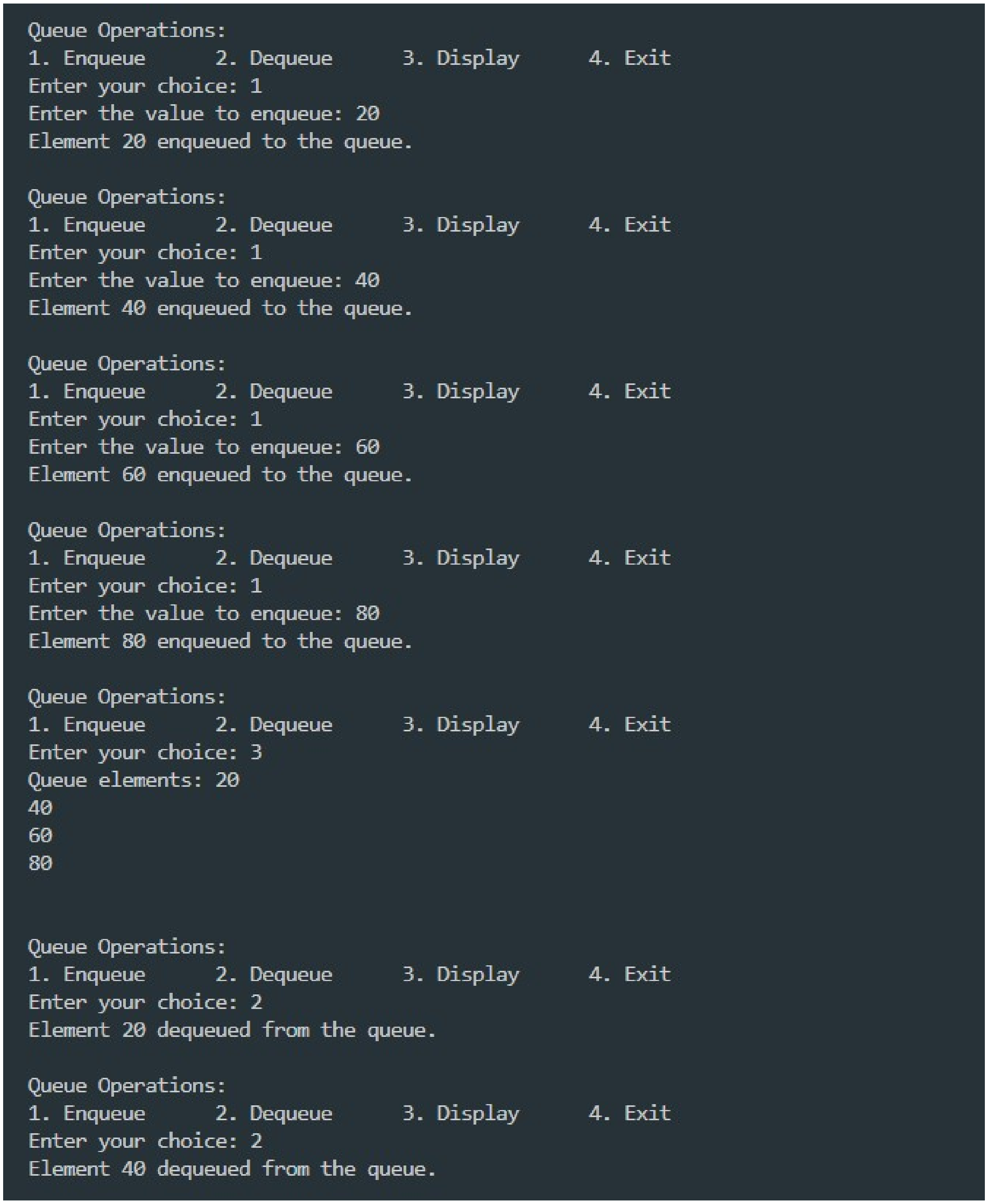
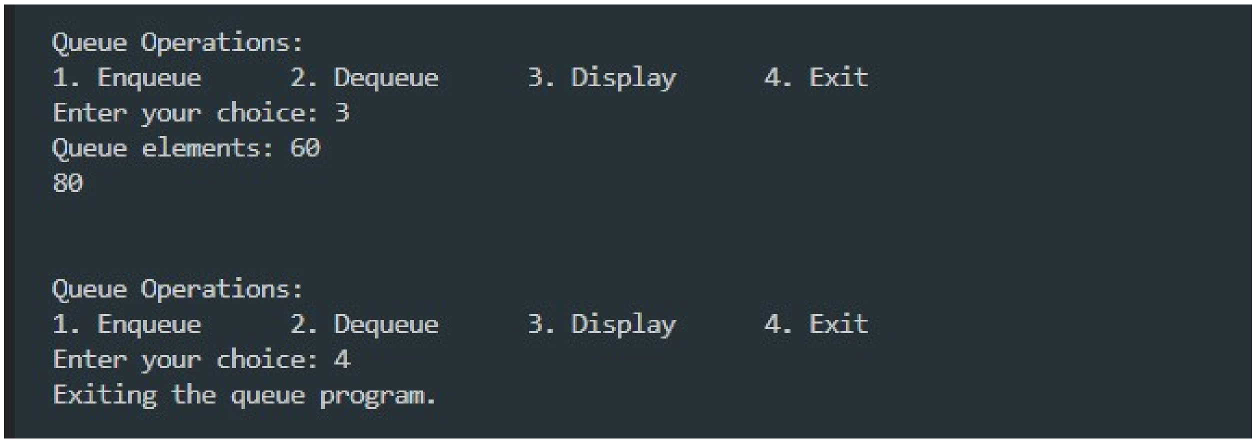
WAP to Implement Single Link List to simulate Queue

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

struct Node
{
    int data;
    struct Node *next;
};

struct Node *front = NULL;
struct Node *rear = NULL;

void enqueue(int value)
{
    struct Node *newNode = (struct Node *)malloc(sizeof(struct Node));
    if (newNode == NULL)
    {
        printf("Memory allocation failed.\n");
        return;
    }
    newNode->data = value;
    newNode->next = NULL;
    if (front == NULL)
    {
        front = rear = newNode;
    }
    else
    {
        rear->next = newNode;
        rear = newNode;
    }
    printf("Element %d enqueued to the queue.\n", value);
}

void dequeue()
{
    if (front == NULL)
    {
        printf("Queue underflow. Cannot dequeue from an empty queue.\n");
        return;
    }
    struct Node *temp = front;
    front = front->next;
    if (front == NULL)
    {
        rear = NULL;
    }
    printf("Element %d dequeued from the queue.\n", temp->data);
    free(temp);
}

void display()
{
    if (front == NULL)
    {
        printf("Queue is empty.\n");
        return;
    }
    struct Node *temp = front;
    printf("Queue elements: ");
    while (temp != NULL)
    {
        printf("%d\n", temp->data);
        temp = temp->next;
    }
    printf("\n");
}

int main()
{
    int choice, value;

    while (1)
    {
        printf("\nQueue Operations:\n");
        printf("1. Enqueue\t");
        printf("2. Dequeue\t");
        printf("3. Display\t");
        printf("4. Exit\n");

        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice)
        {
        case 1:
            printf("Enter the value to enqueue: ");
            scanf("%d", &value);
            enqueue(value);
            break;
        case 2:
            dequeue();
```

```
                break;
            case 3:
                display();
                break;
            case 4:
                printf("Exiting the queue program.\n");
                exit(0);
            default:
                printf("Invalid choice. Please enter a valid option.\n");
            }
        }

        return 0;
    }
```



```
Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 1
Enter the value to enqueue: 20
Element 20 enqueued to the queue.

Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 1
Enter the value to enqueue: 40
Element 40 enqueued to the queue.

Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 1
Enter the value to enqueue: 60
Element 60 enqueued to the queue.

Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 1
Enter the value to enqueue: 80
Element 80 enqueued to the queue.

Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 3
Queue elements: 20
40
60
80


Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 2
Element 20 dequeued from the queue.

Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 2
Element 40 dequeued from the queue.
```



```
Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 3
Queue elements: 60
80


Queue Operations:
1. Enqueue      2. Dequeue      3. Display      4. Exit
Enter your choice: 4
Exiting the queue program.
```