Name: Devanshu Surana

Roll No.: 12

Panel: C**, Batch:** C1

OS Lab 5 Code:

Title: Bankers algorithm for deadlock avoidance.

```
#include <stdio.h>

int safetyFunction(int max[5][3],int alloc[5][3],int avail[3],int n, int m,int
i,int j,int k){

int f[n], ans[n], ind = 0;
for (k = 0; k < n; k++) {

f[k] = 0; }

int need[n][m];
for (i = 0; i < n; i++) {

for (j = 0; j < m; j++)
need[i][j] = max[i][j] - alloc[i][j];
```

```
}
printf("Max matrix is:\n"); for(int a = 0;a < n;a++){

for(int b = 0;b < m;b++){ printf("%d\t",max[a][b]);

} printf("\n");




}
printf("Alloc matrix is:\n"); for(int a = 0;a < n;a++){

for(int b = 0;b < m;b++){ printf("%d\t",alloc[a][b]);

}

printf("\n"); }

printf("Available is: \n"); for(int a = 0;a < 3;a++){

printf("%d\t",avail[a]); }

printf("\n");
printf("Need matrix is:\n"); for(int a = 0;a < n;a++){
```

```
for(int b = 0;b < m;b++){ printf("%d\t",need[a][b]);

}

printf("\n"); }

int y = 0;
for (k = 0; k < 5; k++) {

for (i = 0; i < n; i++) { if (f[i] == 0) {

int flag = 0;




} }

}
int flag = 1;



for (j = 0; j < m; j++) {
if (need[i][j] > avail[j]){

flag = 1;
```

```
break; }

}

if (flag == 0) { ans[ind++] = i;

for (y = 0; y < m; y++) avail[y] += alloc[i][y];

f[i] = 1; }



for(int i=0;i<n;i++) {
if(f[i]==0)
{

flag=0;
printf("The following system is not safe"); break;



} }

if(flag==1)
{
```

```
printf("Following is the SAFE Sequence\n"); for (i = 0; i < n - 1; i++)

printf(" P%d ->", ans[i]); printf(" P%d", ans[n - 1]);

}
}
int main() {

int n, m,i,j,k;
n = 5;
m = 3;
int alloc[5][3] = { { 0, 1, 0 },

        {  2, 0, 0 },

        {  3, 0, 2 },

        {  2, 1, 1 },

        {  0, 0, 2 } };
        int max[5][3] = { { 7, 5, 3 },
         { 3, 2, 2 },

        { 9, 0, 2 },
```

```
{ 2, 2, 2 },
{ 4, 3, 3 } };

int avail[3] = { 3, 3, 2 };

safetyFunction(max,alloc,avail,n,m,i,j,k);

return 0; }
```

OUTPUT:

Output

```
/tmp/n2JYOkiuiU.o
Max matrix is:
7   5   3
3   2   2
9   0   2
2   2   2
4   3   3
Alloc matrix is:
0   1   0
2   0   0
3   0   2
2   1   1
0   0   2
Available is:
3   3   2
Need matrix is:
7   4   3
1   2   2
6   0   0
0   1   1
4   3   1
Following is the SAFE Sequence
 P1 -> P3 -> P4 -> P0 -> P2
```