**NAME : MARIYAM PATEL**

**SRN:202201178**

**ROLL NO:34**

**ASSIGNMENT-6**

Q.Implement Banker’s Safety algorithm for Deadlock Avoidance.

CODE-

#include<stdio.h>

int main() {

int p, c, count = 0, i, j;

int alc[5][3], max[5][3], need[5][3], safe[5], available[3], done[5] = {0};

printf("Enter the number of processes and resources: ");

scanf("%d %d", &p, &c);

printf("Enter the allocation matrix (%dx%d):\n", p, c);

for (i = 0; i < p; i++)

for (j = 0; j < c; j++)

scanf("%d", &alc[i][j]);

printf("Enter the max matrix (%dx%d):\n", p, c);

for (i = 0; i < p; i++)

for (j = 0; j < c; j++)

scanf("%d", &max[i][j]);

printf("Enter the available resources: ");

for (i = 0; i < c; i++)

scanf("%d", &available[i]);

printf("\nNeed matrix:\n");

for (i = 0; i < p; i++) {

for (j = 0; j < c; j++) {

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

printf("%d\t", need[i][j]);

}

printf("\n");

}

while (count < p) {

int found = 0;

for (i = 0; i < p; i++) {

if (!done[i]) {

for (j = 0; j < c; j++)

if (need[i][j] > available[j])

break;

if (j == c) {

safe[count++] = i;

done[i] = 1;

for (j = 0; j < c; j++)

available[j] += alc[i][j];

found = 1;

}

}

}

if (!found) {

printf("Safe sequence does not exist.\n");

return 0;

}

}

printf("\nAvailable resources after completion:\n");

for (i = 0; i < c; i++)

printf("%d\t", available[i]);

printf("\nSafe sequence:\n");

for (i = 0; i < p; i++)

printf("P%d\t", safe[i]);

return 0;

}

OUTPUT-

Enter the number of process and resources

5 3

enter allocation of resource of all process 5x3 matrix

0 1 0

2 0 0

3 0 2

2 1 1

0 0 2

enter the max resource process required 5x3 matrix

7 5 3

3 2 2

9 0 2

4 2 2

5 3 3

enter the available resource 3 3 2

need resources matrix are

7 4 3

1 2 2

6 0 0

2 1 1

5 3 1

available resource after completion

10 5 7

safe sequence are

p1 p3 p4 p0 p2