**Algorithm 5 :-** *Write a program to demonstrate the use of Safe State CPU Scheduling algorithm.*

import java.util.\*;

public class Bankersalgo {

int max[][];

int need[][];

int available[][];

int allocation[][];

int np, nr;

public void input() {

Random random = new Random();

Scanner input = new Scanner(System.in );

System.out.println("Enter no. of Process & no. of Resource ");

np = input.nextInt();

nr = input.nextInt();

max = new int[np][nr];

need = new int[np][nr];

available = new int[1][nr];

allocation = new int[np][nr];

for (int i = 0; i < np; i++) {

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

allocation[i][j] = (random.nextInt((5 - 0) + 1) + 0);

} }

for (int i = 0; i < np; i++) {

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

max[i][j] = (random.nextInt((9 - 5) + 1) + 5);

} }

for (int i = 0; i < nr; i++) {

available[0][i] = (random.nextInt(10));

} }

public void cal\_need() {

for (int i = 0; i < np; i++) {

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

need[i][j] = max[i][j] - allocation[i][j];

} } }

public boolean check(int p) {

for (int i = 0; i < nr; i++) {

if (available[0][i] < need[p][i]) {

return false;

} }

return true;

}

public void alogrithm() {

cal\_need();

int c = 0;

boolean status[] = new boolean[np];

while (c < np) {

boolean allocated = false;

for (int i = 0; i < np; i++) {

if (!status[i] && check(i)) {

status[i] = true;

allocated = true;

c++;

System.out.println("Allocated process : " + i);

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

available[0][j] = available[0][j] + allocation[i][j];

} } }

if (!allocated) break; //if no allocation

}

if (c == np) //if all processes are allocated

System.out.println("\nSafely allocated");

else

System.out.println("All proceess cant be allocated safely");

}

public static void main(String[] args) {

Bankersalgo obj = new Bankersalgo();

obj.input();

obj.alogrithm();

} }

**OUTPUT :-**

