Name – Bramha Nimbalkar
Roll no- 7
Div - E

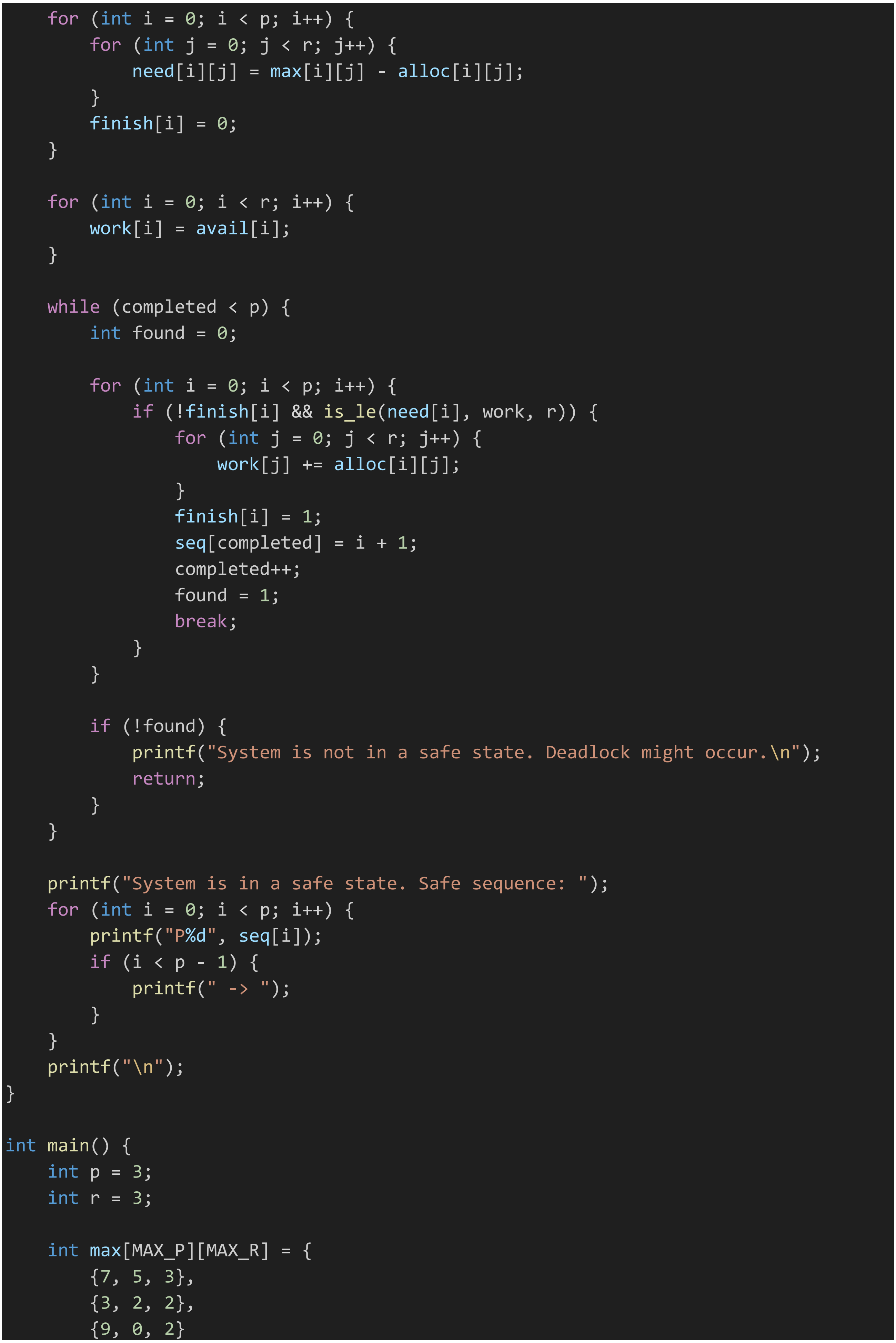
ASSIGNMENT 8

Problem : Implement in C language Banker’s Deadlock Avoidance Algorithm.

Algorithm

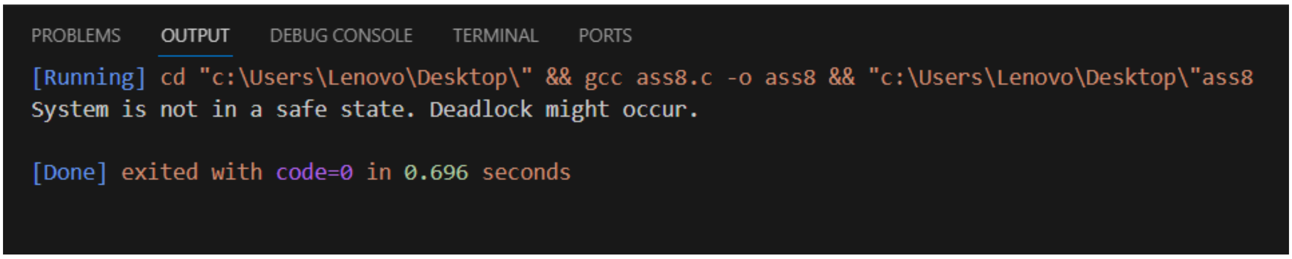
```
1. Start.
2. Calculate the current need for each process, for each resource from the data entered by user.
3. For a process, check if current available resources satisfy all current needs.
4. If all are satisfied completes the process and adds all its current allocations to available resources.
5. If all are not satisfied, check for next process.
6. Repeat from step 3, for as many times as there are processes.
7. If all processes have been completed, system is in safe state and display safe sequence. Else system is not in safe state.
8. Stop.
```

```c
#include <stdio.h>

#define MAX_P 3
#define MAX_R 3

int is_le(int x[], int y[], int r) {
    for (int i = 0; i < r; i++) {
        if (x[i] > y[i]) {
            return 0;
        }
    }
    return 1;
}

void bankers_algo(int p, int r, int max[MAX_P][MAX_R], int alloc[MAX_P][MAX_R], int 
avail[MAX_R]) {
    int need[MAX_P][MAX_R];
    int work[MAX_R];
    int finish[MAX_P];
    int seq[MAX_P];
    int completed = 0;
```



```c
    for (int i = 0; i < p; i++) {
        for (int j = 0; j < r; j++) {
            need[i][j] = max[i][j] - alloc[i][j];
        }
        finish[i] = 0;
    }

    for (int i = 0; i < r; i++) {
        work[i] = avail[i];
    }

    while (completed < p) {
        int found = 0;

        for (int i = 0; i < p; i++) {
            if (!finish[i] && is_le(need[i], work, r)) {
                for (int j = 0; j < r; j++) {
                    work[j] += alloc[i][j];
                }
                finish[i] = 1;
                seq[completed] = i + 1;
                completed++;
                found = 1;
                break;
            }
        }

        if (!found) {
            printf("System is not in a safe state. Deadlock might occur.\n");
            return;
        }
    }

    printf("System is in a safe state. Safe sequence: ");
    for (int i = 0; i < p; i++) {
        printf("P%d", seq[i]);
        if (i < p - 1) {
            printf(" -> ");
        }
    }
    printf("\n");
}

int main() {
    int p = 3;
    int r = 3;

    int max[MAX_P][MAX_R] = {
        {7, 5, 3},
        {3, 2, 2},
        {9, 0, 2}
    };


    int alloc[MAX_P][MAX_R] = {
        {0, 1, 0},
        {2, 0, 0},
        {3, 0, 2}
    };



    int avail[MAX_R] = {3, 3, 3};


    bankers_algo(p, r, max, alloc, avail);

    return 0;
}
```

Output



```
[Running] cd "c:\Users\Lenovo\Desktop\" && gcc ass8.c -o ass8 && "c:\Users\Lenovo\Desktop\"ass8
System is not in a safe state. Deadlock might occur.

[Done] exited with code=0 in 0.696 seconds
```