

26/7/23

Bankers Algorithm

```
#include <stdio.h>
```

```
#include <stdbool.h>
```

```
void main () {
```

```
    int alloc[10][10], max[10][10], avail[10], work[10];
```

```
    int need[10][10];
```

```
    char finish[10] = {0};
```

```
    int n, m;
```

```
    char safe-sequence[10][3];
```

```
    int count = 0;
```

```
    printf("Enter the no of Processes & resources:");
```

```
    scanf("%d %d", &n, &m);
```

```
    printf("Enter the allocation matrix: \n");
```

```
    for (int i = 0; i < n; i++)
```

```
        for (int j = 0; j < m; j++)
```

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

```
    printf("Enter the max resource matrix: \n");
```

```
    for (int i = 0; i < n; i++)
```

```
        for (int j = 0; j < m; j++)
```

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

```

Printf("Enter the available resource vector:");
for (int i=0; i<n; i++){
    scanf("%d", &avail[i]);
    work[i] = avail[i];
}

```

```

}

```

```

for (int i=0; i<n; i++)

```

```

for (int j=0; j<m; j++)

```

```

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

```

```

bool found = false;

```

```

int index = 0;

```

```

while (count < n){

```

```

    found = false;

```

```

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

```

```

        if (!finish[i]){

```

```

            bool can-execute = true;

```

```

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

```

```

                if (need[i][j] > work[j]){

```

```

                    can-execute = false;

```

```

                    break;

```

```

            }

```

```

        }

```

```

        if (can-execute){

```

```

            for (int j = 0; j<m; j++)

```

```

                work[j] += alloc[i][j];

```

finish [i] = 1;

printf ("Safe - Seq, [index++], "P%d", i + 1);

count++;

found = true;

}

}

}

if (!found)

locate;

}

if (count == n) {

printf ("System is in Safe State. \n Safe sequence: ");

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

printf ("P%d ", Safe-Seq[i]);

if (i < n - 1)

printf (" -> ");

}

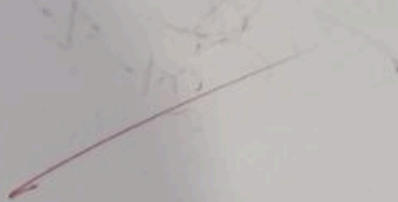
printf ("\n");

} else {

printf ("System is not in a Safe State. \n");

}

}



Output

Enter the no of Processes & resources : 5 3

Enter the allocation matrix

0	1	0
2	0	0
3	0	2
2	1	1
0	0	2

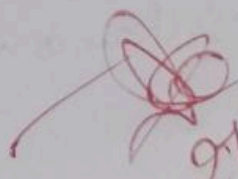
Enter the max resource matrix

7	5	3
3	2	2
9	0	2
2	2	2
4	3	3

Enter the available resource vector : 3 3 2

System is in a safe state.

Safe Sequence : $P_2 \rightarrow P_4 \rightarrow P_5 \rightarrow P_1 \rightarrow P_3$


21-7-22