6b) WAP to Implement Single Link List to simulate Stack & Queue Operations.

```c
#include <stdio.h>
#include <stdlib.h>
// Node structure
struct Node {
    int data;
    struct Node* next;
};
//  STACK USING LINKED LIST
struct Node* top = NULL;
// Push operation
void push(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = value;
    newNode->next = top;
    top = newNode;
    printf("Pushed %d into Stack\n", value);
}
// Pop operation
void pop() {
    if (top == NULL) {
        printf("Stack is Empty\n");
        return;
    }
    struct Node* temp = top;
    printf("Popped %d from Stack\n", temp->data);
    top = temp->next;
    free(temp);
}
```

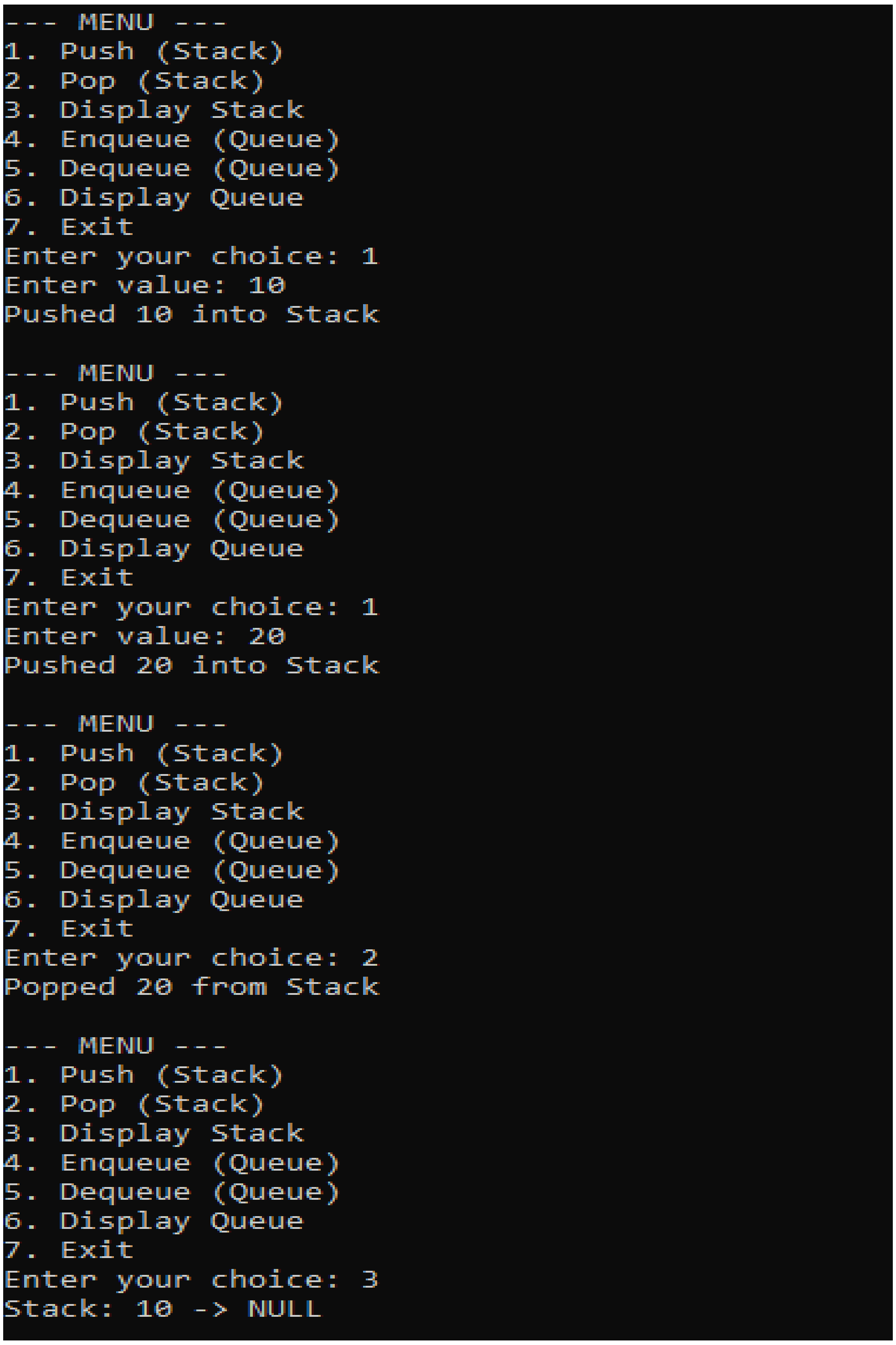
```c
// Display stack
void displayStack() {
    struct Node* temp = top;
    if (temp == NULL) {
        printf("Stack is Empty\n");
        return;
    }
    printf("Stack: ");
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}
//  QUEUE USING LINKED LIST struct Node* front = NULL;
struct Node* rear = NULL;
// Enqueue operation
void enqueue(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = value;
    newNode->next = NULL;
    if (rear == NULL) {
        front = rear = newNode;
    } else {
        rear->next = newNode;
        rear = newNode;
    }
    printf("Enqueued %d into Queue\n", value);
}
```

```c
// Dequeue operation
void dequeue() {
    if (front == NULL) {
        printf("Queue is Empty\n");
        return;
    }
    struct Node* temp = front;
    printf("Dequeued %d from Queue\n", temp->data);
    front = front->next;
    if (front == NULL)
        rear = NULL;
    free(temp);
}
// Display queue
void displayQueue() {
    struct Node* temp = front;
    if (temp == NULL) {
        printf("Queue is Empty\n");
        return;
    }
    printf("Queue: ");
    while (temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}
int main() {
    int choice, value;
```

```
while (1) {
    printf("\n--- MENU ---\n");
    printf("1. Push (Stack)\n");
    printf("2. Pop (Stack)\n");
    printf("3. Display Stack\n");
    printf("4. Enqueue (Queue)\n");
    printf("5. Dequeue (Queue)\n");
    printf("6. Display Queue\n");
    printf("7. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            printf("Enter value: ");
            scanf("%d", &value);
            push(value);
            break;
        case 2:
            pop();
            break;
        case 3:
            displayStack();
            break;
        case 4:
            printf("Enter value: ");
            scanf("%d", &value);
            enqueue(value);
```

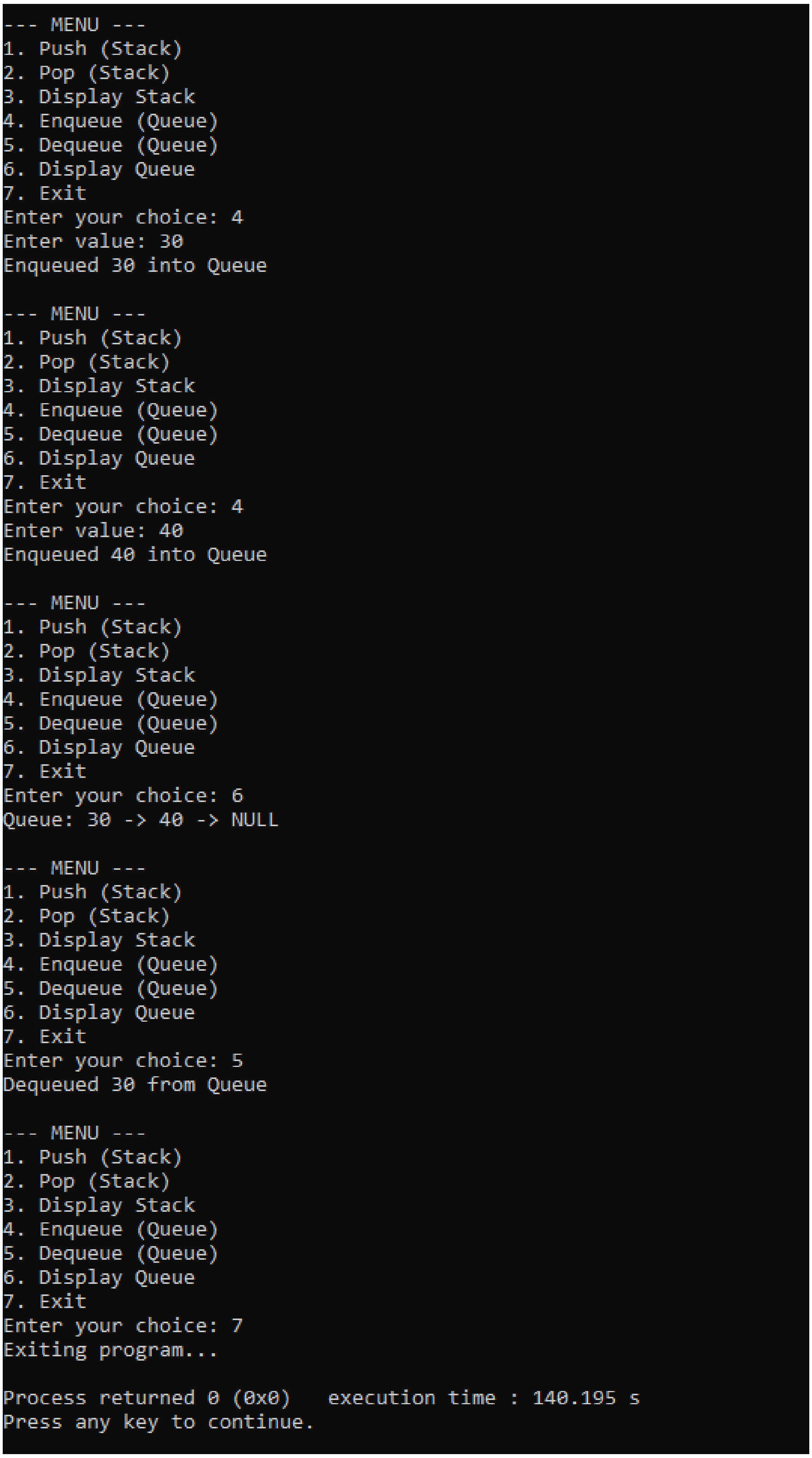
```
             break;
            case 5:
                dequeue();
                break;
            case 6:
                displayQueue();
                break;
            case 7:
                printf("Exiting program...\n");
                exit(0);
            default:
                printf("Invalid choice!\n");
        }
    }
    return 0;
}
```


```
--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 1
Enter value: 10
Pushed 10 into Stack

--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 1
Enter value: 20
Pushed 20 into Stack

--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 2
Popped 20 from Stack

--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 3
Stack: 10 -> NULL
```



```
--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 4
Enter value: 30
Enqueued 30 into Queue

--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 4
Enter value: 40
Enqueued 40 into Queue

--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 6
Queue: 30 -> 40 -> NULL

--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 5
Dequeued 30 from Queue

--- MENU ---
1. Push (Stack)
2. Pop (Stack)
3. Display Stack
4. Enqueue (Queue)
5. Dequeue (Queue)
6. Display Queue
7. Exit
Enter your choice: 7
Exiting program...

Process returned 0 (0x0)   execution time : 140.195 s
Press any key to continue.
```