Synchronization
Abu Shahid
B20CS003

Late Night Pizza

For this part, please refer to the pdf in the root folder of the submission

Sleeping Barber Problem

Find the problem description here.

Execution

`gcc sleeping_barber.c -o sleeper -pthread -w`
`./sleeper <count of chairs> <count of customers>`

- default value of number of chairs and customers is 8 and 50 respectively.
- Implementation synchronizes the availability of chairs.
- Monitor implemented using mutex and semaphores.
- PFA the log of program execution in the root folder (8 chairs and 50 customers)

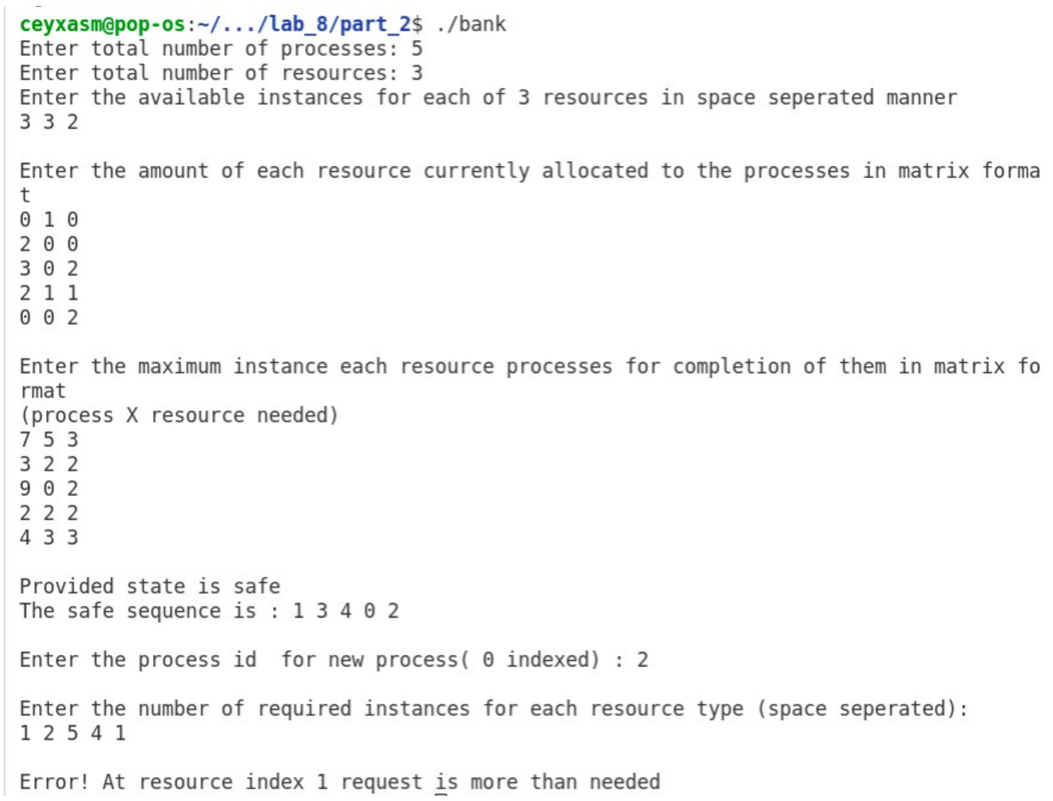
Banker's Algorithm

Execution

`g++ banker_algo.cpp -o banker && ./banker`

- Determines whether the input state is safe or not.
- Decides whether the process request can be granted on not, by computing the safe/unsafe state.

(below stated example can be refered from here, validating the result of our implementation)



```
ceyxasm@pop-os:~/.../lab_8/part_2$ ./bank
Enter total number of processes: 5
Enter total number of resources: 3
Enter the available instances for each of 3 resources in space seperated manner
3 3 2

Enter the amount of each resource currently allocated to the processes in matrix forma
t
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2

Enter the maximum instance each resource processes for completion of them in matrix fo
rmat
(process X resource needed)
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3

Provided state is safe
The safe sequence is : 1 3 4 0 2

Enter the process id  for new process( 0 indexed) : 2

Enter the number of required instances for each resource type (space seperated):
1 2 5 4 1

Error! At resource index 1 request is more than needed
```