Experiment No.5:

Stack or Queue using Array (Static Implementation)
Simulate a parcel handling system at a post office where packages are stacked (LIFO) orqueued (FIFO). Use an array to implement a stack (push, pop, display) or a queue (add, delete, display). Choose the appropriate model based on the scenario.

```c
#include <stdio.h>
#define SIZE 5
// Stack variables
int stack[SIZE];
int top = -1;
// Queue variables
int queue[SIZE];
int front = -1, rear = -1;

// Stack Operations
void push(int item) {
   if (top == SIZE - 1)
      printf("Stack is full!\n");
   else {
      top++;
      stack[top] = item;
      printf("Parcel %d pushed onto the stack.\n", item);
   }
}

void pop() {
   if (top == -1)
      printf("Stack is empty!\n");
   else {
      printf("Parcel %d popped from the stack.\n", stack[top]);
      top--;
   }
}

void displayStack() {
   if (top == -1)
      printf("Stack is empty.\n");
   else {
      printf("Parcels in Stack:\n");
      for (int i = top; i >= 0; i--) {
         printf("%d ", stack[i]);
      }
```

```c
        printf("\n");
    }
}

// Queue Operations
void enqueue(int item) {
    if (rear == SIZE - 1)
        printf("Queue is full!\n");
    else {
        if (front == -1) front = 0;
        rear++;
        queue[rear] = item;
        printf("Parcel %d added to the queue.\n", item);
    }
}

void dequeue() {
    if (front == -1 || front > rear)
        printf("Queue is empty!\n");
    else {
        printf("Parcel %d removed from the queue.\n", queue[front]);
        front++;
    }
}

void displayQueue() {
    if (front == -1 || front > rear)
        printf("Queue is empty.\n");
    else {
        printf("Parcels in Queue:\n");
        for (int i = front; i <= rear; i++) {
            printf("%d ", queue[i]);
        }
        printf("\n");
    }
}

int main() {
    int choice, model, item;

    printf("Parcel Handling System\n");
    printf("Choose Model:\n1. Stack (LIFO)\n2. Queue (FIFO)\n");
    printf("Enter choice: ");
    scanf("%d", &model);
    if (model == 1) {
        // Stack menu
        do {
            printf("\n--- Stack Operations ---\n");
            printf("1. Push Parcel\n2. Pop Parcel\n3. Display Stack\n4. Exit\n");
            printf("Enter choice: ");
            scanf("%d", &choice);

            switch (choice) {
                case 1:
                    printf("Enter parcel ID to push: ");
                    scanf("%d", &item);
                    push(item);
                    break;
                case 2:
                    pop();
                    break;
                case 3:
                    displayStack();
                    break;
                case 4:
                    printf("Exiting Stack operations.\n");
                    break;
                default:
                    printf("Invalid choice.\n");
            }
        } while (choice != 4);
    } else if (model == 2) {
        // Queue menu
        do {
            printf("\n--- Queue Operations ---\n");
            printf("1. Add Parcel (Enqueue)\n2. Remove Parcel (Dequeue)\n3. Display Queue\n4.
Exit\n");
            printf("Enter choice: ");
            scanf("%d", &choice);

            switch (choice) {
                case 1:
                    printf("Enter parcel ID to enqueue: ");
                    scanf("%d", &item);
                    enqueue(item);
                    break;
                case 2:
```

```
                dequeue();
                break;
            case 3:
                displayQueue();
                break;
            case 4:
                printf("Exiting Queue operations.\n");
                break;
            default:
                printf("Invalid choice.\n");
            }
        } while (choice != 4);
    } else {
        printf("Invalid model selection.\n");
    }

    return 0;
}
```