Task 12: Write a program to check whether the given system is in a safe state or not using Banker's Deadlock Avoidance algorithm (input Takes by user).

```
#include <stdio.h>

void calculateNeed(int need[], int max[], int allocation[], int n) {
    for (int i = 0; i < n; i++)
        for (int j = 0; j < 3; j++)
            need[i * 3 + j] = max[i * 3 + j] - allocation[i * 3 + j];
}

int isSafe(int processes[], int avail[], int max[], int allocation[], int n) {
    int need[n * 3];
    calculateNeed(need, max, allocation, n);

    int finish[n];
    for (int i = 0; i < n; i++)
        finish[i] = 0;

    int work[3];
    for (int i = 0; i < 3; i++)
        work[i] = avail[i];

    int count = 0;
    while (count < n) {
        int found = 0;

        for (int i = 0; i < n; i++) {
            if (finish[i] == 0) {
                int j;
```

```
            for (j = 0; j < 3; j++) {
                if (need[i * 3 + j] > work[j])
                    break;
            }

            if (j == 3) {
                for (int k = 0; k < 3; k++)
                    work[k] += allocation[i * 3 + k];

                finish[i] = 1;
                found = 1;
                count++;
            }
        }
    }

    if (found == 0)
        return 0; // System is not in a safe state
  }

  return 1; // System is in a safe state
}

int main() {
  int n;

  printf("Enter the number of processes: ");
  scanf("%d", &n);

  int processes[n], max[n * 3], allocation[n * 3], avail[3];
```

```
    printf("Enter maximum resources for each process:\n");

    for (int i = 0; i < n; i++) {
        printf("Process %d: ", i + 1);
        for (int j = 0; j < 3; j++)
            scanf("%d", &max[i * 3 + j]);
    }

    printf("Enter allocated resources for each process:\n");

    for (int i = 0; i < n; i++) {
        printf("Process %d: ", i + 1);
        for (int j = 0; j < 3; j++)
            scanf("%d", &allocation[i * 3 + j]);
    }

    printf("Enter available resources:\n");

    for (int i = 0; i < 3; i++)
        scanf("%d", &avail[i]);

    if (isSafe(processes, avail, max, allocation, n))
        printf("The system is in a safe state.\n");
    else
        printf("The system is not in a safe state.\n");

    return 0;
}
```

Output-

```
Enter the number of processes: 2
Enter maximum resources for each process:
Process 1: 1 2
3
Process 2: 1 3 4
Enter allocated resources for each process:
Process 1: 1 2 3
Process 2: 4 5 6
Enter available resources:
2 3 4
The system is in a safe state.
```