Bankers Algorithms in C++:

Code:

```
#include<iostream>
#include<iostream>
#include <iomanip>
using namespace std;
class Process{
   int   total_resourses;
         int   total_prcesses;

         int   *max_resourses;
         int   *allocated;
         int   *available;

         bool  *running_processes;

         int   allocate_table[30][20];
         int   max_claim_process[30][20];



         void get_max_resoures(){
               cout<<"Enter the Total MAX resoures Instances"<<endl;
               for(int i=0;i<total_resourses;i++){
                     cout<<"Enter R"<<i+1<<" : ";cin>>max_resourses[i];
```

```
                }
                cout<<endl;
        }
        void print_max_resourses(){
                cout<<"Total MAX resoures Instances Are : "<<endl;
                for(int i=0;i<total_resourses;i++){
                        cout<<"R"<<i+1<<" :
"<<setiosflags(ios::left)<<setw(10)<<max_resourses[i];
                }
                cout<<endl;
                line(70);
        }
        void line(int n){
                for(int a=1;a<=n;a++){
                        cout <<"_";
                }
                cout<<endl;
        }
        void status_running_at_start(){
                for(int i=0 ;i<total_prcesses;i++){
                        running_processes[i] = true;
                }
        }
        void get_allocate_table(){
                cout<<"Enter the allocation table of the Processes !"<<endl;
```

```
        for(int i=0;i<total_prcesses;i++){
            cout<<"Process "<<i+1<<endl;
            for(int j=0;j<total_resourses;j++){
                cout<<"R"<<j+1<<" Allocated : ";cin>>allocate_table[i][j];
            }
            cout<<endl;
        }
    }
    void print_allocate_table(){
        cout<<"Allocated Table "<<endl;
        line(70);
        cout<<setiosflags(ios::left)<<setw(12)<<"Process";
        for(int p=1;p<=total_resourses;p++){
            cout<<"R"<<setiosflags(ios::left)<<setw(12)<<p;
        }
        cout<<endl;
        line(70);
        for(int i=0;i<total_prcesses;i++){

            cout<<setiosflags(ios::left)<<setw(12)<<i+1;
            for(int j=0;j<total_resourses;j++){
                cout<<setiosflags(ios::left)<<setw(12)<<allocate_table[i][j];
            }
            cout<<endl;
        }
```

```
                line(70);
        }
        void get_max_claim_table(){
                cout<<"Enter the MAX Claim table of the Processes !"<<endl;
                for(int i=0;i<total_prcesses;i++){
                        cout<<"Process "<<i+1<<endl;
                        for(int j=0;j<total_resourses;j++){
                                cout<<"R"<<j+1<<" MAX Claim :
";cin>>max_claim_process[i][j];
                        }
                        cout<<endl;
                }
        }
        void print_max_claim_table(){
                cout<<"MAX Claim Table "<<endl;
                line(70);
                cout<<setiosflags(ios::left)<<setw(12)<<"Process";
                for(int p=1;p<=total_resourses;p++){
                        cout<<"R"<<setiosflags(ios::left)<<setw(12)<<p;
                }
                cout<<endl;
                line(70);
                for(int i=0;i<total_prcesses;i++){

                        cout<<setiosflags(ios::left)<<setw(12)<<i+1;
```

```
                for(int j=0;j<total_resourses;j++){

        cout<<setiosflags(ios::left)<<setw(12)<<max_claim_process[i][j];
                }
                cout<<endl;
            }
            line(70);
        }

        void get_allocated_resourses(){
            for(int i=0;i<total_prcesses;i++){
                for(int j=0;j<total_resourses;j++){
                    allocated[j] += allocate_table[i][j];
                }
            }
        }
        void print_allocated_resoures(){
            cout<<"Allocated Resourse are : "<<endl;
                for(int i=0;i<total_resourses;i++){
                cout<<"R"<<i+1<<" :
"<<setiosflags(ios::left)<<setw(10)<<allocated[i];
            }
            cout<<endl;
            line(70);
        }
        void get_available_resourses(){
```

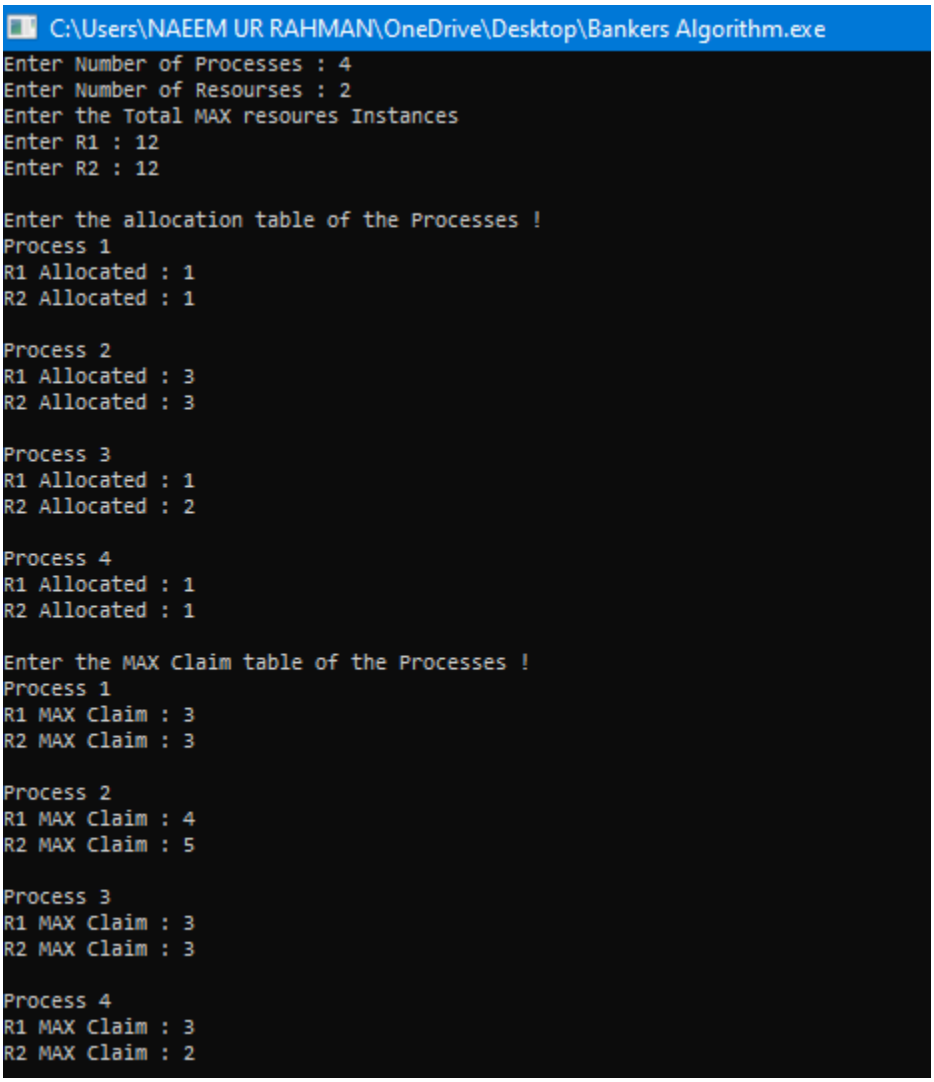
```
                for(int i=0;i<total_prcesses;i++){
                        for(int j=0;j<total_resourses;j++){
                                available[j] = max_resourses[j] - allocated[j];
                        }
                }
        }
        void print_available_resoures(){
                cout<<"Available Resourse are : "<<endl;
                        for(int i=0;i<total_resourses;i++){
                        cout<<"R"<<i+1<<" :
"<<setiosflags(ios::left)<<setw(10)<<available[i];
                }
                cout<<endl;
                line(70);
        }
        void set_array(int *value,int size){
                for(int i=0;i<size;i++){
                        value[i]=0;
                }
        }
        void check_save(){
                int safe,exec,count=total_prcesses;
                while (count != 0){
       safe = 0;
       for (int i = 0; i < total_prcesses; i++){
```

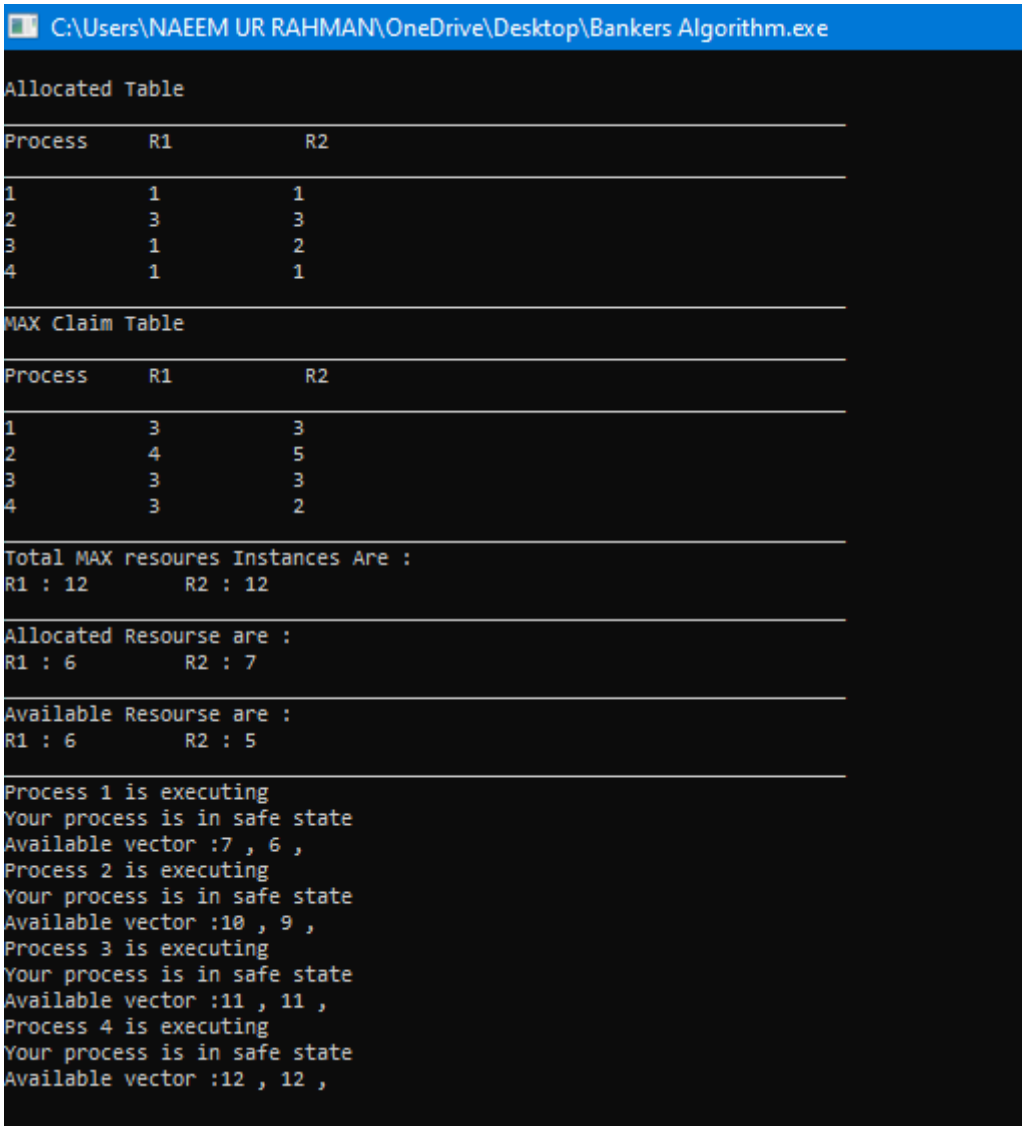
```
            if (running_processes[i]){
                exec = 1;
                for (int j = 0; j < total_resourses; j++){
                    if (max_claim_process[i][j] - allocate_table[i][j] > available[j]){
                        exec = 0;
                        break;
                    }
                }
                if (exec){


                                                cout<<"Process "<< i + 1<<" is
executing"<<endl;
                    running_processes[i] = false;
                    count--;
                    safe = 1;
                    for (int j = 0; j < total_resourses; j++){
                                                        available[j] += allocate_table[i][j];
                    }
                    break;
                }
            }
        }
        if (!safe) {


                            cout<<"The process is in unsafe state."<<endl;
```

```
            break;
        }
        else {


            cout<<"Your process is in safe state"<<endl;
            cout<<"Available vector :";


            for (int i = 0; i < total_resourses; i++)
            {
                cout<< available[i] <<"  ";
            }
            cout<<endl<<endl;
        }
    }
     }
     public :
     Process(int resourses,int processes){
                total_prcesses = processes;
                total_resourses = resourses;
                max_resourses = new int [total_resourses];
                allocated     = new int [total_resourses];
                available     = new int [total_resourses];
                set_array(allocated,total_resourses);
                set_array(available,total_resourses);
```

```
                get_max_resoures();


                running_processes = new bool[total_prcesses];
                status_running_at_start();


                get_allocate_table();
                get_max_claim_table();
                get_allocated_resourses();
                get_available_resourses();


                print_allocate_table();
                print_max_claim_table();
                print_max_resourses();
                print_allocated_resoures();
                print_available_resoures();
                check_save();
        }
};
int main(){
        int processes,resourses;
        cout<<"Enter Number of Processes : ";cin>>processes;
        cout<<"Enter Number of Resourses : ";cin>>resourses;
        Process p = Process(resourses,processes);
        return 0;
}
```

Output Example 1:




C:\Users\NAEEM UR RAHMAN\OneDrive\Desktop\Bankers Algorithm.exe
Allocated Table
Process R1 R2
1 1 1
2 3 3
3 1 2
4 1 1
MAX Claim Table
Process R1 R2
1 3 3
2 4 5
3 3 3
4 3 2
Total MAX resoures Instances Are :
R1 : 12 R2 : 12
Allocated Resourse are :
R1 : 6 R2 : 7
Available Resourse are :
R1 : 6 R2 : 5
Process 1 is executing
Your process is in safe state
Available vector :7 , 6 ,
Process 2 is executing
Your process is in safe state
Available vector :10 , 9 ,
Process 3 is executing
Your process is in safe state
Available vector :11 , 11 ,
Process 4 is executing
Your process is in safe state
Available vector :12 , 12 ,

Output Example 2:

```
C:\Users\NAEEM UR RAHMAN\OneDrive\Desktop\Bankers Algorithm.exe
Enter Number of Processes : 5
Enter Number of Resourses : 3
Enter the Total MAX resoures Instances
Enter R1 : 5
Enter R2 : 5
Enter R3 : 5

Enter the allocation table of the Processes !
Process 1
R1 Allocated : 1
R2 Allocated : 0
R3 Allocated : 0

Process 2
R1 Allocated : 2
R2 Allocated : 2
R3 Allocated : 0

Process 3
R1 Allocated : 0
R2 Allocated : 0
R3 Allocated : 2

Process 4
R1 Allocated : 0
R2 Allocated : 0
R3 Allocated : 0

Process 5
R1 Allocated : 1
R2 Allocated : 1
R3 Allocated : 1

Enter the MAX Claim table of the Processes !
Process 1
R1 MAX Claim : 3
R2 MAX Claim : 3
R3 MAX Claim : 2

Process 2
R1 MAX Claim : 3
R2 MAX Claim : 3
R3 MAX Claim : 3

Process 3
R1 MAX Claim : 1
R2 MAX Claim : 1
R3 MAX Claim : 2

Process 4
R1 MAX Claim : 1
R2 MAX Claim : 1
R3 MAX Claim : 1

Process 5
R1 MAX Claim : 2
R2 MAX Claim : 2
R3 MAX Claim : 1
```

```
Allocated Table
_____________________________________________________________________
Process     R1          R2          R3
_____________________________________________________________________
1           1           0           0
2           2           2           0
3           0           0           2
4           0           0           0
5           1           1           1
_____________________________________________________________________
MAX Claim Table
_____________________________________________________________________
Process     R1          R2          R3
_____________________________________________________________________
1           3           3           2
2           3           3           3
3           1           1           2
4           1           1           1
5           2           2           1
_____________________________________________________________________
Total MAX resoures Instances Are :
R1 : 5         R2 : 5         R3 : 5
_____________________________________________________________________
Allocated Resourse are :
R1 : 4         R2 : 3         R3 : 3
_____________________________________________________________________
Available Resourse are :
R1 : 1         R2 : 2         R3 : 2
_____________________________________________________________________
Process 3 is executing
Your process is in safe state
Available vector :1  2  4

Process 2 is executing
Your process is in safe state
Available vector :3  4  4

Process 1 is executing
Your process is in safe state
Available vector :4  4  4

Process 4 is executing
Your process is in safe state
Available vector :4  4  4

Process 5 is executing
Your process is in safe state
Available vector :5  5  5


--------------------------------
Process exited after 226.7 seconds with return value 0
Press any key to continue . . .
```