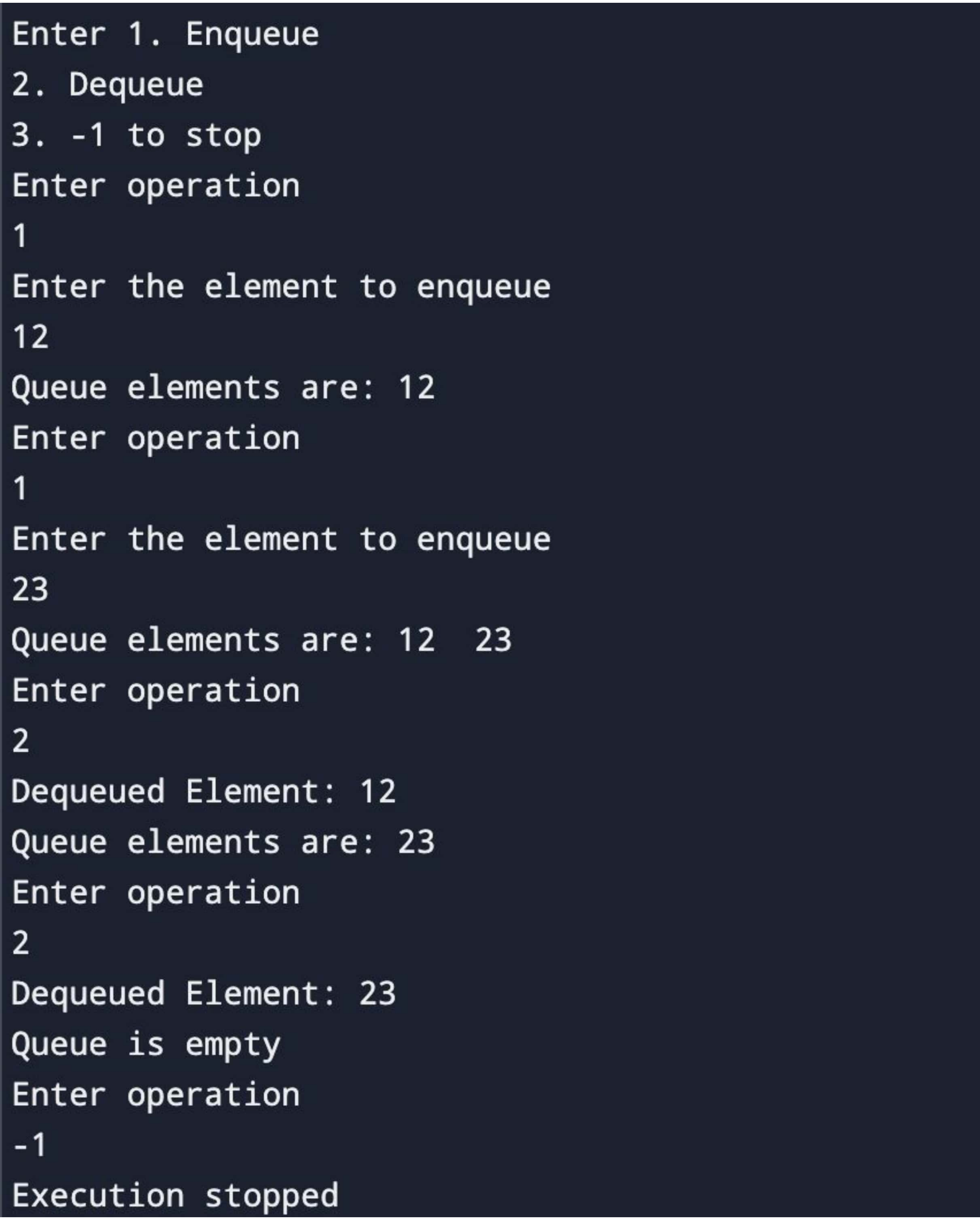
* WAP to implement queue using linked list:

⇒

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

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

void display (struct Node * front){
        if (front == NULL) {
                printf ("Queue is empty \n
                return;
        }

        struct Node * temp = front;
        printf ("Queue Elements are:");
        while (temp != NULL) {
                printf ("%d\t", temp->data
                temp = temp->next;
        }
        printf("\n")
}
```

```
void enqueue (struct Node ** front,
 struct Node ** rear, int data) {
        struct Node * newNode =
        (struct Node *) malloc (sizeof(struct
                                      Node));

        if (newNode == NULL) {
             printf ("Queue Overflow\n");
             return;
        }

          newNode -> data = data;
          newNode -> next = NULL;

          if (*rear == NULL) {
             *front = *rear = newNode;
             return;
          }

          (*rear) -> next = newNode;
          *rear = newNode;
}

int dequeue (struct Node ** front,
 struct Node ** rear) {
        if (*front == NULL) {
           printf("Queue Underflow");
           return -1;
        }
        struct Node *temp = *front;
        int dequeuedData = temp -> data;

        *front = (*front) -> next;

        if (*front == NULL){
              *rear = NULL
        }
        free (temp)
        return dequeuedData;
}

int main(){
        int op, n, dequeuedElement;
        struct Node *front = NULL;
        struct Node *rear = NULL;
        printf ("Enter 1. Enqueue\n 2.
         Dequeue\n 3. -1 to stop\n");
        while(1) {
              printf("Enter operation);
              scanf ("%d", &op)

              if (op == -1){
                    printf ("Execution stopped!);
                    break;
              }
        switch (op) {
            case 1:
                printf ("Enter the element to
                         enqueue \n");
                scanf ("%d", &n);
                enqueue (&front, &rear, n);
                break.

            case 2:
                dequeedElement = dequeue (&front,
                      &rear);
                if (dequeuedElement != -1)
                    printf ("Dequeued Element
                      : %d \n", dequeuedElement);
                }
                break;
        }
        display (front);
    }
    return 0;

}
```



```
Enter 1. Enqueue
2. Dequeue
3. -1 to stop
Enter operation
1
Enter the element to enqueue
12
Queue elements are: 12
Enter operation
1
Enter the element to enqueue
23
Queue elements are: 12  23
Enter operation
2
Dequeued Element: 12
Queue elements are: 23
Enter operation
2
Dequeued Element: 23
Queue is empty
Enter operation
-1
Execution stopped
```