Name: Nikhil Gupta
Roll No.: 20051523

LAB ASSIGNMENT 12

Bankers Algorithm

```
#include <stdio.h>
#include <stdbool.h>

struct nikhil_process_info
{
    int nikhil_max[10];
    int nikhil_allocated[10];
    int nikhil_need[10];
};

int nikhil_no_of_process, nikhil_no_of_resources;

void input(struct nikhil_process_info
nikhil_process[nikhil_no_of_process], int
nikhil_available[nikhil_no_of_resources])
{

    for (int i = 0; i < nikhil_no_of_process; i++)
    {
        printf("Enter nikhil_process[%d] info\n", i);
        printf("Enter Maximum Need: ");
        for (int j = 0; j < nikhil_no_of_resources; j++)
            scanf("%d",
&nikhil_process[i].nikhil_max[j]);
        printf("Enter No. of Allocated Resources for this
nikhil_process: ");
        for (int j = 0; j < nikhil_no_of_resources; j++)
        {
            scanf("%d",
&nikhil_process[i].nikhil_allocated[j]);
```

Name: Nikhil Gupta
Roll No.: 20051523

```
            nikhil_process[i].nikhil_need[j] =
nikhil_process[i].nikhil_max[j] -
nikhil_process[i].nikhil_allocated[j];
        }
    }
    printf("Enter Available Resources: ");
    for (int i = 0; i < nikhil_no_of_resources; i++)
    {
        scanf("%d", &nikhil_available[i]);
    }
}

void showTheInfo(struct nikhil_process_info
nikhil_process[nikhil_no_of_process])
{
    printf("\nPID\tMaximum\t\tAllocated\tNeed\n");
    for (int i = 0; i < nikhil_no_of_process; i++)
    {
        printf("P[%d]\t", i);
        for (int j = 0; j < nikhil_no_of_resources; j++)
            printf("%d ",
nikhil_process[i].nikhil_max[j]);
        printf("\t\t");
        for (int j = 0; j < nikhil_no_of_resources; j++)
            printf("%d ",
nikhil_process[i].nikhil_allocated[j]);
        printf("\t\t");
        for (int j = 0; j < nikhil_no_of_resources; j++)
            printf("%d ",
nikhil_process[i].nikhil_need[j]);
        printf("\n");
    }
}

bool applySafetyAlgo(struct nikhil_process_info
nikhil_process[nikhil_no_of_process], int
nikhil_available[nikhil_no_of_resources], int
nikhil_safeSequence[nikhil_no_of_process])
{
    bool nikhil_finish[nikhil_no_of_process];
```

Name: Nikhil Gupta
Roll No.: 20051523

```
    int nikhil_work[nikhil_no_of_resources];
    for (int i = 0; i < nikhil_no_of_resources; i++)
    {
        nikhil_work[i] = nikhil_available[i];
    }
    for (int i = 0; i < nikhil_no_of_process; i++)
        nikhil_finish[i] = false;
    bool nikhil_proceed = true;
    int k = 0;
    while (nikhil_proceed)
    {
        nikhil_proceed = false;
        for (int i = 0; i < nikhil_no_of_process; i++)
        {
            bool nikhil_flag = true;

            if (nikhil_finish[i] == false)
            {

                for (int j = 0; j <
nikhil_no_of_resources; j++)
                {

                    if (nikhil_process[i].nikhil_need[j]
<= nikhil_work[j])
                    {
                        continue;
                    }
                    else
                    {
                        nikhil_flag = false;
                        break;
                    }
                }
                if (nikhil_flag == false)
                    continue;

                for (int j = 0; j <
nikhil_no_of_resources; j++)
```

Name: Nikhil Gupta
Roll No.: 20051523

```
                    nikhil_work[j] = nikhil_work[j] +
nikhil_process[i].nikhil_allocated[j];
                nikhil_finish[i] = true;
                nikhil_safeSequence[k++] = i;
                nikhil_proceed = true;
            }
        }
    }

    int i;
    for (i = 0; i < nikhil_no_of_process &&
nikhil_finish[i] == true; i++)
    {
        continue;
    }

    if (i == nikhil_no_of_process)
        return true;
    else
        return false;
}

bool isSafeState(struct nikhil_process_info
nikhil_process[nikhil_no_of_process], int
nikhil_available[nikhil_no_of_resources], int
nikhil_safeSequence[nikhil_no_of_process])
{

    if (applySafetyAlgo(nikhil_process, nikhil_available,
nikhil_safeSequence) == true)
        return true;
    return false;
}

int main()
{
    printf("Enter No of Process\n");
    scanf("%d", &nikhil_no_of_process);
    printf("Enter No of Resource Instances in system\n");
    scanf("%d", &nikhil_no_of_resources);
```

Name: Nikhil Gupta
Roll No.: 20051523

```
    int nikhil_available[nikhil_no_of_resources];
    int nikhil_safeSequence[nikhil_no_of_process];

    struct nikhil_process_info
nikhil_process[nikhil_no_of_process];

    printf("***********Enter details of
processes************\n");

    input(nikhil_process, nikhil_available);

    showTheInfo(nikhil_process);
    if (isSafeState(nikhil_process, nikhil_available,
nikhil_safeSequence))
    {

        printf("\nSystem is in SAFE State\n");
        printf("Safe Sequence is: ");
        for (int i = 0; i < nikhil_no_of_process; i++)
            printf("P[%d] ", nikhil_safeSequence[i]);
        printf("1");
    }
    else
        printf("0");
    return 0;
}
```

Name: Nikhil Gupta
Roll No.: 20051523

OUTPUT

```
Enter No of Process
5
Enter No of Resource Instances in system
3
***********Enter details of processes*************
Enter nikhil_process[0] info
Enter Maximum Need: 7 5 3
Enter No. of Allocated Resources for this nikhil_process: 0 1 0
Enter nikhil_process[1] info
Enter Maximum Need: 3 2 2
Enter No. of Allocated Resources for this nikhil_process: 2 0 0
Enter nikhil_process[2] info
Enter Maximum Need: 9 0 2
Enter No. of Allocated Resources for this nikhil_process: 3 0 3
Enter nikhil_process[3] info
Enter Maximum Need: 4 2 2
Enter No. of Allocated Resources for this nikhil_process: 2 1 1
Enter nikhil_process[4] info
Enter Maximum Need: 5 3 3
Enter No. of Allocated Resources for this nikhil_process: 0 0 2
Enter Available Resources: 3 3 2

PID     Maximum         Allocated       Need
P[0]    7 5 3           0 1 0           7 4 3
P[1]    3 2 2           2 0 0           1 2 2
P[2]    9 0 2           3 0 3           6 0 -1
P[3]    4 2 2           2 1 1           2 1 1
P[4]    5 3 3           0 0 2           5 3 1

System is in SAFE State
Safe Sequence is: P[1] P[3] P[4] P[0] P[2] 1
PS E:\Mega Sync\Programming\C\Scheduling Algorithms>
```