Lab 2

Assign	
Status	Labs

Question 3

To obtain the time complexity of the banker's algorithm, we first examine the following.

- The resource request algorithm.

  Running the resource request algorithm, we have to traverse through three arrays of the same size. Hence, taking two for loops to do so in the requestResources function. To check the request against the need and the request against the available would teach take O(n) time complexity. To modify the state, it would take another O(3n) to go through the available, allocation and need.

- The safety algorithm.

  Traversing two arrays that are multidimensional, We go through one outer loop of the finish array of length n and the inner loop for the available matrix of size m. Hence, to check that each of the elements has finished in the finish array, the number of times we have to loop is the number of processes, which in this case is the number of customers. This would take **O(number of customers)** time complexity. We do a check within each process present in the tempNeed array for each resources allocated for each process. This ensures the 'safe' portion that for each process, the need to each to happen is less than the available supplied by the OS. This would take another inner multiplication of **O(number of resources for each customer**) time complexity. Intuitively, we also need to include the outer while loop that takes note that the task is done. Hence, this would require another **O(number of resources)**. We also include the other for loops for checking the finish array for each process that all processes are done that have a time complexity of O(number of processes) that are not within the same loops.

  Due to the checking of whether each task is completed, we obtain a total time complexity of the upper bound of

$$T(n) = O(nm^2) + O(3n) + O(n) + O(m) = O(nm^2)$$

where n is the number of processes while m is the number of resources.

This is the higher time complexity than the resource request algorithm. Hence, the safety algorithm time complexity takes higher precedence.

Proof of higher time complexity in code:

```
// in the checkSafe function

//portion with higher time complexity
int possible = 1;
    while(possible) {
        possible = 0;
        for (int i = 0; i < numberOfCustomers; i++) {
            if (finish[i] == 0) {
                for (int j = 0; j < numberOfResources; j++) {
                    if (tempNeed[i][j] < work[j]) {
                        work[j] += tempAllocation[i][j];
                        finish[i] = 1;
                        possible = 1;
                    }
                }
            }
        }
    }
```