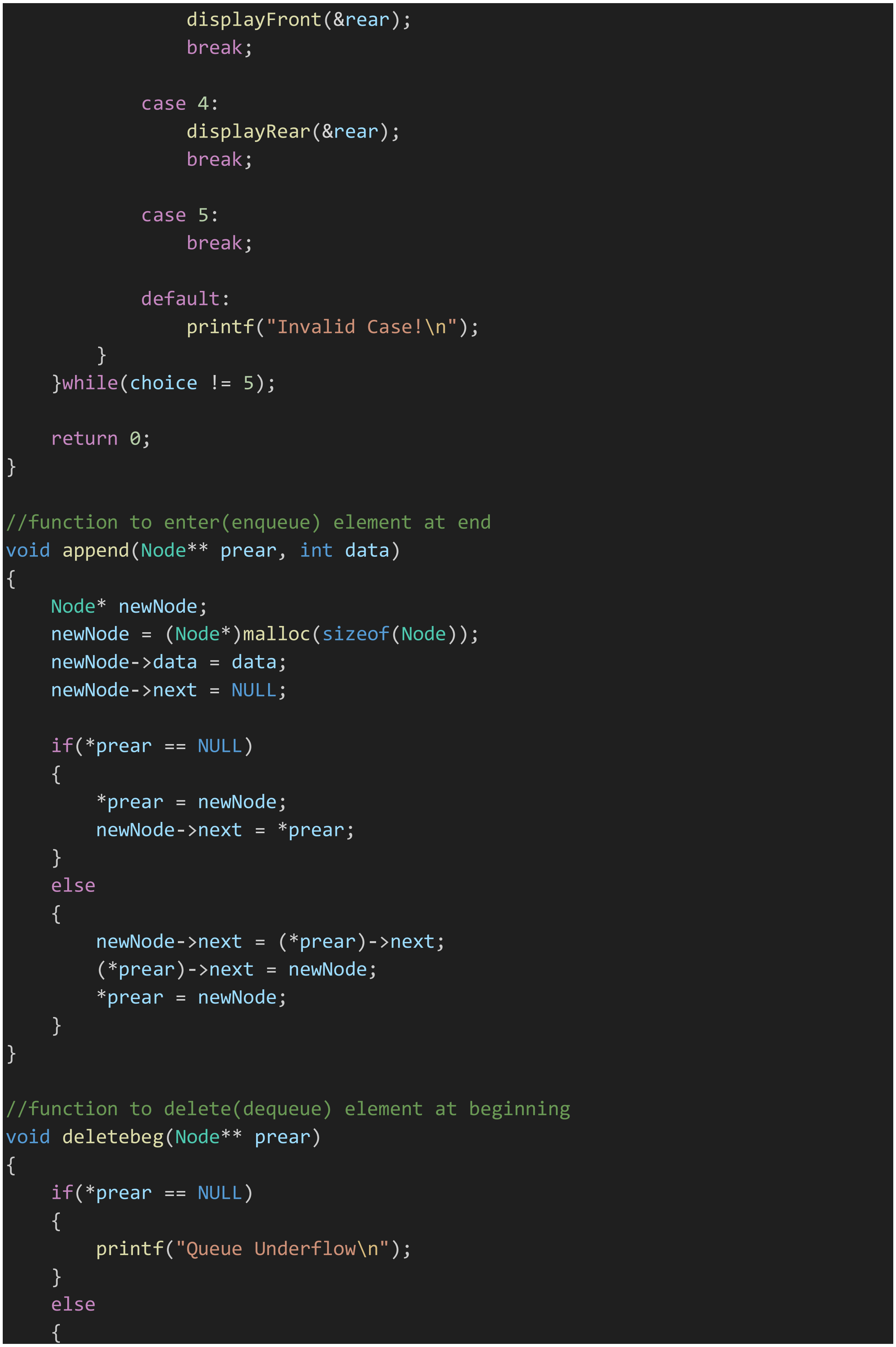


```
1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 2

Empty Stack

1. PUSH
2. POP
3. DISPLAY TOP
4. DISPLAY STACK
5. STOP

Enter your choice: 5
PS C:\Users\CHANDREYEE SHOME\Desktop\C C++>
```

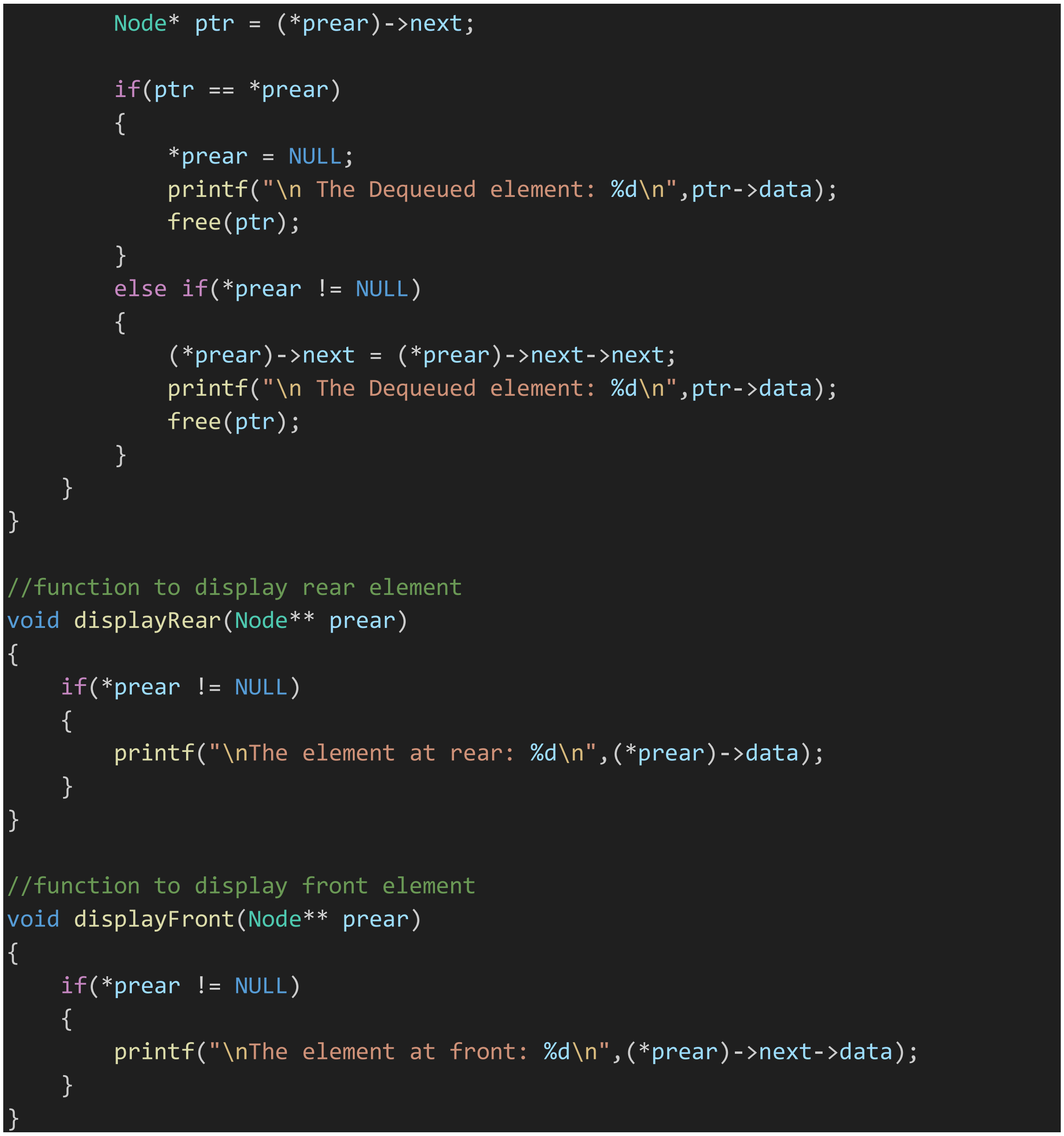
Q 2. Implement Circular Linked List, queue using linked lists.

CODE:

```
//QUEUE USING LINKED LIST

//header files
#include<stdio.h>
#include<stdlib.h>

//user defined data structure Node
typedef struct Node
{
    int data;
    struct Node *next;
}Node;

//function declaration
void append(Node** prear, int data);
void deletebeg(Node** prear);

void displayRear(Node** prear);
void displayFront(Node** prear);

int main()
{
    Node* rear;
    rear = NULL;
    int data, choice;
    system("cls");

    do
    {
        printf("\n1. ENQUEUE\n2. DEQUEUE\n3. DISPLAY FRONT\n4. DISPLAY
REAR\n5. STOP\n");
        printf("\nEnter your choice: ");
        scanf("%d",&choice);

        switch (choice){
            case 1:
                printf("Enter the element: ");
                scanf("%d",&data);
                append(&rear,data);
                break;

            case 2:
                deletebeg(&rear);
                break;

            case 3:
```

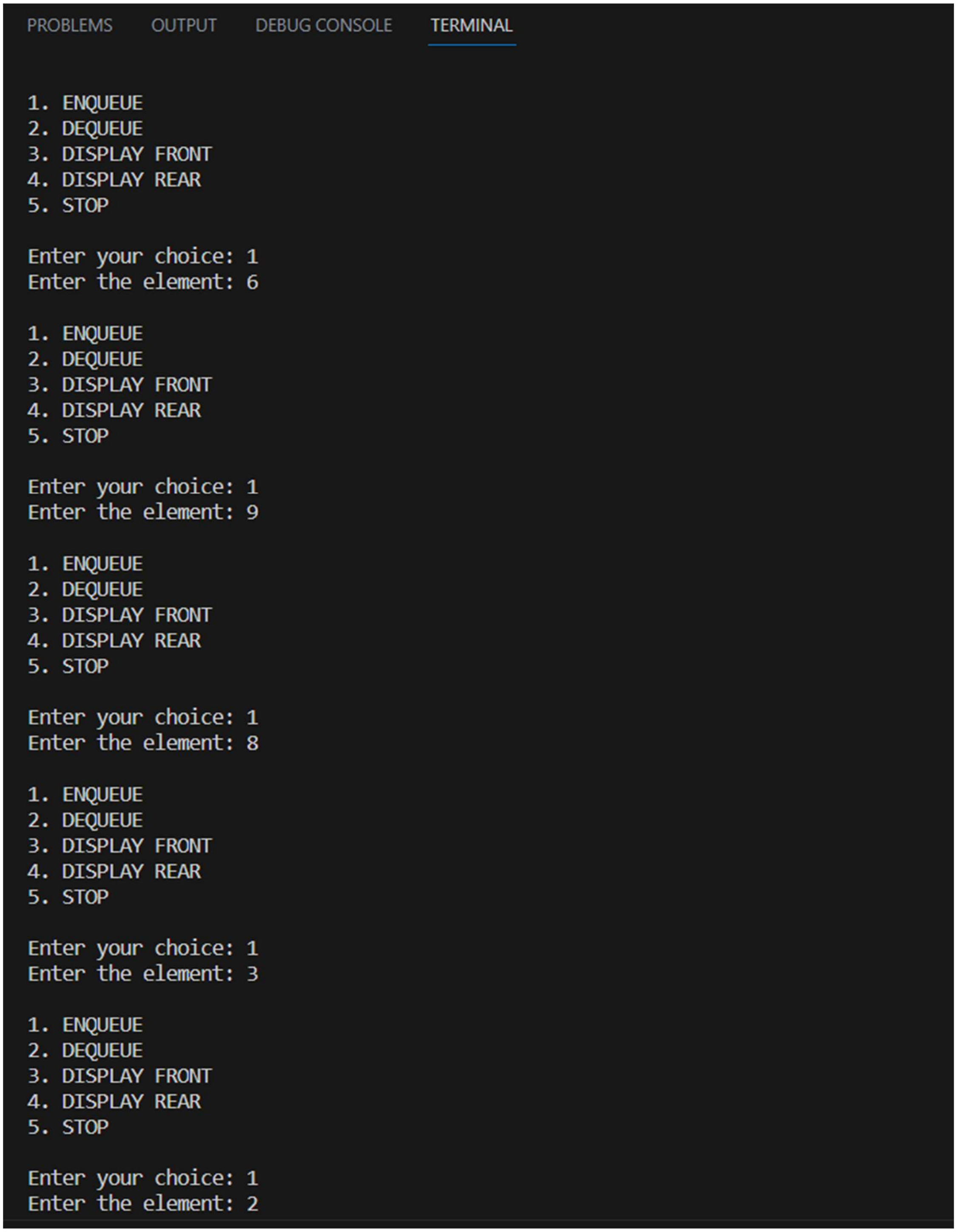


```
                displayFront(&rear);
                break;

            case 4:
                displayRear(&rear);
                break;

            case 5:
                break;

            default:
                printf("Invalid Case!\n");
        }
    }while(choice != 5);

    return 0;
}

//function to enter(enqueue) element at end
void append(Node** prear, int data)
{
    Node* newNode;
    newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->next = NULL;

    if(*prear == NULL)
    {
        *prear = newNode;
        newNode->next = *prear;
    }
    else
    {
        newNode->next = (*prear)->next;
        (*prear)->next = newNode;
        *prear = newNode;
    }
}

//function to delete(dequeue) element at beginning
void deletebeg(Node** prear)
{
    if(*prear == NULL)
    {
        printf("Queue Underflow\n");
    }
    else
    {
```

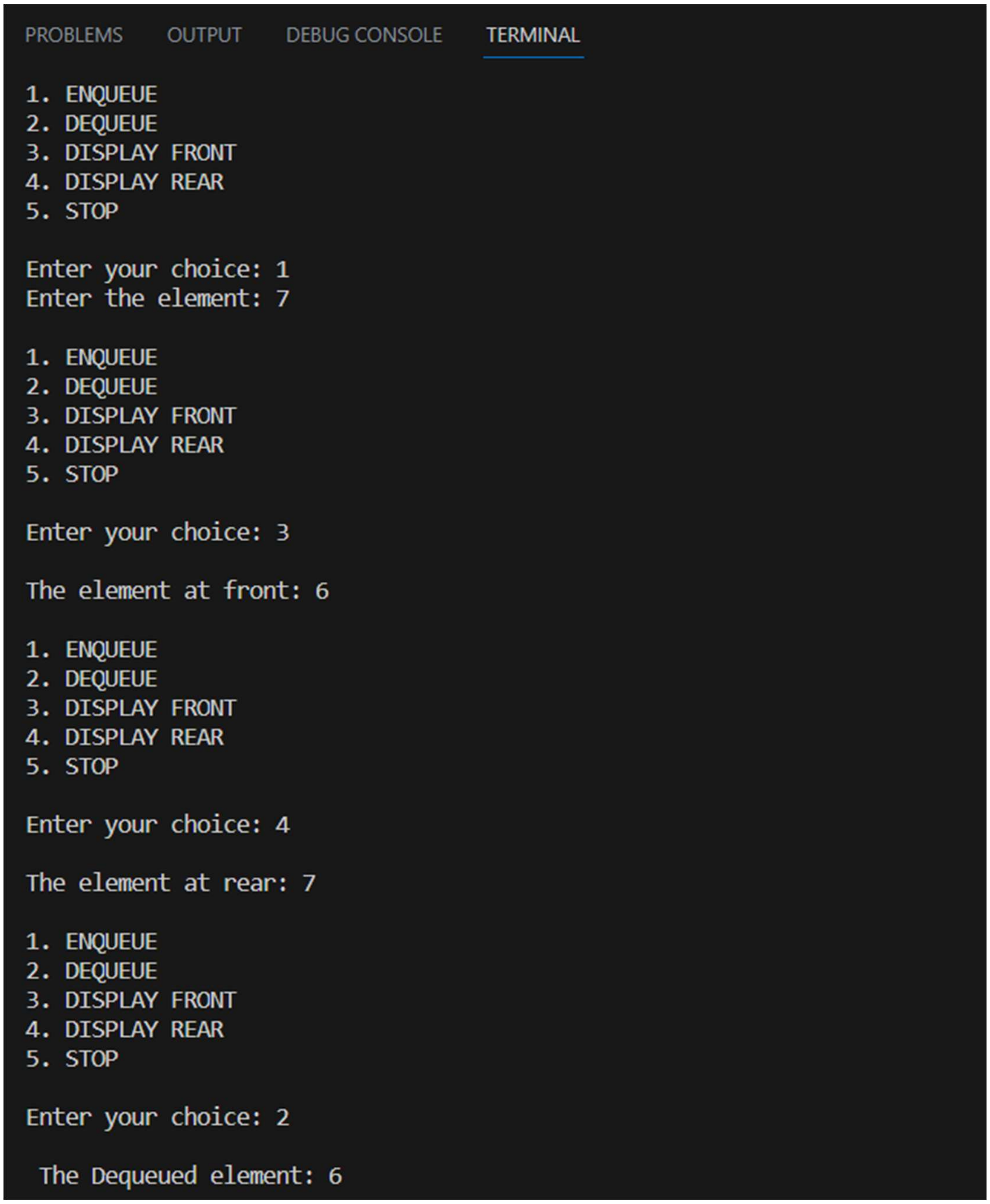


```
        Node* ptr = (*prear)->next;

        if(ptr == *prear)
        {
            *prear = NULL;
            printf("\n The Dequeued element: %d\n",ptr->data);
            free(ptr);
        }
        else if(*prear != NULL)
        {
            (*prear)->next = (*prear)->next->next;
            printf("\n The Dequeued element: %d\n",ptr->data);
            free(ptr);
        }
    }
}

//function to display rear element
void displayRear(Node** prear)
{
    if(*prear != NULL)
    {
        printf("\nThe element at rear: %d\n",(*prear)->data);
    }
}

//function to display front element
void displayFront(Node** prear)
{
    if(*prear != NULL)
    {
        printf("\nThe element at front: %d\n",(*prear)->next->data);
    }
}
```

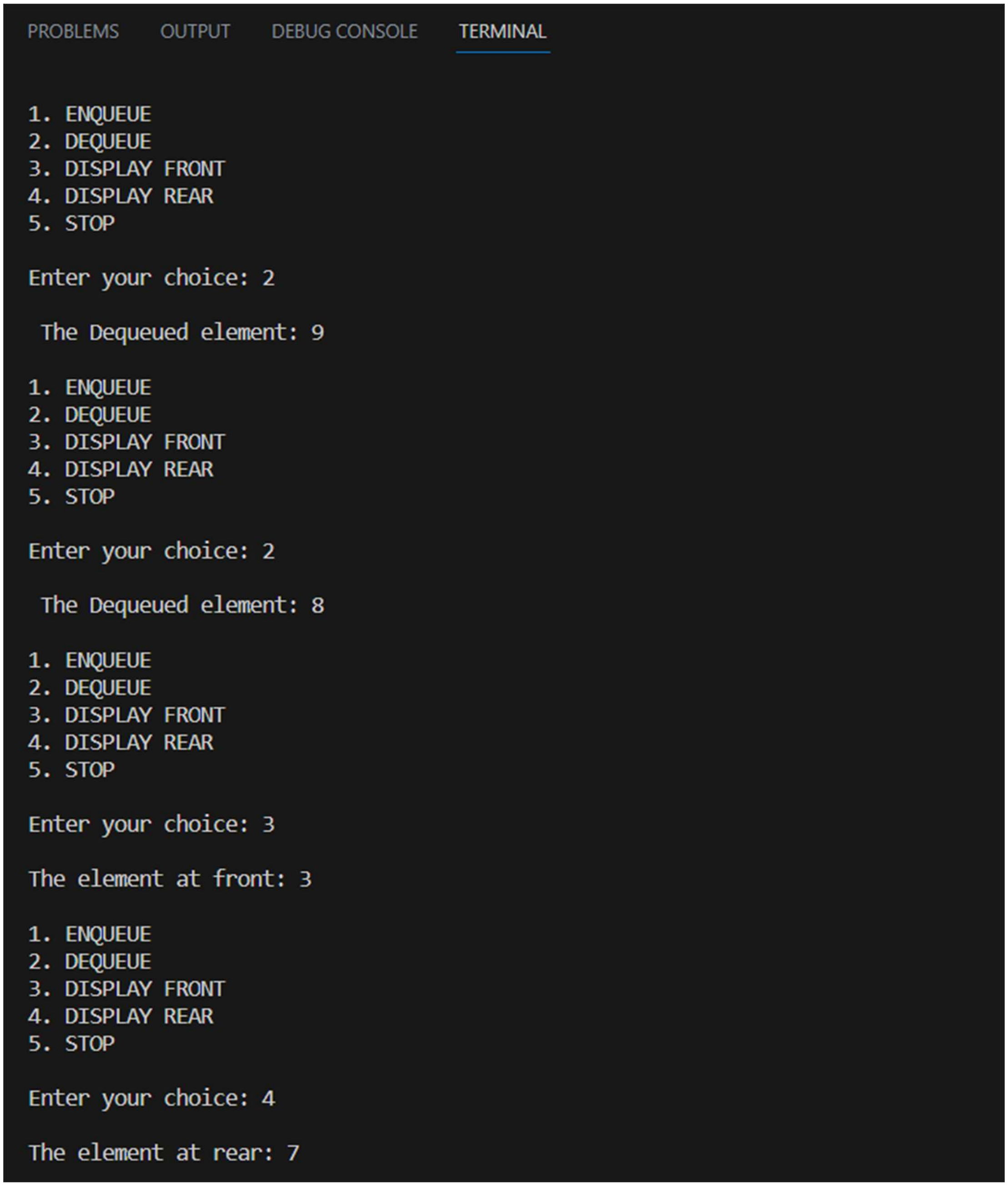
OUTPUT:



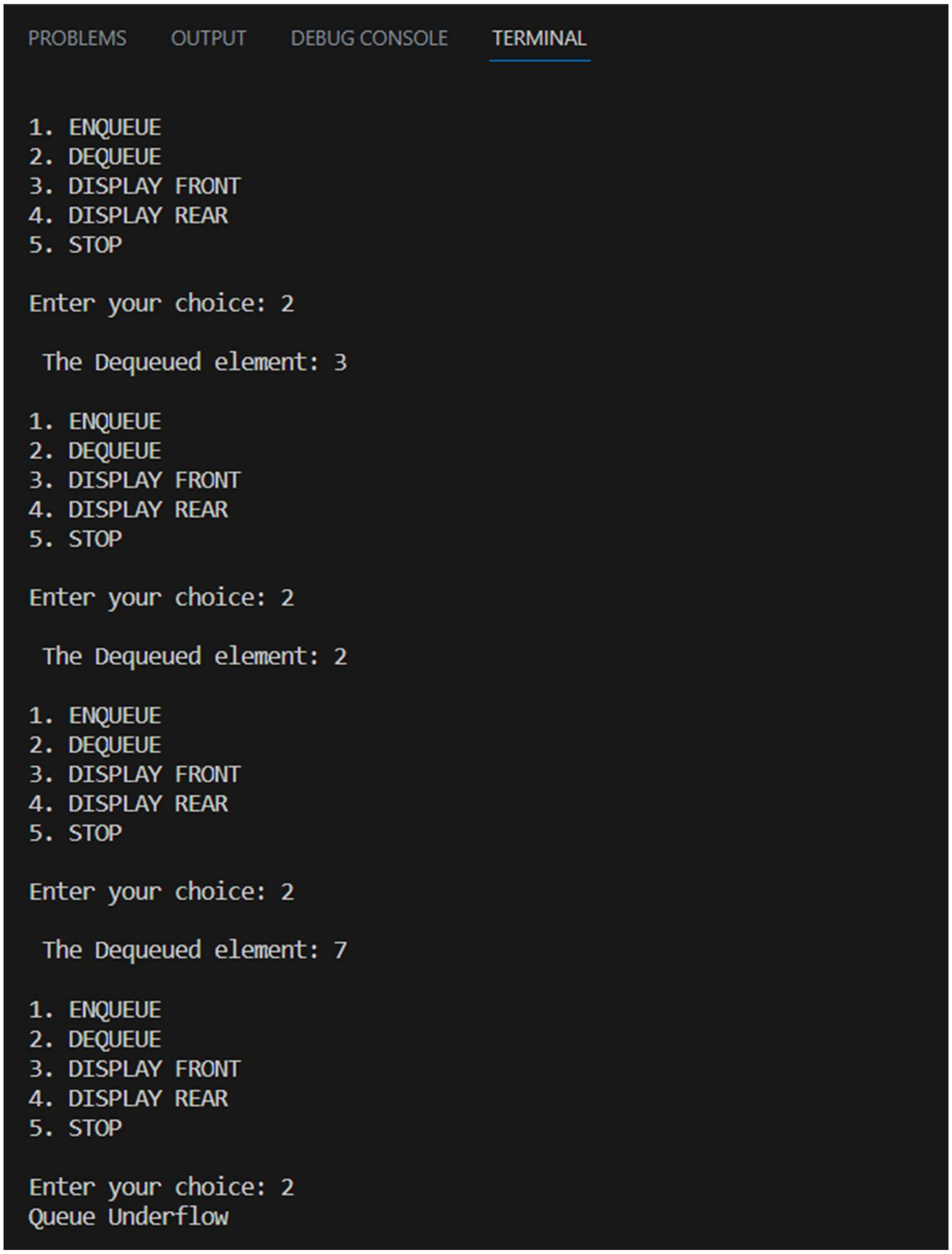
```
1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 1
Enter the element: 6

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 1
Enter the element: 9

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 1
Enter the element: 8

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 1
Enter the element: 3

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 1
Enter the element: 2
```



```
1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 1
Enter the element: 7

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 3

The element at front: 6

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 4

The element at rear: 7

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 2

 The Dequeued element: 6
```



```
1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 2

 The Dequeued element: 9

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 2

 The Dequeued element: 8

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 3

The element at front: 3

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 4

The element at rear: 7
```



```
1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 2

 The Dequeued element: 3

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 2

 The Dequeued element: 2

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 2

 The Dequeued element: 7

1. ENQUEUE
2. DEQUEUE
3. DISPLAY FRONT
4. DISPLAY REAR
5. STOP

Enter your choice: 2
Queue Underflow
```