**Ashesh Kumar**

**501254**

**I.T. – 5**

**Q. Implement Bankers Algorithm**

**Bankers.java**  
import java.util.Scanner;  
public class Bankers{  
 private int need[][],allocate[][],max[][],avail[][],np,nr;  
 private void input(){  
 Scanner sc=new Scanner(System.in);  
 System.out.print("Enter no. of processes and resources : ");  
 np=sc.nextInt();  
 nr=sc.nextInt();  
 need=new int[np][nr];  
 max=new int[np][nr];  
 allocate=new int[np][nr];  
 avail=new int[1][nr];  
   
 System.out.println("Enter allocation matrix --&gt;");  
 for(int i=0;i&lt;np;i++)  
 for(int j=0;j&lt;nr;j++)  
 allocate[i][j]=sc.nextInt();  
   
 System.out.println("Enter max matrix --&gt;");  
 for(int i=0;i&lt;np;i++)  
 for(int j=0;j&lt;nr;j++)  
 max[i][j]=sc.nextInt();  
   
 System.out.println("Enter available matrix --&gt;");  
 for(int j=0;j&lt;nr;j++)  
 avail[0][j]=sc.nextInt();  
   
 sc.close();  
 }

private int[][] calc\_need(){  
 for(int i=0;i&lt;np;i++)  
 for(int j=0;j&lt;nr;j++)  
 need[i][j]=max[i][j]-allocate[i][j];  
   
 return need;  
 }

private boolean check(int i){  
 for(int j=0;j&lt;nr;j++)  
 if(avail[0][j]&lt;need[i][j])  
 return false;  
   
 return true;  
 }

public void isSafe(){  
 input();  
 calc\_need();  
 boolean done[]=new boolean[np];  
 int j=0;  
   
 while(j&lt;np){  
 boolean allocated=false;  
 for(int i=0;i&lt;np;i++)  
 if(!done[i] && check(i)){  
 for(int k=0;k&lt;nr;k++)  
 avail[0][k]=avail[0][k]-need[i][k]+max[i][k];  
 System.out.println("Allocated process : "+i);  
 allocated=done[i]=true;  
 j++;  
 }  
 if(!allocated) break;  
 }  
 if(j==np)  
 System.out.println("\nSafely allocated");  
 else  
 System.out.println("All proceess cant be allocated safely");  
 }  
   
 public static void main(String[] args) {  
 new Bankers().isSafe();  
 }  
}

**C:\Users\Ashesh\Documents\Prog\OS>java Bankers**

**Enter no. of processes and resources : 5 4**

**Enter allocation matrix -->**

**0 0 1 2**

**1 0 0 0**

**1 3 5 4**

**0 6 3 2**

**0 0 1 4**

**Enter max matrix -->**

**0 0 1 2**

**1 7 5 0**

**2 3 5 6**

**0 6 5 2**

**0 6 5 6**

**Enter available matrix -->**

**1 5 2 0**

**Allocated process : 0**

**Allocated process : 2**

**Allocated process : 3**

**Allocated process : 4**

**Allocated process : 1**

**Safely allocated**