Pontifícia Universidade Católica do Rio Grande do Sul – PUCRS
Faculdade de Informática
Programação Distribuída – Turma 128 – Prof. Roland Teodorowitsch – 2018/1

TRABALHO DA ÁREA 1: UNO DISTRIBUÍDO EM JAVA RMI

Objetivo

Implementar uma aplicação distribuída em Java RMI (*Remote Method Invocation*) que permita que usuários remotos disputem o jogo de cartas Uno (2003).

Regras do Jogo

Uno é o nome da versão comercial (produzida pela empresa Mattel) de um jogo de cartas em que o objetivo dos jogadores é se livrar de suas cartas. Há variações que podem ser jogadas com um baralho comum (por exemplo, Mau-mau) ou produzidas por outros fabricantes (por exemplo, Can Can da Grow Jogos e Brinquedos Ltda.), mas para a definição deste trabalho serão usadas as regras do Uno.

Uno é um jogo formado por 108 cartas:

- 19 cartas azuis (uma com valor 0 e duas de cada um dos valores de 1 a 9);
- 19 cartas verdes (uma com valor 0 e duas de cada um dos valores de 1 a 9);
- 19 cartas vermelhas (uma com valor 0 e duas de cada um dos valores de 1 a 9);
- 19 cartas amarelas (uma com valor 0 e duas de cada um dos valores de 1 a 9);
- 8 cartas +2 (comprar 2 cartas) – 2 de cada cor (azul, verde, vermelho e amarelo);
- 8 cartas Inverter – 2 de cada cor (azul, verde, vermelho e amarelo);
- 8 cartas Pular – 2 de cada cor (azul, verde, vermelho e amarelo);
- 4 Curingas;
- 4 Curingas +4 (comprar 4 cartas).

A Figura 1 mostra 9 exemplos de cartas de Uno. O verso das cartas corresponde à ilustração que está na coluna mais à esquerda e na linha inferior da Figura 1.

Figura 1 – Algumas cartas de Uno (BARALHO-DO-JOGO-UNO.JPG, [s.d.])

O jogo pode ser disputado de 2 a 10 jogadores, acumulando pontos de várias partidas. No entanto para a finalidade deste trabalho será considerada uma única partida entre 2 jogadores.

Desta forma, as regras apresentadas a seguir foram em grande parte transcritas do Manual de Instruções do jogo (UNO, 2003), com as adaptações e simplificações para 2 jogadores jogando de forma distribuída já aplicadas.

Inicialmente as 108 cartas são embaralhadas, sendo que cada jogador receberá 7 destas cartas. A próxima carta do baralho é colocada com a face voltada para cima e corresponderá à pilha de descarte. As demais cartas corresponderão à pilha de compra e devem ficar com a face voltada para baixo. Iniciará a partida o primeiro jogador a ter se registrado no servidor para jogar Uno. A partir daí os jogadores farão jogadas alternadas tentando se livrar das cartas que têm em suas mãos.

Eventualmente a sequência de jogo poderá ser alterada por uma das Cartas de Ação. Para dois jogadores a única alteração possível na sequência de jogo corresponde a pular o próximo jogador, o que, para dois jogadores, corresponde a jogar duas vezes.

A maioria das cartas possui duas características básicas: uma cor e um número. Na sua vez de jogar o jogador deve descartar uma carta que tenha ou a mesma cor ou o mesmo valor que a carta que está no topo da pilha de descarte. Por exemplo, se a carta que está no topo da pilha de descarte é a carta com o número 7 em azul. Pode-se jogar qualquer carta com 7 (de qualquer cor), qualquer carta azul (com qualquer número), ou uma das cartas de curinga (que permitem inclusive que o jogador selecione a cor ativa para a próxima jogada).

Desta forma, na sua vez de jogar o jogador deve descartar uma das cartas de sua mão que corresponda ao número, ou à cor ou que seja um curinga. Caso o jogador não possua nenhuma destas cartas, ele deverá comprar uma carta da pilha de compra. Se esta carta servir, o jogador poderá descartá-la imediatamente. Se não servir, o jogador deverá passar a vez.

Mesmo que o jogador tenha uma carta que possa ser jogada, pode ser uma estratégia de jogo não descartá-la. Por exemplo, se tiver em sua mão uma carta e não quiser jogá-la, o jogador deverá comprar uma carta da pilha de compra. Se tiver comprado uma carta da pilha de compra, o jogador poderá também optar em descartar esta carta ou não. Mas não é permitido comprar uma carta da pilha de compra, para na sequência descartar uma carta que o jogador já tinha na sua mão.

Algumas cartas não contêm números. Estas são chamadas Cartas de Ação. Elas são as seguintes:

- "+2": quando esta carta é jogada, o próximo jogador deverá comprar 2 cartas e perderá a vez (ou seja, em um jogo com dois jogadores, o próximo jogador compra 2 cartas e o mesmo jogador continua jogando). Esta carta só pode ser jogada sobre uma de mesma cor e sobre outras cartas "+2". Se ela for aberta no começo do jogo, as mesmas regras se aplicam.
- "pular": esta carta pular o próximo jogador, o que em um jogo entre dois jogadores corresponde a fazer com que o jogador que a jogou continue jogando. Esta carta somente pode ser jogada sobre uma carta da mesma cor ou sobre uma outra carta "pular". Se uma carta "pular" for aberta no início do jogo, o jogador que estava previsto para iniciar o jogo perde a vez e o outro jogador iniciará o jogo.
- "inverter": em um jogo entre dois jogadores, esta carta tem exatamente o mesmo efeito da carta "pular".
- "curinga": o jogador que jogar esta carta pode jogá-la como se ela tivesse qualquer cor ou número. Se esta carta for aberta no início do jogo, o primeiro jogador pode jogar qualquer uma de suas cartas (independentemente de cor e número). O jogador que jogar esta carta deverá escolher a cor para a próxima jogada.
- "curinga +4": esta é a melhor carta do jogo. Além de poder ser jogada sobre qualquer cor ou número e do jogador escolher a cor para a próxima jogada, o próximo jogador deverá comprar 4 cartas da pilha de compra. Esta carta somente poderá ser jogada se o jogador não tiver nenhuma carta da mesma cor ativa no monte de descarte. Se esta carta for aberta no começo do jogo, ela é descartada (vai para a pilha de descarte) e outra carta é aberta.

Caso um jogador consiga descartar todas as suas cartas, ele vence a partida. E recebe o número de

pontos correspondente à soma dos valores das cartas que estavam na mão do outro jogador.

O jogo também pode terminar quando a pilha de compra tiver se esgotado e o próximo jogador não tiver mais nenhuma opção de jogo possível. Neste caso somam-se os valores das cartas de cada jogador. Os pontos de um jogador correspondem à soma das cartas do outro jogador. Vence quem tiver mais pontos. Ou se a pontuação for a mesma, é decretado empate.

Os pontos das cartas são definidos da seguinte forma:

- cartas com número valem o respectivo valor do número;
- cartas "+2", "pular" e "inverter" valem 20 pontos;
- cartas de curinga valem 50 pontos.

Na versão distribuída não é necessário avisar que está com uma única carta gritando "Uno!". Também não há penalidades.

Funcionamento da Aplicação

O servidor deverá funcionar de modo que:

- sejam suportadas 500 partidas simultâneas de Uno entre 2 jogadores devidamente registrados (ou identificados) no servidor;
- quando um jogador se registra, ele deverá esperar que outro jogador também se registre (quando o próximo jogador se registrar, será formada uma dupla que disputará a próxima partida);
- o primeiro jogador a se registrar inicia jogando;
- responda a invocações remotas de métodos realizadas pelos clientes (por exemplo, conforme a descrição de operações descrita a seguir);
- haja limites de tempo para determinados eventos: 2 minutos (120 segundos) pelo registro do segundo jogador; 60 segundos pelas jogadas de cada jogador; e 60 segundos para "destruir" a partida depois de definido o vencedor.

O cliente será responsável: pela interface com o usuário (que poderá ser tanto em modo texto quanto em modo gráfico); e por executar as invocações remotas de métodos disponíveis no servidor, de modo que os usuários possam jogar partidas consistentes.

Operações

As seguintes operações remotas deverão ser implementadas pelo servidor:

1) registraJogador
   Recebe: *string* com o nome do usuário/jogador.
   Retorna: id (valor inteiro) do usuário (que corresponde a um número de identificação único para este usuário durante uma partida), -1 se este usuário já está cadastrado ou -2 se o número máximo de jogadores tiver sido atingido.

2) encerraPartida
   Recebe: id do usuário (obtido através da chamada registraJogador).
   Retorna: -1 (erro), 0 (ok).

3) temPartida
   Recebe: id do usuário (obtido através da chamada registraJogador).

Retorna: -2 (tempo de espera esgotado), -1 (erro), 0 (ainda não há partida), 1 (sim, há partida e o jogador inicia jogando) ou 2 (sim, há partida e o jogador é o segundo a jogar).

4) obtemOponente
   Recebe: id do usuário (obtido através da chamada registraJogador).
   Retorna: *string* vazio para erro ou *string* com o nome do oponente.

5) ehMinhaVez
   Recebe: id do usuário (obtido através da chamada registraJogador).
   Retorna: -2 (erro: ainda não há 2 jogadores registrados na partida), -1 (erro), 0 (não), 1 (sim), 2 (é o vencedor), 3 (é o perdedor), 4 (houve empate), 5 (vencedor por WO), 6 (perdedor por WO).

6) obtemNumCartasBaralho
   Recebe: id do usuário (obtido através da chamada registraJogador).
   Retorna: -2 (erro: ainda não há 2 jogadores registrados na partida), -1 (erro) ou um valor inteiro com o número de cartas do baralho de compra.

7) obtemNumCartas
   Recebe: id do usuário (obtido através da chamada registraJogador).
   Retorna: -2 (erro: ainda não há 2 jogadores registrados na partida), -1 (erro) ou um valor inteiro com o número de cartas do próprio jogador.

8) obtemNumCartasOponente
   Recebe: id do usuário (obtido através da chamada registraJogador).
   Retorna: -2 (erro: ainda não há 2 jogadores registrados na partida), -1 (erro) ou um valor inteiro com o número de cartas do oponente.

9) mostraMao
   Recebe: id do usuário (obtido através da chamada registraJogador).
   Retorna: *string* vazio em caso de erro ou *string* representando o conjunto de cartas que um jogador tem na sua mão. Uma forma para identificar e representar todas as cartas e principalmente as cartas que fazem parte da mão do jogador corresponde a atribuir a cada uma das 108 cartas do jogo um valor numérico de 0 a 107 e também um rótulo que permita identificar a carta e suas características (cor e valor, por exemplo) de uma forma mais amigável. O Quadro 1 mostra um conjunto de equivalências possível. Neste caso, a operação mostraMao, poderia, por exemplo, retornar a seguinte cadeia de caracteres “3/Az|1/Vd|0/Az|+2/Vm|In/Am|Cg/*|3/Am”, representando as 7 cartas iniciais para determinado jogador (3 azul, 1 verde, 0 azul, “+2” vermelho, “inverter” amarelo, curinga, 3 amarelo), separadas pelo caractere “|”.

Quadro 1 – Representação das Cartas

valor	carta	valor	carta	valor	carta	valor	carta	valor	carta
0	0/Az	25	0/Am	50	0/Vd	75	0/Vm	100	Cg/*
1	1/Az	26	1/Am	51	1/Vd	76	1/Vm	101	Cg/*
2	1/Az	27	1/Am	52	1/Vd	77	1/Vm	102	Cg/*
3	2/Az	28	2/Am	53	2/Vd	78	2/Vm	103	Cg/*
4	2/Az	29	2/Am	54	2/Vd	79	2/Vm	104	C4/*
5	3/Az	30	3/Am	55	3/Vd	80	3/Vm	105	C4/*
6	3/Az	31	3/Am	56	3/Vd	81	3/Vm	106	C4/*
7	4/Az	32	4/Am	57	4/Vd	82	4/Vm	107	C4/*
8	4/Az	33	4/Am	58	4/Vd	83	4/Vm		
9	5/Az	34	5/Am	59	5/Vd	84	5/Vm		
10	5/Az	35	5/Am	60	5/Vd	85	5/Vm		
11	6/Az	36	6/Am	61	6/Vd	86	6/Vm		
12	6/Az	37	6/Am	62	6/Vd	87	6/Vm		
13	7/Az	38	7/Am	63	7/Vd	88	7/Vm		
14	7/Az	39	7/Am	64	7/Vd	89	7/Vm		
15	8/Az	40	8/Am	65	8/Vd	90	8/Vm		
16	8/Az	41	8/Am	66	8/Vd	91	8/Vm		
17	9/Az	42	9/Am	67	9/Vd	92	9/Vm		
18	9/Az	43	9/Am	68	9/Vd	93	9/Vm		
19	Pu/Az	44	Pu/Am	69	Pu/Vd	94	Pu/Vm		
20	Pu/Az	45	Pu/Am	70	Pu/Vd	95	Pu/Vm		
21	In/Az	46	In/Am	71	In/Vd	96	In/Vm		
22	In/Az	47	In/Am	72	In/Vd	97	In/Vm		
23	+2/Az	48	+2/Am	73	+2/Vd	98	+2/Vm		
24	+2/Az	49	+2/Am	74	+2/Vd	99	+2/Vm		

Observação: uma pilha de cartas ou um baralho pode ser representado internamente de várias formas. O exemplo de código a seguir mostra o uso de uma estrutura de dados baseada em um vetor. Inicialmente, para viabilizar a realização de testes posteriores, será preciso criar **para cada partida** um gerador de números aleatórios com semente definida a partir da soma dos identificadores dos dois jogadores envolvidos na partida (`id1` e `id2`):

```
// Inicializacao do gerador de numeros aleatorios
Random gerador = new Random(id1+id2);
```

Os valores obtidos a partir deste gerador serão usados exclusivamente no embaralhamento das cartas. Para criar o baralho ordenado usa-se então:

```
// Criacao do baralho com as 108 cartas
final int totalCartas = 108;
int[] baralho = new int[totalCartas];
for (int i=0;i<totalCartas;++i)
    baralho[i] = i;
```

Finalmente pode-se embaralhar as 108 cartas em 2 etapas:

```
// Embaralhamento
for (int c=0;c<totalCartas;++c) {
    int outra = gerador.nextInt(totalCartas);
    int aux = baralho[c];
    baralho[c] = baralho[outra];
    baralho[outra] = aux;
}
for (int c=0;c<totalCartas*totalCartas;c++) {
    int c1 = gerador.nextInt(totalCartas);
    int c2 = gerador.nextInt(totalCartas);
    int aux = baralho[c1];
    baralho[c1] = baralho[c2];
    baralho[c2] = aux;
}
int numCartas = totalCartas;
```

Para comprar uma carta deste baralho, deve-se remover sempre a carta da última posição ocupada do baralho:

```
// O baralho sera o monte de compras
int carta = baralho[--numCartas];
```

A distribuição de cartas deverá seguir também uma ordem predefinida. Começar comprando uma carta e atribuindo ela ao primeiro jogador, depois ao segundo e assim alternadamente até que ambos tenham recebido as 7 cartas iniciais. Cada jogador deve ter também a sua pilha que deverá

ser gerenciada como uma lista sem ordenação: cartas são colocadas no final da lista e podem ser removidas de qualquer posição.

10) obtemCartaMesa
Recebe: id do usuário (obtido através da chamada registraJogador).
Retorna: *string* vazio em caso de erro ou *string* representando a carta que está no topo da pilha de descarte (usando o mesmo padrão definido pela operação mostraMao).

11) obtemCorAtiva
Recebe: id do usuário (obtido através da chamada registraJogador).
Retorna: valor inteiro correspondendo à cor que está ativa no topo do baralho de descarte (0 = azul; 1 = amarelo; 2 = verde; 3 = vermelho) – esta informação é necessária nos casos onde um jogador descartou um curinga e escolheu determinada cor.

12) compraCarta
Recebe: id do usuário (obtido através da chamada registraJogador).
Retorna: código de sucesso (0) ou código de erro (-1).

13) jogaCarta
Recebe: id do usuário (obtido através da chamada registraJogador), índice da carta da mão que deve ser jogada (de 0 até o número máximo de cartas do jogador menos 1) e cor da carta, no caso da carta jogada ser um curinga (0 = azul; 1 = amarelo; 2 = verde; 3 = vermelho).
Retorna: 1 (tudo certo), 0 (jogada inválida: por exemplo, a carta não corresponde à cor que está na mesa), -1 (jogador não encontrado), -2 (partida não iniciada: ainda não há dois jogadores registrados na partida), -3 (parâmetros inválidos), -4 (não é a vez do jogador).

14) obtemPontos
Recebe: id do usuário (obtido através da chamada registraJogador).
Retorna: valor inteiro com o número de pontos conquistado no jogo, -1 (jogador não encontrado), -2 (partida não iniciada: ainda não há dois jogadores registrados na partida), -3 (a partida ainda não foi concluída).

15) obtemPontosOponente
Recebe: id do usuário (obtido através da chamada registraJogador).
Retorna: valor inteiro com o número de pontos conquistado no jogo pelo adversário, -1 (jogador não encontrado), -2 (partida não iniciada: ainda não há dois jogadores registrados na partida), -3 (a partida ainda não foi concluída).

Avaliação

Trabalhos com trechos copiados integralmente ou parcialmente serão avaliados com a nota mínima (ZERO). Os demais trabalhos serão avaliados numa escala de 0 (ZERO) até 10 (DEZ), levando em consideração as características descritas neste documento. Serão utilizados os seguintes pesos nesta avaliação:

- 40%: a aplicação executou corretamente sem erros, apresentando o comportamento esperado, conforme as regras do jogo.
- 20%: todas as particularidades (tais como número de partidas e usuários, etc.) definidas neste documento foram implementadas.
- 20%: o aluno usou padrões de programação adequados (programação estruturada, nomes de variáveis significativos, comentários, etc.).

- 10%: usou bloqueios para proteger variáveis compartilhadas contra inconsistências referentes a acesso concorrente;
- 10%: foi implementado um mecanismo de temporização que funciona corretamente?

Entrega

O trabalho deve ser desenvolvido individualmente.

A data de entrega do trabalho é **10 de maio de 2018**.

Cada aluno deverá entregar todos os arquivos com extensão ".java" necessários para compilar e executar o projeto. E também deverá apresentar a execução de sua aplicação para o professor.

Em caso de cópia de trabalhos serão avaliados com a nota mínima (zero).

REFERÊNCIAS

BARALHO-DO-JOGO-UNO.JPG. Altura: 558 *pixels*. Largura: 900 *pixels*. 300 dpi. 24 bits RGB. 159 KiB. Formato JPEG bitmap. Compactado. [s.d.]. Disponível em: <http://www.metropolybar.com.br/wp-content/uploads/2017/09/baralho-do-jogo-uno.jpg>. Acesso em: 22 mar. 2018.

UNO Jogo de Cartas. Cajamar/SP: Mattel, 2003. Manual de Intruções.