Úloha č. 1 – RMI

Herní server pro hru 'sirky'

Zadání:
Implementujte server, který nabídne klientovi možnost hrát hru 'sirky'. Pravidla hry jsou následující: 2 hráči si určí počet sirek, se kterými budou hrát, pak si zvolí kolik sirek budou moci odebrat při jednom tahu a kolik sirek může zbýt. V průběhu hry se střídají v odebírání sirek. Každý musí odebrat 1 až předem dohodnutý počet sirek. Na koho zbude určený minimální počet sirek, ten prohrál.
V našem případě budeme automaticky předpokládat, že se může brát maximálně 5 sirek (včetně) a že prohrává ten, kdo sebere poslední sirku.

Server by měl poskytovat možnost výběru z více současných her – funkce `list`, `create` a `join`.
Hra je identifikována názvem partie (= jméno hry, id hry). Hra získává informace o tazích od klienta a ukládá je do svého logu a mění svůj vnitřní stav (počet sirek, hráč na tahu, status hry). Hra se může nacházet v následujících stavech – ready, played, ended.

Klient má možnost přihlásit se k serveru, vybrat/založit hru a poté odesílat posloupnost jednotlivých tahů (za oba hráče).

Vnitřní struktury
Hra musí ukládat svůj průběh do logovacího souboru. Každá hra má svůj logovací soubor.

Např.:
```
0;2;30
tah#;hráčID;počet_sirek
```

Úvodní řádek má formát: `0;pocet_hracu;pocet_sirek`

Dávkové zpracování
```
./server [IP_adresa] [cislo_portu]
```

, kde *IP_adresa* a *cislo_portu* specifikuje, kde má server naslouchat

```
./client IP_adresa cislo_portu konf_soubor
```

, kde *IP_adresa* a *cislo_portu* specifikují, kde je potřeba hledat server a v souboru je posloupnost příkazů pro klienta. Příkazy mohou mít jeden z následujících tvarů:

```
list
join game_id
create game_id num_of_matches
turn player_id num_of_matches
```

Návrh implementace:
(pouze nezávazný návrh)
class Server
    Game[] list()
    void create(game_id, num_of_matches)
    Game join(game_id)
class Game
    Status turn(player_id, num_of_matches)
    Status getStatus()

Ukázkový výstup:

Obsah souboru *konf_soubor*

```
list
create pokus1 25
list
create pokus2 10
list
join pokus2
turn 1 4
turn 2 1
turn 1 4
turn 2 1
```

Výstup z klientské aplikace

```
>> list
<< Games:
>> create pokus1 25
<< Game 'Pokus1' (25) created
>> list
<< Games: Pokus1
>> create pokus2 10
<< Game 'Pokus2' (10) created
>> list
<< Games: Pokus1;Pokus2
>> join Pokus2
<< Game 'Pokus2' joined
<< Game 'Pokus2' status: player 1 turn, 10 matches left
>> turn 1 4
<< Game 'Pokus2' status: player 2 turn, 6 matches left
>> turn 2 1
<< Game 'Pokus2' status: player 1 turn, 5 matches left
>> turn 1 4
<< Game 'Pokus2' status: player 2 turn, 1 matches left
>> turn 2 1
<< Game 'Pokus2' status: Player 1 wins
```