Overview of assignment functions

* Main
  + Get user in
  + Create and run threads
* processSimulator
  + thread simulates a process
  + get random resources request vector
  + call banker alg
  + only get resources if allocation is safe
    - else busy loop calling bankers algo
  + release resources if it gets its needs met
* BankersAlgorithim
  + Possible implementation of bankers algorithm for deadlock avoidance
  + Try to implement it like the class slides
  + Calls isSafe . and also release resources within (?) follow slides (2 slides)
* Fault\_simulator()
  + Thread running in background removing resources with probability described in the spec
* Deadlock\_checker
  + Simple checkin if deadlock occoured due to resource fault

Main inputs – scanf?

Malloc for arrays. Loop through array and scanf.

For each matrix: int\*\* data;  
data = (int\*\*) malloc(sizeof(int\*) num\_proc);  
 for(i=0; i<num\_proc; i++){  
 data[i]=(int\*)malloc(sizeof(int) x num\_res)  
 }  
 data[i][j

* Num process
* Num resources
* Availj: 1-d array of available resources
* Maxij: 2d max resources each process can claim

Need = max-hold

Request is randomly generated

E.g.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MAX | A | B | C | D |
| P1 | 3 | 3 | 2 | 2 |
| P2 | 1 | 2 | 3 | 4 |
| P3 | 1 | 3 | 5 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HOLD | A | B | C | D |
| P1 | 1 | 2 | 2 | 1 |
| P2 | 1 | 0 | 3 | 3 |
| P3 | 1 | 2 | 1 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| NEED | A | B | C | D |
| P1 | 2 | 1 | 0 | 1 |
| P2 | 0 | 2 | 0 | 1 |
| P3 | 0 | 1 | 4 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AVAIL | A | B | C | D |
| 3 | 1 | 1 | 2 |