Analysis of Banker's Algorithm

Let m be the totla number of resource types

Let n be the total number of customers

Consider this snippet of code from the Banker's Algorithm

```
        if (finish(Ci) == false && temp_need(Ci) <= work) {
                possible = true;
                work += temp_allocation(Ci);
                finish(Ci) = true;
        }
```

Temp_need and work are 1 dimensional arrays of size m. Thus the comparison of temp_need and work [`temp_need(Ci) <= work`] and updating of the work array [`work += temp_allocation(Ci)`] each have complexity m.

Thus, this if loop has complexity O(m+m) = O(2m) = O(m)

This if loop is nested within a for loop

```
for(customer Ci = 1:n) {
        if (finish(Ci) == false && temp_need(Ci) <= work) {
                possible = true;
                work += temp_allocation(Ci);
                finish(Ci) = true;
        }
}
```

The for loop iterates through all customers [`(customer Ci = 1:n)`], hence it would iterate n times. Thus the for loop would have a complexity of O(n*m).

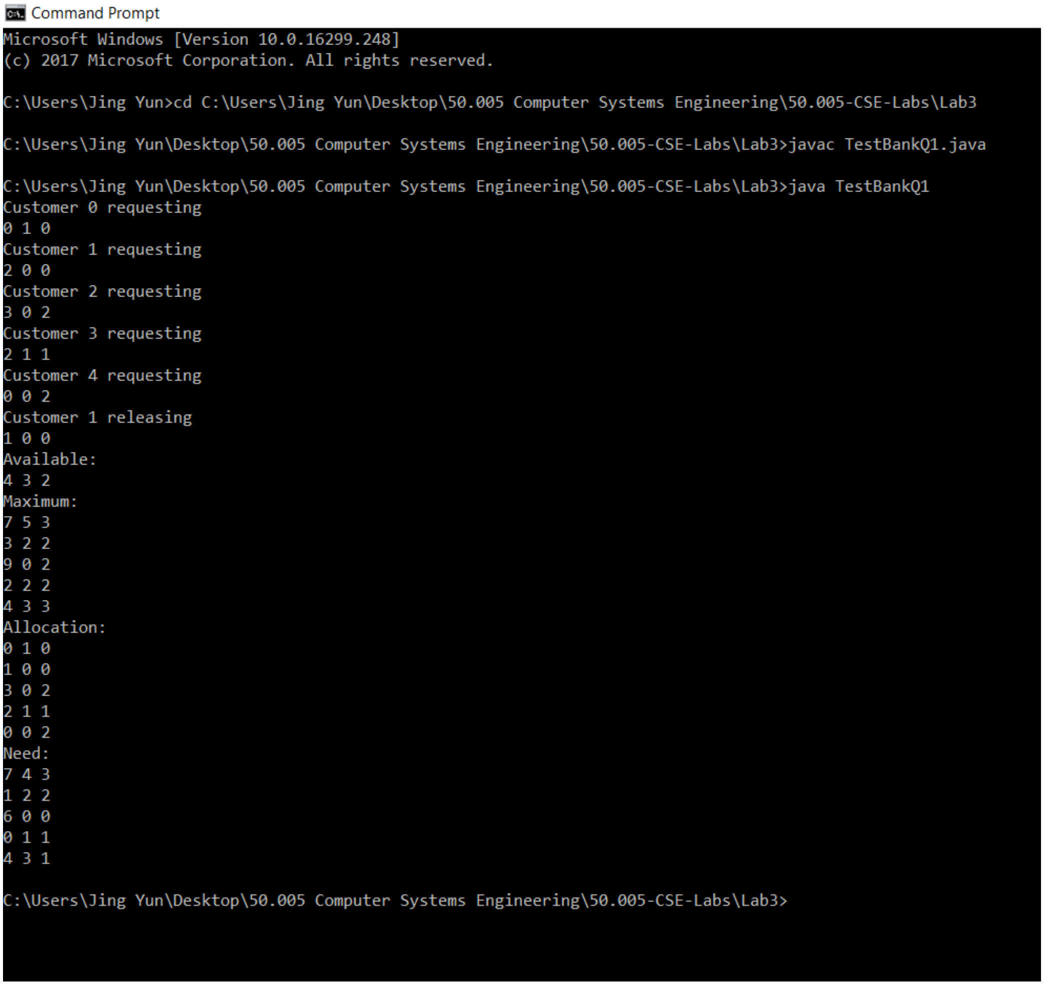
This for loop is in turn nested in a while loop

```
while(possible) {
        possible = false;
        for(customer Ci = 1:n) {
                if (finish(Ci) == false && temp_need(Ci) <= work) {
                        possible = true;
                        work += temp_allocation(Ci);
                        finish(Ci) = true;
                }
        }
}
```

In the worst case the while loop would iterate n times.

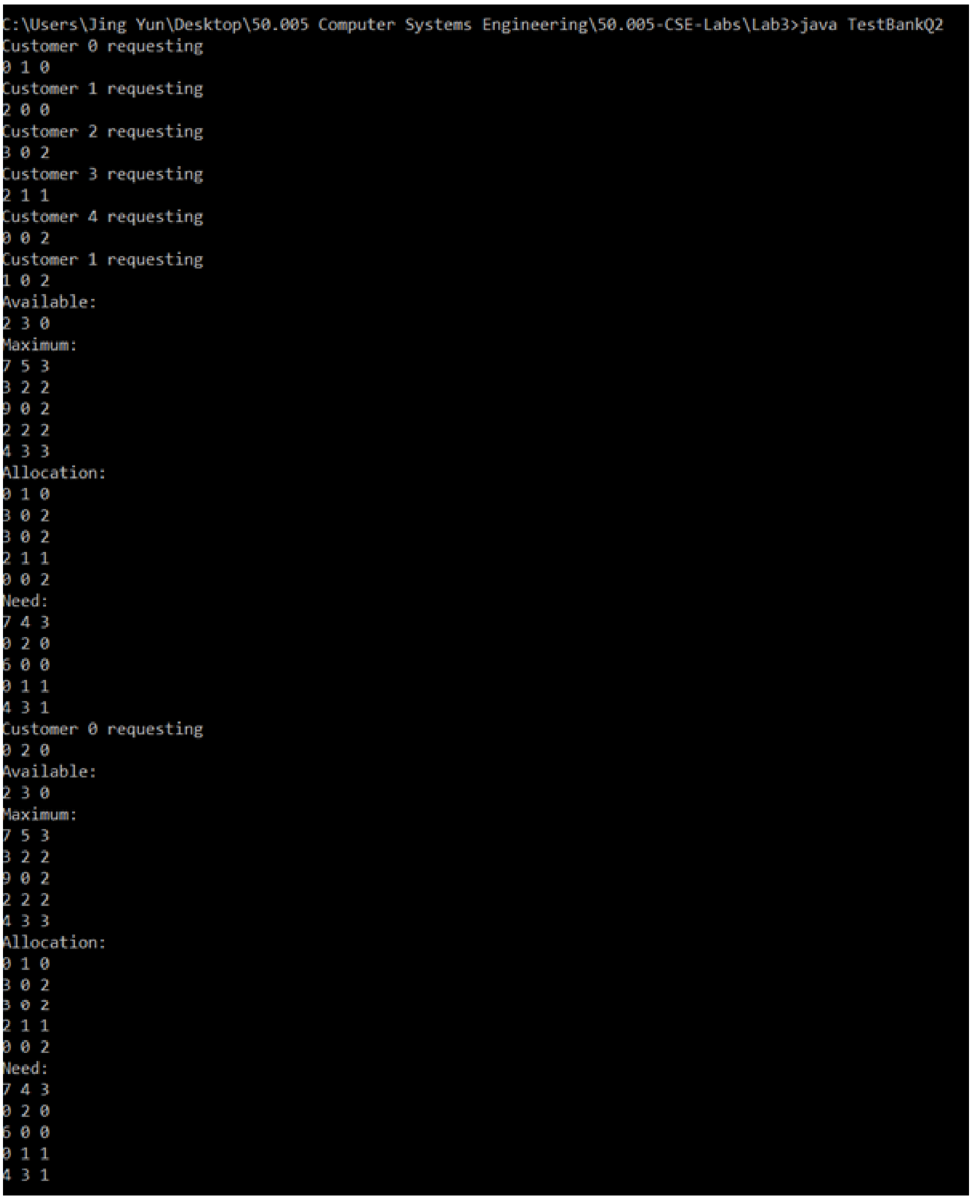
Thus, the complexity of Banker's Algorithm is **O(m*n*n)**

Test Bank Q1



```
Microsoft Windows [Version 10.0.16299.248]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\Jing Yun>cd C:\Users\Jing Yun\Desktop\50.005 Computer Systems Engineering\50.005-CSE-Labs\Lab3

C:\Users\Jing Yun\Desktop\50.005 Computer Systems Engineering\50.005-CSE-Labs\Lab3>javac TestBankQ1.java

C:\Users\Jing Yun\Desktop\50.005 Computer Systems Engineering\50.005-CSE-Labs\Lab3>java TestBankQ1
Customer 0 requesting
0 1 0
Customer 1 requesting
2 0 0
Customer 2 requesting
3 0 2
Customer 3 requesting
2 1 1
Customer 4 requesting
0 0 2
Customer 1 releasing
1 0 0
Available:
4 3 2
Maximum:
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Allocation:
0 1 0
1 0 0
3 0 2
2 1 1
0 0 2
Need:
7 4 3
1 2 2
6 0 0
0 1 1
4 3 1

C:\Users\Jing Yun\Desktop\50.005 Computer Systems Engineering\50.005-CSE-Labs\Lab3>
```

TestBankQ2



```
C:\Users\Jing Yun\Desktop\50.005 Computer Systems Engineering\50.005-CSE-Labs\Lab3>java TestBankQ2
Customer 0 requesting
0 1 0
Customer 1 requesting
2 0 0
Customer 2 requesting
3 0 2
Customer 3 requesting
2 1 1
Customer 4 requesting
0 0 2
Customer 1 requesting
1 0 2
Available:
2 3 0
Maximum:
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Allocation:
0 1 0
3 0 2
3 0 2
2 1 1
0 0 2
Need:
7 4 3
0 2 0
6 0 0
0 1 1
4 3 1
Customer 0 requesting
0 2 0
Available:
2 3 0
Maximum:
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Allocation:
0 1 0
3 0 2
3 0 2
2 1 1
0 0 2
Need:
7 4 3
0 2 0
6 0 0
0 1 1
4 3 1
```