Assignment: Stack and Queue Implementation in C

Objective:
To understand the concepts of Stack and Queue data structures and implement them using C programming. This assignment includes complete code, sample outputs, and elaborative explanations suitable for training and academic submission.

Part A: Stack Data Structure

Definition:
A Stack is a linear data structure that follows the LIFO (Last In First Out) principle. The last element inserted into the stack is the first one to be removed.

Stack Operations

- Push – Insert an element
- Pop – Remove an element
- Peek – View top element

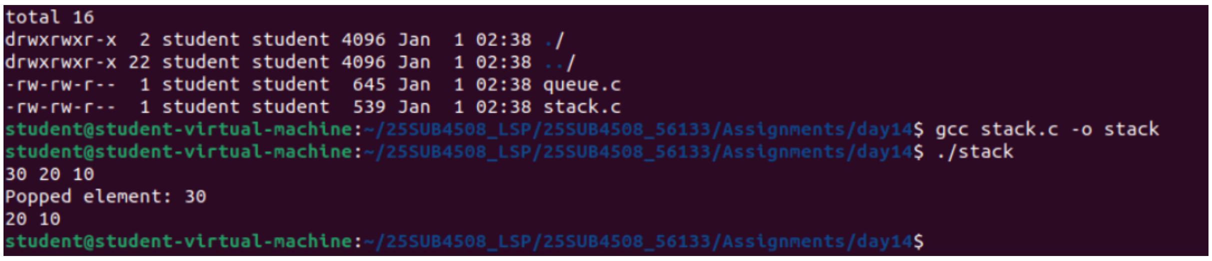
Stack Program Code (C)

```
#include <stdio.h>
 #define MAX 5

 int stack[MAX];
 int top = -1;

 void push(int val) {
           if (top == MAX - 1)
           printf("Stack Overflow\n");
           else
           stack[++top] = val;
}

 void pop() {
           if (top == -1)
           printf("Stack Underflow\n");
           else
```

```
        printf("Popped element: %d\n", stack[top--]);
}

void display() {
        for (int i = top; i >= 0; i--)
        printf("%d ", stack[i]);
        printf("\n");
}

int main() {
        push(10);
        push(20);
        push(30);
        display();
        pop();
        display();
        return 0;
}
```

Stack Output

```
30 20 10
 Popped element: 30
 20 10
```



```
total 16
drwxrwxr-x  2 student student 4096 Jan  1 02:38 ./
drwxrwxr-x 22 student student 4096 Jan  1 02:38 ../
-rw-rw-r--  1 student student  645 Jan  1 02:38 queue.c
-rw-rw-r--  1 student student  539 Jan  1 02:38 stack.c
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/Assignments/day14$ gcc stack.c -o stack
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/Assignments/day14$ ./stack
30 20 10
Popped element: 30
20 10
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/Assignments/day14$
```

Stack Explanation

• Stack grows from bottom to top
• Push increments the top index
• Pop decrements the top index
• Access is restricted to the top element

Part B: Queue Data Structure

Definition:
A Queue is a linear data structure that follows the FIFO (First In First Out) principle. The first element inserted is the first one to be removed.

Queue Operations

• Enqueue – Insert an element
• Dequeue – Remove an element

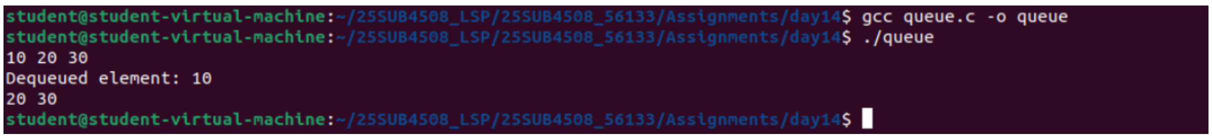
Queue Program Code (C)

```
#include <stdio.h>
 #define MAX 5

 int queue[MAX];
 int front = -1, rear = -1;

 void enqueue(int val) {
         if (rear == MAX - 1)
         printf("Queue Overflow\n");
         else {
         if (front == -1) front = 0;
         queue[++rear] = val;
         }
 }

 void dequeue() {
         if (front == -1 || front > rear)
         printf("Queue Underflow\n");
         else
         printf("Dequeued element: %d\n", queue[front++]);
 }

 void display() {
         for (int i = front; i <= rear; i++)
         printf("%d ", queue[i]);
         printf("\n");
 }

 int main() {
         enqueue(10);
```

```
        enqueue(20);
        enqueue(30);
        display();
        dequeue();
        display();
        return 0;
}
```

Queue Output

```
10 20 30
 Dequeued element: 10
 20 30
```



```
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/Assignments/day14$ gcc queue.c -o queue
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/Assignments/day14$ ./queue
10 20 30
Dequeued element: 10
20 30
student@student-virtual-machine:~/25SUB4508_LSP/25SUB4508_56133/Assignments/day14$ 
```

Queue Explanation

• Queue inserts at rear and removes from front
• FIFO order is maintained
• Front and rear pointers manage operations

Conclusion

This assignment demonstrates the implementation and working of Stack and Queue data structures using C. These data structures are fundamental and widely used in operating systems, scheduling, recursion, and data processing.