Midterm#3: Dec. 21st

Goal: Write a “Board Keeping” client-server application.

- Server maintains a <Cell-Value> data structure (i.e. an array)
    - Cell **name** and **value** are both **strings**
- Clients connect, query and update cells, using the following set of messages (**OK** and **NOK** are #define’d constants):
    - **Add <Cell> <Val>**, returns (OK, NOK)
    - **Delete <Cell>**, returns (OK, NOK)
    - **Lock** <Cell>, returns (OK, NOK)
    - **Unlock** <Cell>, returns (OK, NOK)
    - **Get** <Cell>, returns <Current Value of Cell>
    - **Set** <Cell> <newValue>, returns (<oldValue>, NOK)

Server Flow:

- Creates a shared memory segment for the of data structure
  - A maximum size N can be defined as a parameter
- Creates IPC
- Initializes data structures and IPC (your choice)
- Creates a stream socket, then:
  - bind(the port number could be #define’d or a parameter)
  - listen()
- Loops on accept()
  - Forks a new child to handle each connection
- close()/exit()

Client Flow:

- Loop:
  - Read a request from stdin
  - Performs the operation (lock/get/set/unlock/..)
  - Prints result on stdout
- close()/exit()

Example

```
$./server
IPC .. Done
Ready.

$./client
> add Key1 abcd
 .. - OK
> set Key1 5678
.. abcd - OK
> get Key1
.. 5678 - OK
> get Key2
.. - Error
> exit
$
```