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

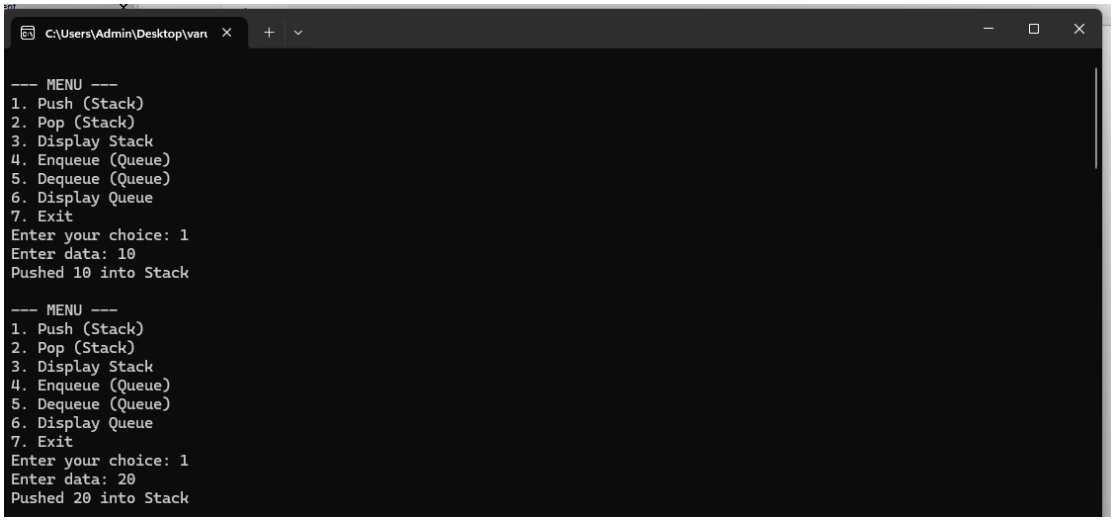
```
#include <stdio.h>
#include <stdlib.h>

/* Node structure */
struct node {
   int data;
   struct node *next;
};

/* Global pointers */
struct node *top = NULL;       // Stack
struct node *front = NULL;     // Queue
struct node *rear = NULL;

/* Create new node */
struct node* createNode(int data) {
   struct node *newNode = (struct node*)malloc(sizeof(struct node));
   newNode->data = data;
   newNode->next = NULL;
   return newNode;
}

/* -------- STACK OPERATIONS -------- */

/* Push */
void push(int data) {
   struct node *newNode = createNode(data);
   newNode->next = top;
   top = newNode;
   printf("Pushed %d into Stack\n", data);
}

/* Pop */
void pop() {
   if (top == NULL) {
      printf("Stack Underflow\n");
      return;
   }
   struct node *temp = top;
   printf("Popped element: %d\n", temp->data);
   top = top->next;
   free(temp);
}

/* Display Stack */
```

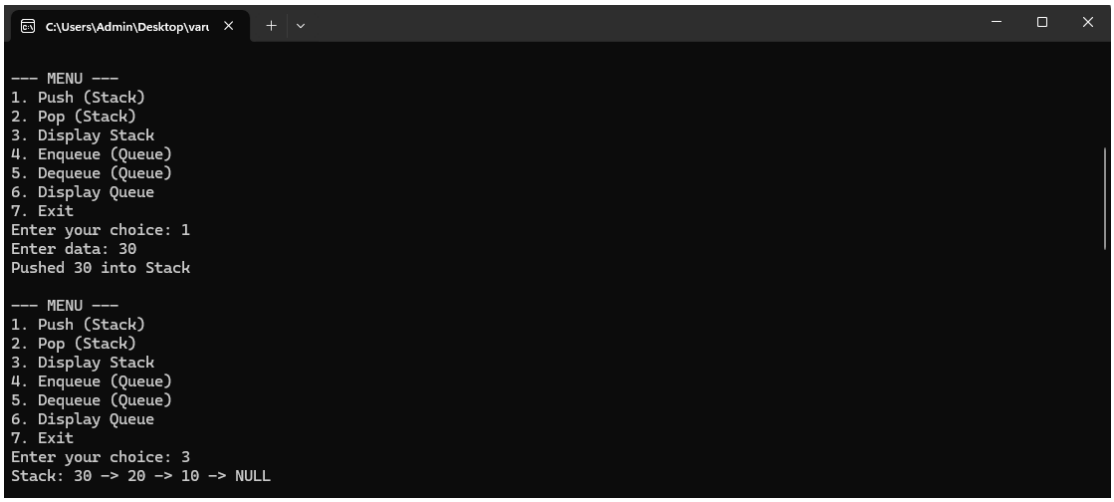
```
void displayStack() {
   struct node *temp = top;
   if (temp == NULL) {
      printf("Stack is empty\n");
      return;
   }
   printf("Stack: ");
   while (temp != NULL) {
      printf("%d -> ", temp->data);
      temp = temp->next;
   }
   printf("NULL\n");
}

/* -------- QUEUE OPERATIONS -------- */

/* Enqueue */
void enqueue(int data) {
   struct node *newNode = createNode(data);
   if (rear == NULL) {
      front = rear = newNode;
   } else {
      rear->next = newNode;
      rear = newNode;
   }
   printf("Enqueued %d into Queue\n", data);
}

/* Dequeue */
void dequeue() {
   if (front == NULL) {
      printf("Queue Underflow\n");
      return;
   }
   struct node *temp = front;
   printf("Dequeued element: %d\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");
}

/* -------- MAIN FUNCTION -------- */

int main() {
    int choice, data;

    do {
        printf("\n--- MENU ---");
        printf("\n1. Push (Stack)");
        printf("\n2. Pop (Stack)");
        printf("\n3. Display Stack");
        printf("\n4. Enqueue (Queue)");
        printf("\n5. Dequeue (Queue)");
        printf("\n6. Display Queue");
        printf("\n7. Exit");
        printf("\nEnter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter data: ");
                scanf("%d", &data);
                push(data);
                break;
            case 2:
                pop();
                break;
            case 3:
                displayStack();
                break;
            case 4:
                printf("Enter data: ");
                scanf("%d", &data);
                enqueue(data);
                break;
            case 5:
```

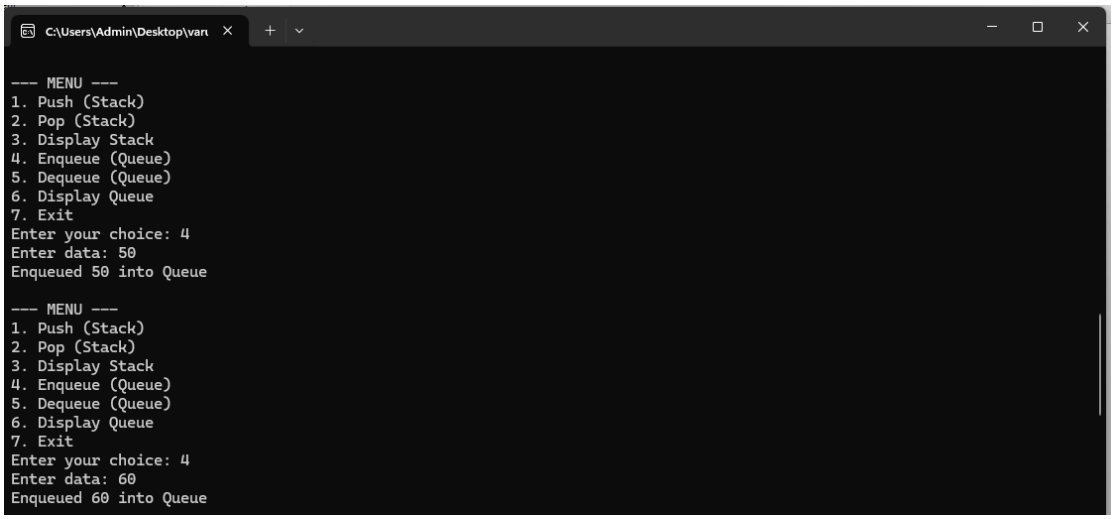
```
            dequeue();
            break;
        case 6:
            displayQueue();
            break;
        case 7:
            printf("Exiting program...\n");
            exit(0);
        default:
            printf("Invalid choice\n");
      }
   } while (1);

   return 0;
}
```

Output:






C:\Users\Admin\Desktop\var
--- 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 element: 30
--- 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: 20 -> 10 -> NULL


C:\Users\Admin\Desktop\var
--- 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 data: 50
Enqueued 50 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 data: 60
Enqueued 60 into Queue


C:\Users\Admin\Desktop\var
--- 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 data: 70
Enqueued 70 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: 50 -> 60 -> 70 -> NULL


C:\Users\Admin\Desktop\van
--- 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 element: 50
--- 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: 60 -> 70 -> 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: 6
Queue: 60 -> 70 -> 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: 7
Exiting program...
Process returned 0 (0x0) execution time : 80.080 s
Press any key to continue.