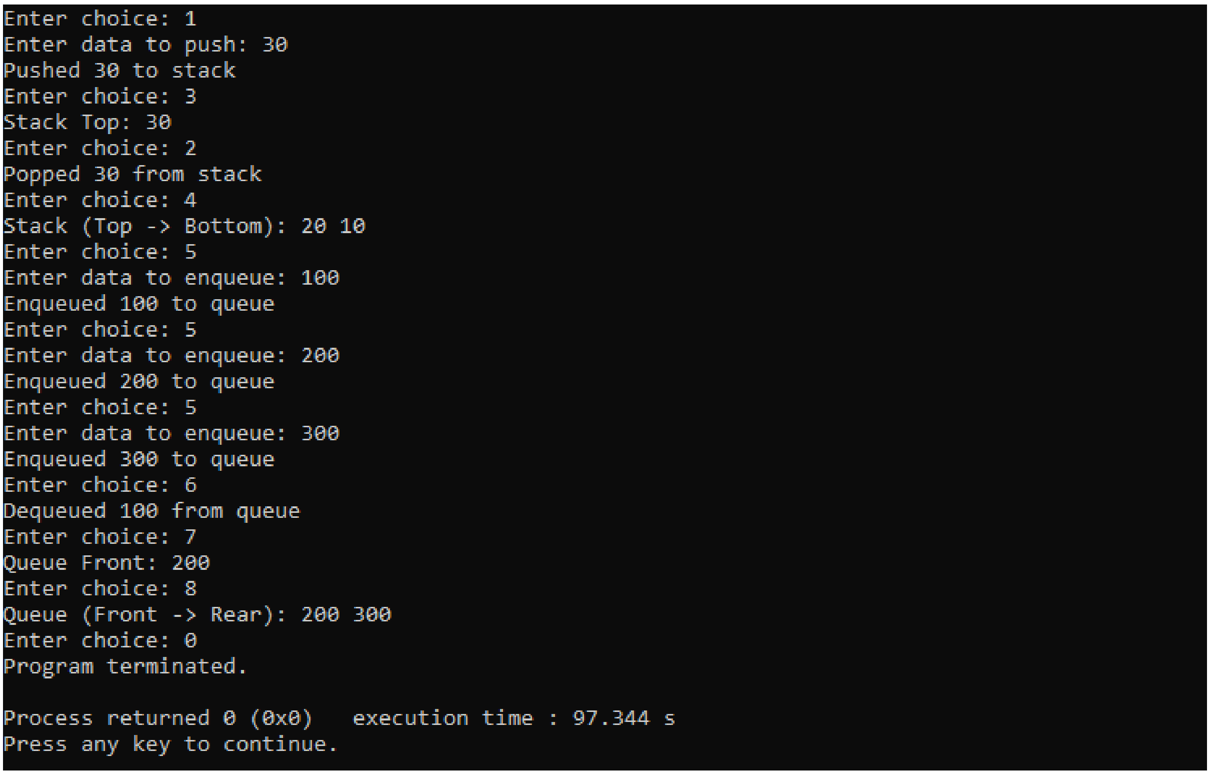
Implement Single Link List to simulate Stack & Queue Operations.

```
#include <stdio.h>

#include <stdlib.h>

struct Node {

    int data;

    struct Node* next;

};


struct Node* stackTop = NULL;

void stackPush(int value) {

    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));

    newNode->data = value;

    newNode->next = stackTop;

    stackTop = newNode;

}

int stackPop() {

    if (stackTop == NULL) {

        printf("Stack Underflow\n");

        return -1;

    }

    struct Node* temp = stackTop;

    int popped = temp->data;

    stackTop = stackTop->next;

    free(temp);
    return popped;
}

int stackPeek() {
    if (stackTop == NULL) {
        printf("Stack is empty\n");
        return -1;
    }
    return stackTop->data;
}

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

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;
        return;
    }
    rear->next = newNode;
    rear = newNode;
}

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

    if (front == rear) {
        front = rear = NULL;
    } else {
        front = front->next;
    }
    free(temp);
    return dequeued;
}

int queueFront() {
    if (front == NULL) {
        printf("Queue is empty\n");
        return -1;
    }
    return front->data;
}

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

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

int main() {
    int choice, data, cont = 1;

    printf("=== Singly Linked List: Stack & Queue Operations ===\n");

    while (cont) {
        printf("\n--- MENU ---\n");
        printf("1. Stack Push\n");
        printf("2. Stack Pop\n");
        printf("3. Stack Peek\n");
        printf("4. Stack Display\n");
        printf("5. Queue Enqueue\n");
        printf("6. Queue Dequeue\n");
        printf("7. Queue Front\n");
        printf("8. Queue Display\n");
        printf("0. Exit\n");
printf("Enter choice: ");
scanf("%d", &choice);

switch (choice) {
    case 1:
        printf("Enter data to push: ");
        scanf("%d", &data);
        stackPush(data);
        printf("Pushed %d to stack\n", data);
        break;

    case 2:
        data = stackPop();
        if (data != -1) printf("Popped %d from stack\n", data);
        break;

    case 3:
        data = stackPeek();
        if (data != -1) printf("Stack Top: %d\n", data);
        break;

    case 4:
        displayStack();
        break;
```

```
case 5:
    printf("Enter data to enqueue: ");
    scanf("%d", &data);
    enqueue(data);
    printf("Enqueued %d to queue\n", data);
    break;

case 6:
    data = dequeue();
    if (data != -1) printf("Dequeued %d from queue\n", data);
    break;

case 7:
    data = queueFront();
    if (data != -1) printf("Queue Front: %d\n", data);
    break;

case 8:
    displayQueue();
    break;

case 0:
    cont = 0;
    break;
```

```
            default:
                printf("Invalid choice!\n");
        }
    }
    printf("Program terminated.\n");
    return 0;
}
```

Output:



```
Enter choice: 1
Enter data to push: 30
Pushed 30 to stack
Enter choice: 3
Stack Top: 30
Enter choice: 2
Popped 30 from stack
Enter choice: 4
Stack (Top -> Bottom): 20 10
Enter choice: 5
Enter data to enqueue: 100
Enqueued 100 to queue
Enter choice: 5
Enter data to enqueue: 200
Enqueued 200 to queue
Enter choice: 5
Enter data to enqueue: 300
Enqueued 300 to queue
Enter choice: 6
Dequeued 100 from queue
Enter choice: 7
Queue Front: 200
Enter choice: 8
Queue (Front -> Rear): 200 300
Enter choice: 0
Program terminated.

Process returned 0 (0x0)   execution time : 97.344 s
Press any key to continue.
```