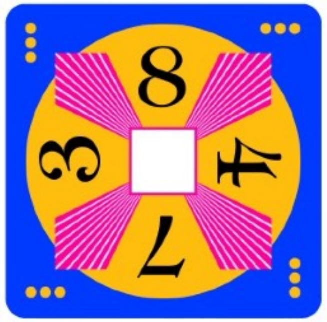
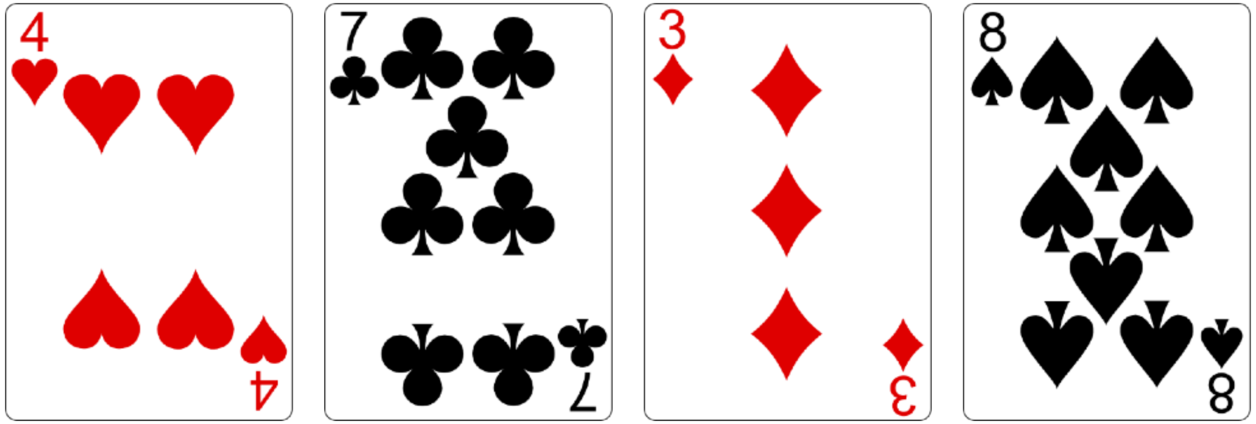
Assignment 1

COMP3358 Distributed and Parallel Computing

Overview



- A **24-game** game system played with poker playing card
- Consists of:
  - A **server** that handle user login, user matching, and gaming support
  - A **client** that allow users to login and play games



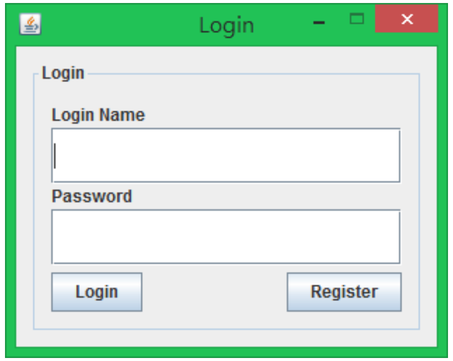
Solution: 4*(7-3)+8 = 24

Tentative assignments

- Assignment 1: **RMI** – handles user registration and login
- Assignment 2: **JDBC** – handles data storage and retrieval
- Assignment 3: **JMS** – handle client-server communication

UI flow

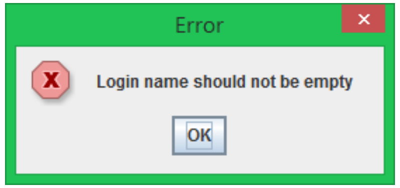
Note that you only need to follow the flow. You can design your own UI, this is just an example.



Login

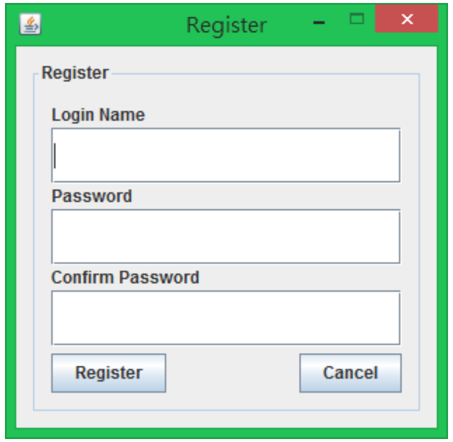
Register

Error

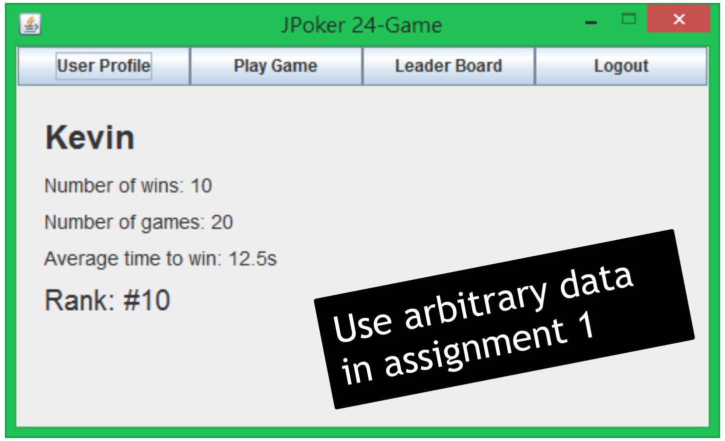


Login name should not be empty

Error



Register



Kevin

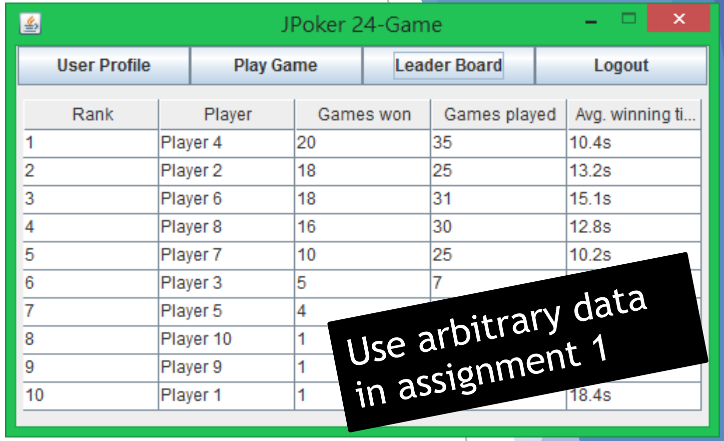
Number of wins: 10

Number of games: 20

Average time to win: 12.5s

Rank: #10

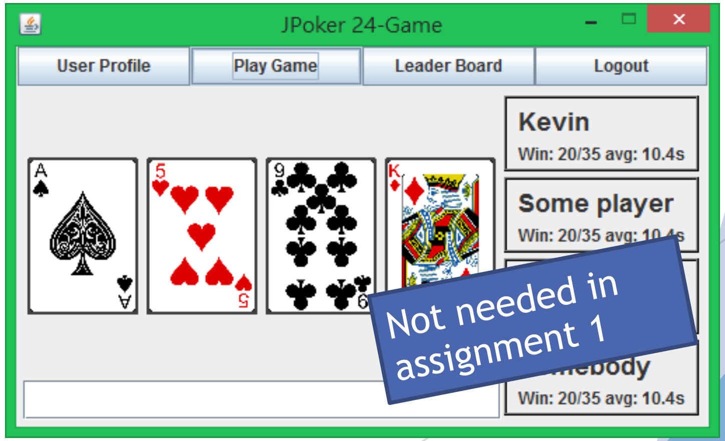
Use arbitrary data in assignment 1



Rank	Player	Games won	Games played	Avg. winning ti...
1	Player 4	20	35	10.4s
2	Player 2	18	25	13.2s
3	Player 6	18	31	15.1s
4	Player 8	16	30	12.8s
5	Player 7	10	25	10.2s
6	Player 3	5	7	
7	Player 5	4		
8	Player 10	1		
9	Player 9	1		
10	Player 1	1		18.4s

Use arbitrary data in assignment 1

Section changes



Kevin
Win: 20/35 avg: 10.4s

Some player
Win: 20/35 avg: 10.4s

...mebody
Win: 20/35 avg: 10.4s

Not needed in assignment 1

Assignment 1 – user data

- In assignment 1, the server must maintain two files
  - `UserInfo.txt` – storing registered user information
    - Persistence across server execution
    - Used to authenticate user
    - Updated in successful user registration
    - Checked to avoid duplicating user name during registration
  - `OnlineUser.txt` – keeping a list of online users
    - Cleared when server start
    - Updated when user login/logout
    - Checked to avoid multiple login
- Note that this will be replaced by the use of database in assignment 2
  - You may want to plan ahead

Assignment 1 - RMI

- The server must use RMI to support the following functions:
  - **Login**
    - validate user from `UserInfo.txt`
    - avoid repeated login using `OnlineUser.txt`
    - update `OnlineUser.txt`
  - **Register**
    - avoid duplicating user name with `UserInfo.txt`
    - login user (update `OnlineUser.txt`)
  - **Logout**
    - update `OnlineUser.txt`

New policy file

```
grant {
    permission java.net.SocketPermission "*:1024-65535", "connect,accept";
    permission java.net.SocketPermission "*:80", "connect";
    permission java.io.FilePermission "UserInfo.txt", "read,write";
    permission java.io.FilePermission "OnlineUser.txt", "read,write";
};
```

Need these because of security manager

Submission

- **Please do the programming assignments on your virtual machine. Submit the code you have write** (all necessary *.java files) **and a document to Moodle (better in pdf format). The doc should contain screen shots and some short description on how you ran the GUI, including login, register, and logout.**