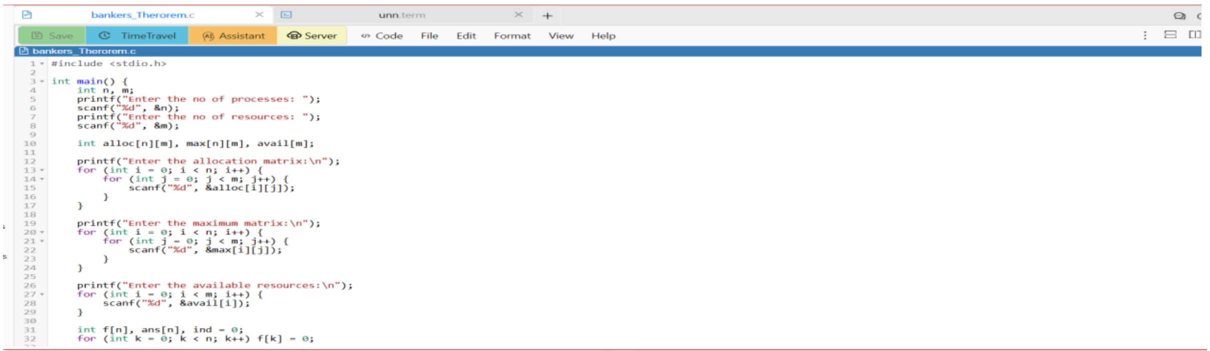
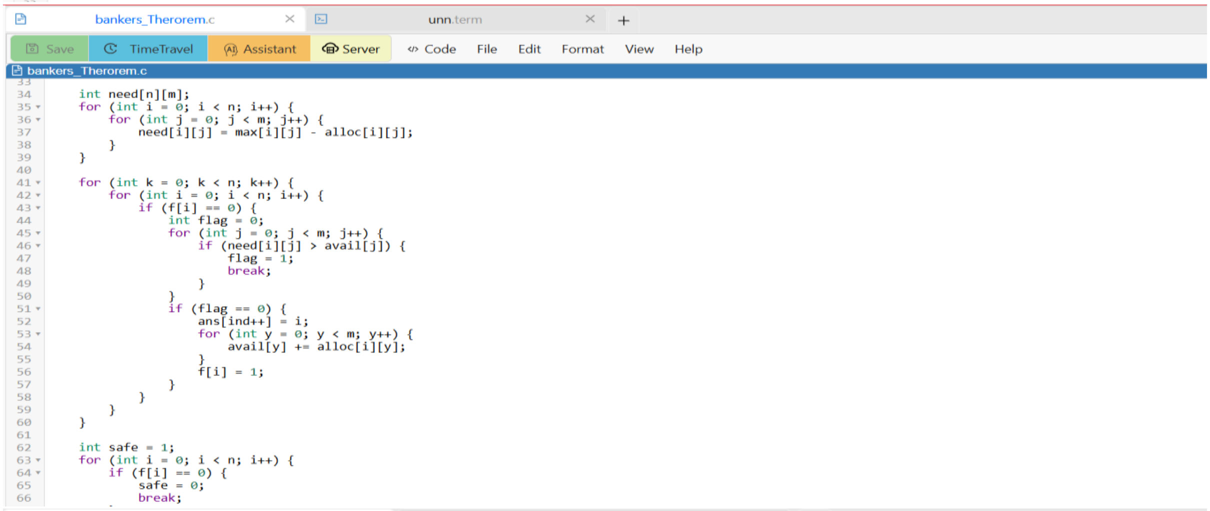
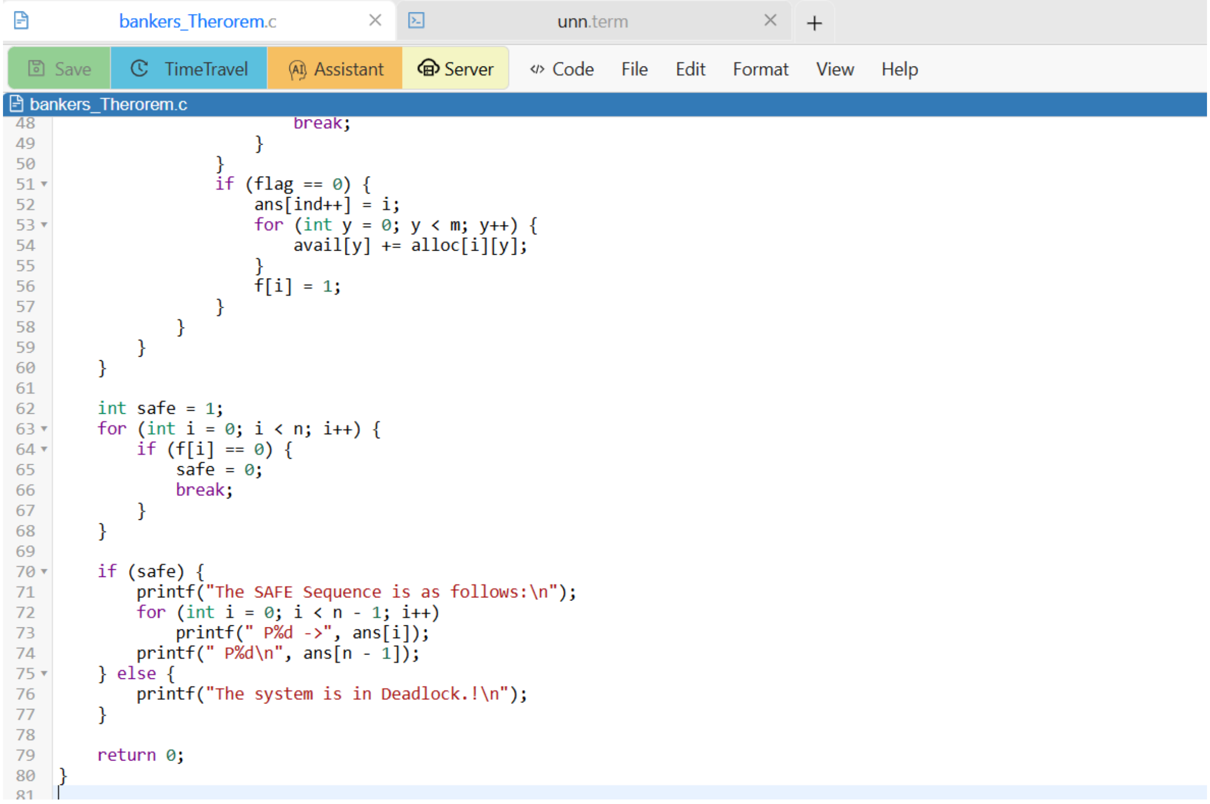
Bankers Theorem Yash Singhal 2400290120285



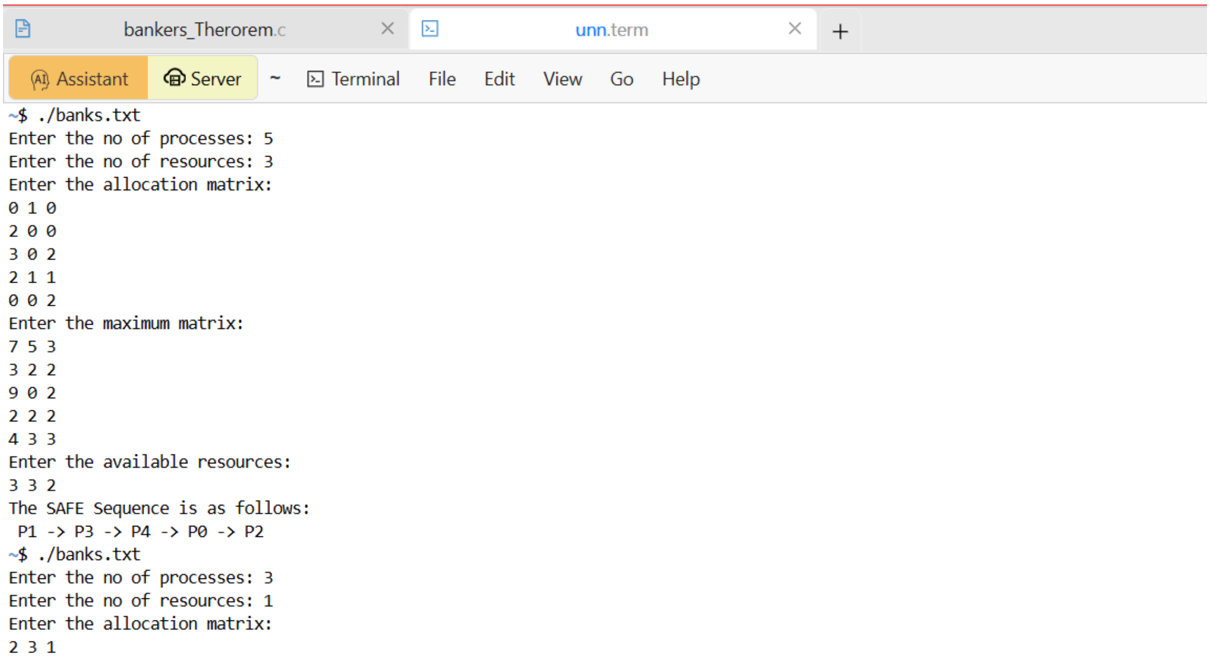
```c
#include <stdio.h>

int main() {
    int n, m;
    printf("Enter the no of processes: ");
    scanf("%d", &n);
    printf("Enter the no of resources: ");
    scanf("%d", &m);

    int alloc[n][m], max[n][m], avail[m];

    printf("Enter the allocation matrix:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            scanf("%d", &alloc[i][j]);
        }
    }

    printf("Enter the maximum matrix:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            scanf("%d", &max[i][j]);
        }
    }

    printf("Enter the available resources:\n");
    for (int i = 0; i < m; i++) {
        scanf("%d", &avail[i]);
    }

    int f[n], ans[n], ind = 0;
    for (int k = 0; k < n; k++) f[k] = 0;
```



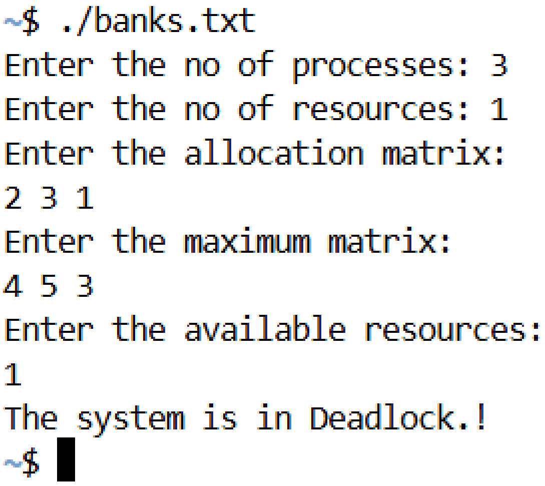
```c

    int need[n][m];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            need[i][j] = max[i][j] - alloc[i][j];
        }
    }

    for (int k = 0; k < n; k++) {
        for (int i = 0; i < n; i++) {
            if (f[i] == 0) {
                int flag = 0;
                for (int j = 0; j < m; j++) {
                    if (need[i][j] > avail[j]) {
                        flag = 1;
                        break;
                    }
                }
                if (flag == 0) {
                    ans[ind++] = i;
                    for (int y = 0; y < m; y++) {
                        avail[y] += alloc[i][y];
                    }
                    f[i] = 1;
                }
            }
        }
    }

    int safe = 1;
    for (int i = 0; i < n; i++) {
        if (f[i] == 0) {
            safe = 0;
            break;
```



```c
                        break;
                    }
                }
                if (flag == 0) {
                    ans[ind++] = i;
                    for (int y = 0; y < m; y++) {
                        avail[y] += alloc[i][y];
                    }
                    f[i] = 1;
                }
            }
        }
    }

    int safe = 1;
    for (int i = 0; i < n; i++) {
        if (f[i] == 0) {
            safe = 0;
            break;
        }
    }

    if (safe) {
        printf("The SAFE Sequence is as follows:\n");
        for (int i = 0; i < n - 1; i++)
            printf(" P%d ->", ans[i]);
        printf(" P%d\n", ans[n - 1]);
    } else {
        printf("The system is in Deadlock.!\n");
    }

    return 0;
}

```



```
~$ ./banks.txt
Enter the no of processes: 5
Enter the no of resources: 3
Enter the allocation matrix:
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2
Enter the maximum matrix:
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Enter the available resources:
3 3 2
The SAFE Sequence is as follows:
 P1 -> P3 -> P4 -> P0 -> P2
~$ ./banks.txt
Enter the no of processes: 3
Enter the no of resources: 1
Enter the allocation matrix:
2 3 1
```



```
~$ ./banks.txt
Enter the no of processes: 3
Enter the no of resources: 1
Enter the allocation matrix:
2 3 1
Enter the maximum matrix:
4 5 3
Enter the available resources:
1
The system is in Deadlock.!
~$
```