

Universidade Federal de Mato Grosso do Sul

Redes de Computadores

Jogo Interativo Online

Alunos: Rafael Torres Nantes
Raphaella Brandão Jacques
Professor: Hana Rubinsztein

30 de Outubro de
2023

1 Jogo: Blackjack Game

O jogo desenvolvido tem como inspiração o famoso jogo de baralho **Black-Jack** (ou 21). O objetivo dos jogadores é obter o **maior total** de pontos **sem ultrapassar 21**.

O jogo é dividido em rodadas. Na primeira rodada:

1. O sistema distribui **2 cartas** para cada jogador e calcula suas respectivas pontuações;
2. O **Jogador 1** escolhe se gostaria de comprar mais **uma** carta do baralho.
 - Se **sim**, o sistema entrega mais um carta ao jogador e calcula sua nova pontuação.
 - Se **não**, passa a sua vez para o **Jogador 2**.
3. O **Jogador 2** escolhe se gostaria de comprar mais **uma** carta do baralho.
 - Se **sim**, o sistema entrega mais um carta ao jogador e calcula sua nova pontuação.
 - Se **não**, a rodada acaba.

As rodadas subsequentes são constituídas dos passos **2** e **3** acima.

O jogo se **encerra** caso:

- O dois jogadores **recusarem** comprar mais cartas na rodada; ou,
- Algum jogador **ultrapassar** 21 pontos.

O total de pontos é a **soma** dos valores das cartas na mão de cada jogador. A pontuação das cartas do baralho funciona da seguinte forma:

1. Se for um **dígito**, o valor é o proprio número da carta: [**2, 3, 4 ... 10**];
2. As figuras '**J**', '**Q**' e '**K**' valem **10**;
3. A figura '**A**' tem valor **11**.

Obs: Importante ressaltar que os jogadores **não** conseguem visualizar os pontos do oponente deixando a disputa mais desafiadora.



2 Protocolos do Jogo

Os protocolos do jogo são o conjunto de regras que definem como as informações são trocadas entre os jogadores e o servidor. Eles são essenciais para garantir que o jogo funcione corretamente.

Os protocolos do jogo estão listados a seguir:

1. **ACTION : SIM ou NAO:** As mensagens trocadas entre os jogadores. Tem como função *informar* se o jogador deseja (SIM) ou não(NAO) receber uma carta do baralho.
2. **ACTION : END_GAME:** As mensagens trocadas entre os jogadores. Tem como função *encerrar* o jogo do cliente-clinte.
3. **ACTION : GAME_OVER:** As mensagens trocadas entre cliente-servidor. Tem como função *encerrar* o jogo do cliente-servidor.

```

198 #-----Def do jogo-----
199
200 # Função para encerrar o jogo pelo cliente-oponente
201 def end_game():
202     data = {'action': 'end_game'}
203     opponent_socket.send(json.dumps(data).encode())
204
205 # Função para receber carta
206 def sim():
207     data = {'action': 'sim'}
208     opponent_socket.send(json.dumps(data).encode())
209
210 # Função para não receber carta
211 def nao():
212     data = {'action': 'nao'}
213     opponent_socket.send(json.dumps(data).encode())
214
215 # Função para encerrar o jogo pelo cliente-servidor
216 def game_over():
217     data = {'action': 'game_over', 'username': 'oponente'}
218     conn.close()
219     client_socket.send(json.dumps(data).encode())
220
221 # Imprime um cabeçalho para organizar
222 def organizar_linhas(seg=2):
223     print(40 * '+')
224     time.sleep(seg)
225

```

Figura 1: Protocolos do Jogo em Código



3 Implementações do Trabalho

3.1 Servidor de Autenticação e Informação (SAI)

Register and Login Account: O código consegue cadastrar e acessar os usuários por meio de controle das informações em uma **memória** de dicionário no python. Caso o cliente tenha sido cadastrado com um mesmo *username*, ele pede para refazer a operação.

Mesmo o usuário finalizando (Sair) a interface, as informações são registradas para que o cliente possa logar novamente com o *username* e *password* definidas no registro. Além de possuir um arquivo: **data.json** que armazena os *username* e seus respectivos *name* e *password*.

ATIVO e INATIVO: Com o auxílio da interface, e *LIST_USER_ONLINE* é possível visualizar os usuários e os *status do jogador*. Sendo:

1. **ATIVO:** Usuário Online e jogando.
2. **INATIVO:** Usuário Online e aguardando pedido para jogar.

```
1. Listar usuários online
2. Listar usuários jogando
3. Iniciar um jogo
4. Sair
Escolha a opção (1/2/3/4): 1
rafinha (INATIVO) - IP: 192.168.1.15, Porta: 50980
saritcha (INATIVO) - IP: 192.168.1.16, Porta: 57544
```

Figura 2: Status dos Jogadores

LIST_USER_ONLINE e LIST_USER_PLAYING: Com o auxílio da interface, torna-se possível solicitar as duas listagens para o servidor.

```
1. Listar usuários online
2. Listar usuários jogando
3. Iniciar um jogo
4. Sair
Escolha a opção (1/2/3/4):
```

Figura 3: Listagens de Users Online e Users Playing

GAME_INI: Um usuário A pode solicitar iniciar um jogo com qualquer outro usuário online B e receber como retorno as seguintes mensagens *GAME_NEG* para recusado ou *GAME_ACK* para aceite.

E além disso, o usuário A pode receber solicitações durante o próprio jogo, podendo escolher (*GAME_NEG* ou *GAME_ACK*) se deseja sair do jogo atual e entrar em um novo jogo com o usuário C.



3.2 Log do Blackjack

O servidor deve armazenar, em um dado instante, quais usuários estão on-line e quais estão jogando. Além disso, deve escrever na saída padrão e em um arquivo de log denominado **game.log** os eventos:

```
! game_log U x
! game_log
38 2023-10-30 02:49:36,606 - rafa nao responde ('morreu')
39 2023-10-30 02:50:30,850 - rafa realizou cadastro
40 2023-10-30 02:50:30,851 - rafa ficou inativo
41 2023-10-30 03:00:33,167 - rafa realizou cadastro
42 2023-10-30 03:00:33,167 - rafa ficou inativo
43 2023-10-30 03:01:37,004 - rafa realizou cadastro
44 2023-10-30 03:01:37,004 - rafa ficou inativo
45 2023-10-30 03:01:54,859 - rafa desconectou-se da rede
46 2023-10-30 03:02:00,777 - rafa conectou-se
47 2023-10-30 03:02:00,777 - rafa ficou inativo
48 2023-10-30 12:16:47,226 - Sarah amor da vida do Rafinha realizou cadastro
49 2023-10-30 12:16:47,226 - Sarah amor da vida do Rafinha ficou inativo
50 2023-10-30 12:17:04,548 - Sarah realizou cadastro
51 2023-10-30 12:17:04,548 - Sarah ficou inativo
52 2023-10-30 12:20:35,985 - Sarah amor da vida do rafinha realizou cadastro
53 2023-10-30 12:20:35,985 - Sarah amor da vida do rafinha ficou inativo
54 2023-10-30 12:28:30,084 - Sarah amor da vida do rafinha realizou cadastro
55 2023-10-30 12:28:30,084 - Sarah amor da vida do rafinha ficou inativo
56 2023-10-30 12:28:43,696 - Sarah amor da vida do rafinha nao responde ('morreu')
57 2023-10-30 12:45:27,807 - Sarah amor da vida do rafinha realizou cadastro
58 2023-10-30 12:45:27,807 - Sarah amor da vida do rafinha ficou inativo
59 2023-10-30 12:48:21,341 - rafa realizou cadastro
60 2023-10-30 12:48:21,342 - rafa ficou inativo
61 2023-10-30 12:49:42,735 - rafa desconectou-se da rede
62 2023-10-30 13:17:12,758 - rafa realizou cadastro
63 2023-10-30 13:17:12,759 - rafa ficou inativo
64 2023-10-30 13:19:15,762 - saritch realizou cadastro
65 2023-10-30 13:19:15,762 - saritch ficou inativo
66 2023-10-30 13:20:17,694 - saritch ficou ativo
67 2023-10-30 13:20:17,694 - rafa ficou ativo
68 2023-10-30 13:20:17,694 - Usuarios saritch e rafa: PLAYING
69 2023-10-30 13:22:47,468 - saritch conectou-se
70 2023-10-30 13:22:47,469 - saritch ficou inativo
71 2023-10-30 13:22:47,526 - rafa conectou-se
72 2023-10-30 13:22:47,527 - rafa ficou inativo
73 2023-10-30 13:23:09,627 - rafa ficou ativo
74 2023-10-30 13:23:09,627 - saritch ficou ativo
75 2023-10-30 13:23:09,627 - Usuarios rafa e saritch: PLAYING
76 2023-10-30 13:25:13,997 - rafa realizou cadastro
77 2023-10-30 13:25:13,998 - rafa ficou inativo
78 2023-10-30 13:25:31,105 - rafa desconectou-se da rede
79 2023-10-30 13:25:33,758 - saritcha realizou cadastro
80 2023-10-30 13:25:33,758 - saritcha ficou inativo
```

Figura 4: Arquivo game.log

3.3 Outras Implementações

Servidor Concorrente: O servidor de autenticação não participa do jogo, sendo utilizado para listar e conectar clientes (PeerToPeer).



4 Executar o Programa

Para a criação do jogo Blackjack utilizou-se à linguagem de programação *Python* e *makefile* para execução. Para executar o Servidor de Autenticação e Informação (SAI), escreva os seguintes comandos:

```
make servidor
```

Em outro terminal ou computador dentro da mesma rede, escreva os comandos a seguir para executar o cliente:

```
make cliente
```

