Bernardo Morais Alfredo – 565524

Sumário

1.1 PROBLEMA 1 – BATALHA NAVAL - O PROBLEMA

O problema da batalha naval consiste em projetar um jogo de batalha naval onde existe um tabuleiro do cliente e um tabuleiro do servidor, os dois estão jogando a mesma partida, o cliente inicia a partida atacando o servidor, caso ele acerta o servidor retorna uma mensagem informando que o cliente acertou, caso erre o servidor informa para o cliente que ele errou e realiza sua jogada, o servidor continua jogando até errar. Caso um dos lados perca todos os navios ele reporta para o outro que ele perdeu, encerrando assim a partida. Cliente-servidor devem se comunicar via TCP.

O tabuleiro possui um tamanho 10x10 e não é compartilhado entre os jogadores. Existem os seguintes navios para cada jogador:

1- Um porta-aviões, cada um ocupa 5 quadrados
2- Dois navios-tanque, cada um ocupa 4 quadrados
3- Três contratorpedeiros, cada um ocupa 3 quadrados
4- Quatro submarinos, cada um ocupa 2 quadrados

O problema foi solucionado em python com auxilio das bibliotecas socket, ipaddress, _thread, sys, random, time e numpy.

1.2 IMPLEMENTAÇÃO DO CLIENTE

O programa cliente, ao ser inicializado, pedirá para que seja digitado *cliente <ip> <porta>*, enquanto não detectar essa entrada ele reportará *Dados inválidos*. Caso os dados digitados sejam válidos, porém o programa não estabeleça uma conexão com o servidor, ele irá encerrar.

Caso os dados para conexão sejam válidos, ele irá estabelecer uma conexão com o servidor, avisará que o jogo começou e irá esperar pela primeira jogada. Se o jogador cliente acertar, ele poderá jogar novamente, caso contrário a próxima rodada é do servidor, até que ele erre, todas as jogadas efetuadas pelo servidor são reportadas ao cliente, informando erro, acerto, ou o final da partida.

O programa cliente.py possui um método denominado **shot()**, que é chamado na rodada em que o cliente deve enviar uma coordenada de tiro para o servidor, através da função **.sendall()** da biblioteca socket, que envia uma mensagem TCP para o servidor, contendo a informação de que o cliente enviou um tiro nas coordenadas x,y.

O programa cliente.py possui um loop infinito ao final do arquivo, responsável por realizar a comunicação constante com o servidor, o loop acaba quando o servidor recusar ou encerrar a conexão no socket estabelecido anteriormente. O cliente está preparado para receber 5 tipos de mensagens do servidor, sendo elas: StartGame, informando o começo da partida; HIT, informando que você acertou o servidor e é sua vez novamente; MISS, informando que você errou e é a rodada do oponente; SHOT, recebendo a coordenada em que o servidor atirou e respondendo para ele se ele acertou ou errou, se errar passa a rodada para você; END, informando o fim da partida e sua vitória ou derrota.

O programa cliente.py possui as declarações básicas para utilização da biblioteca socket e comunicação com o servidor, consiste também na classe Tabuleiro, definida abaixo:

Sempre que o servidor acertar um tiro ou o cliente for realizar um disparo, é imprimido na tela o campo inimigo e o campo do cliente.

Class Tabuleiro:

campo -- propriedade (matriz) que define o campo do jogador
campo_inimigo -- propriedade (matriz) que define acertos no inimigo
quantidade_a -- propriedade (int) que define quantidade de Porta-Aviões
quantidade_n -- propriedade (int) que define quantidade de Navios-Tanques
quantidade_c -- propriedade (int) que define quantidade de Contratorpedeiros
quantidade_s -- propriedade (int) que define quantidade de Porta-Aviões

def __init__
-- Método que inicializa o objeto
def setEnemyField
-- Método que inicializa o campo inimigo
def get (x,y)
-- Retorna o valor do campo na posição (x,y)
def set (x,y,valor)
-- Define o valor do campo na posição (x,y), sempre que receber um tiro e trocar um pedaço de navio por X, esse pedaço é decrementado de sua respectiva quantidade.
def setEnemy (x,y,valor)
- Quando acertar o inimigo marca a posição acertada e agora conhecida do campo inimigo
def insertAircraftCarrier (x,y,orientação)
-- Insere um porta-avião a partir de (x,y) vertical ou horizontalmente
def insertTankShip (x,y,orientação)
-- Insere um navio-tanque a partir de (x,y,) vertical ou horizontalmente
def insertDestroyer (x,y,orientação)
-- Insere um contratorpedeiro a partir de (x,y,) vertical ou horizontalmente
def insertSub (x,y,orientação)
-- Insere um submarino a partir de (x,y,) vertical ou horizontalmente
def setShipsPositions
-- Localiza as coordenadas para inserir os navios no campo do tabuleiro do jogador com base no arquivo coordenadas.txt, que deve estar no diretório do programa.
O arquivo é lido linha a linha e deve conter, por linha:
(linha do tabuleiro) (coluna do tabuleiro) (orientação) (tipo de navio)
A linha deve ser um inteiro de 0-9
A coluna deve ser um inteiro de 0-9
A orientação deve ser um inteiro, sendo 0 (horizontal) e 1(vertical)
O tipo é A para porta-aviões, N para navio-tanque, C para contra torpedeiro, S para submarino.

Obs: Caso o ataque do cliente não seja "x,y" ele irá reportar erro, obrigatoriamente deve-se utilizar uma virgula!

1.3 IMPLEMENTAÇÃO DO SERVIDOR

O programa servidor, ao ser inicializado, pedirá para que seja digitado *servidor <porta>*, enquanto não detectar essa entrada ele reportará *Dados inválidos*. Caso os dados digitados sejam válidos ele irá prosseguir.

Após validar os dados, será criado um novo socket com a porta especificada que irá escutar por pacotes enviados por um cliente, quando detectar que recebeu um pacote contendo ***StartConnection*** ele iniciará a partida, criando um tabuleiro para o servidor e exibe-o após finalizar a criação, a classe tabuleiro do servidor possui algumas diferenças se comparada com a do cliente, serão explanadas mais a frente. O servidor então estabelece uma conexão para esse cliente em uma nova thread e informa que o cliente está conectado enviando a mensagem ***StartGame*** para o cliente e aguarda a resposta com a jogada do cliente, dando assim início a partida.

O servidor encerra a partida caso o socket seja fechado ou perceba que todos os seus navios foram afundados, avisando o cliente de que ele venceu. Caso o servidor receba uma mensagem do tipo ***SHOT*** ele verifica se o cliente acertou ou errou o tiro, se tiver acertado informa ao cliente que ele acertou e avisa para o cliente atirar novamente, caso o cliente tenha errado o tiro a rodada passa para o servidor que realiza um tiro em uma posição aleatória válida no tabuleiro inimigo, se o servidor receber ***HIT*** significa que ele acertou e então atira novamente, caso receba ***MISS***, significa que errou, passando a rodada para o cliente, esse loop continua até que um dos dois feche a conexão ou vença a partida.

O programa servidor.py possui as declarações básicas para estabelecer conexões com o auxílio da biblioteca socket na porta passada como parâmetro, o método getTime que retorna a hora atual do servidor para reportar no log, uma classe Cliente, que define um cliente conectado ao servidor, tendo os atributos nome, um socket denominado conexão e o endereço do cliente, denominado clientaddress.

O programa servidor.py possui o método ***shot(conexão)*** que recebe uma conexão (objeto do tipo socket) na qual deve enviar, via TCP, as coordenadas de tiro, de forma aleatória, sem ultrapassar o tamanho máximo do tabuleiro.

O programa servidor.py possui o método ***startGame(cliente)*** que recebe um cliente o qual iniciou uma partida com o servidor, aqui ocorre o envio e recebimento das jogadas de cada lado, com as mensagens definidas anteriormente.

O programa servidor.py possui o método ***startConnection()*** que é responsável por escutar novas partidas de possíveis clientes, criando uma thread para cada partida estabelecida com algum cliente.

A classe tabuleiro do programa servidor.py está definida abaixo:

Class Tabuleiro:

campo -- propriedade (matriz) que define o campo do jogador
quantidade_a -- propriedade (int) que define quantidade de Porta-Aviões
quantidade_n -- propriedade (int) que define quantidade de Navios-Tanques
quantidade_c -- propriedade (int) que define quantidade de Contratorpedeiros
quantidade_s -- propriedade (int) que define quantidade de Porta-Aviões

def __init__
-- Método que inicializa o objeto
def setEnemyField
-- Método que inicializa o campo inimigo
def get (x,y)
-- Retorna o valor do campo na posição (x,y)
def set (x,y,valor)
-- Define o valor do campo na posição (x,y), sempre que receber um tiro e trocar um pedaço de navio por X, esse pedaço é decrementado de sua respectiva quantidade.
def insertAircraftCarrier (x,y,orientação)
-- Insere um porta-avião a partir de (x,y) vertical ou horizontalmente
def insertTankShip (x,y,orientação)
-- Insere um navio-tanque a partir de (x,y,) vertical ou horizontalmente
def insertDestroyer (x,y,orientação)
-- Insere um contratorpedeiro a partir de (x,y,) vertical ou horizontalmente
def insertSub (x,y,orientação)
-- Insere um submarino a partir de (x,y,) vertical ou horizontalmente
def checkPosition (x,y, orientação, posição)
-- Checa se a posição recebida para inserir é válida
def pickPos (x)
-- Gera uma posição (linha, coluna, orientação) aleatória
def setShipsPositions
-- insere os navios no tabuleiro, de forma aleatória utilizando a biblioteca random.

1.4 TESTES

- Servidor Inicializado na porta 8080:

```
Para começar digite:
servidor <porta>

servidor 8080
```

- Cliente estabelece a conexão no ip da máquina (192.168.100.29) na porta 8080:

```
Para começar digite:
cliente <ip> <porta>

cliente 192.168.100.29 8080
[['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' 'C' 'C' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' ' ' ' ' 'S' 'S' ' ' 'N' 'N' 'N' 'N']
 ['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'C' ' ' 'N' ' ' ' ' ' ' 'S' ' ' ' ']
 [' ' 'C' ' ' 'N' 'S' 'S' ' ' 'S' ' ' ' ']
 [' ' ' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' ' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' 'C' 'C' 'C' ' ' ' ' ' ' ' ' ' ' ' ']]
Enviado -> StartConnection CLIENTE
O Jogo Começou!
Sua vez de atirar!
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar:
```

- Servidor exibe conexão e partida do cliente conectado:

```
servidor 8080
Connected by ('192.168.100.29', 59177)
MensagemRecebida = StartConnection CLIENTE
[['C' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['C' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['C' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'C' 'C']
 [' ' ' ' 'N' ' ' 'S' 'S' 'S' ' ' 'C' 'C']
 ['A' 'A' 'A' 'A' 'A' ' ' 'S' ' ' 'C' 'C']
 [' ' ' ' 'S' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'S' ' ' ' ' 'N' 'N' 'N' 'N' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' 'S' 'S' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']]
24/05/2020 14:32:44: Server conectado por CLIENTE
O Jogo Começou!
```

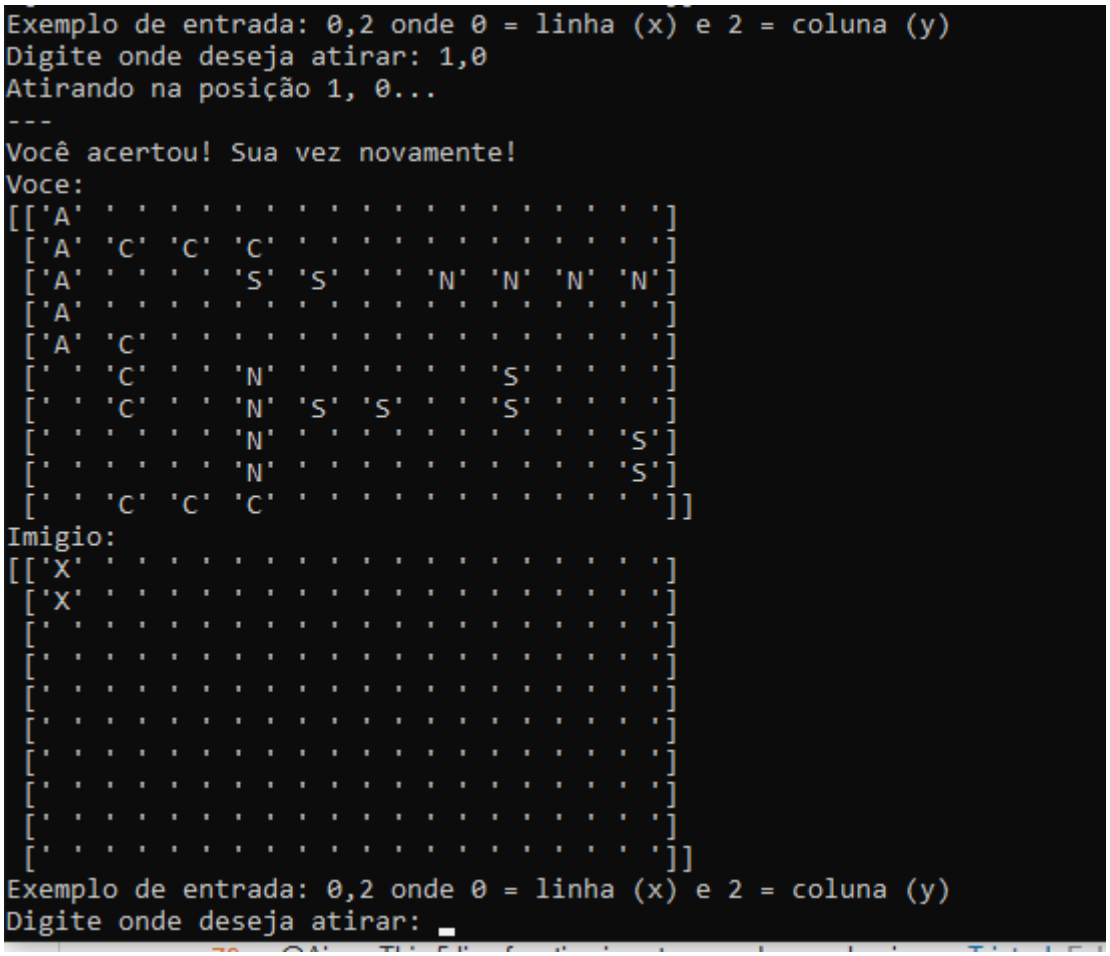
- Cliente ataca em 0,0

```
C:\Windows\system32\cmd.exe
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar: 0,0
Atirando na posição 0, 0...
---
Você acertou! Sua vez novamente!
Voce:
[['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' 'C' 'C' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' ' ' ' ' 'S' 'S' ' ' 'N' 'N' 'N' 'N']
 ['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'C' ' ' 'N' ' ' ' ' ' ' 'S' ' ' ' ']
 [' ' 'C' ' ' 'N' 'S' 'S' ' ' 'S' ' ' ' ']
 [' ' ' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' ' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' 'C' 'C' 'C' ' ' ' ' ' ' ' ' ' ' ' ']]
Imigio:
[['X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']]
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar: _
```

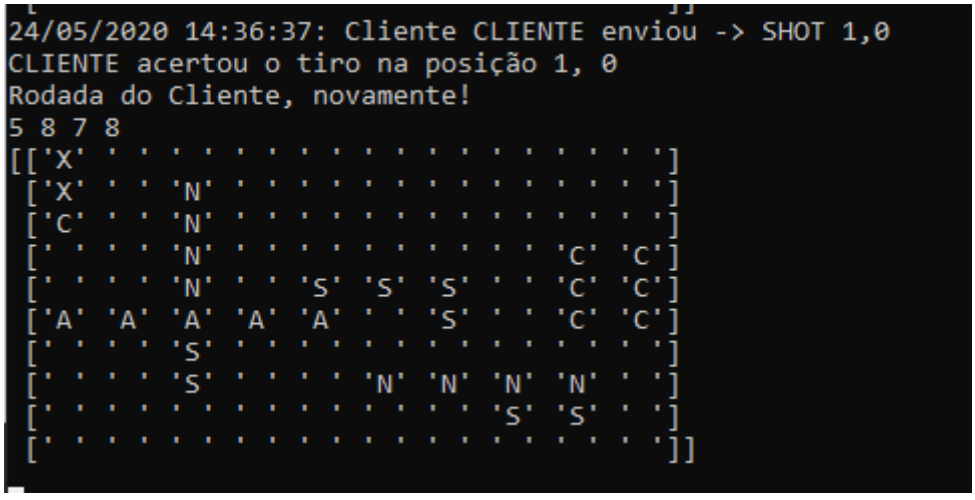
- Servidor recebe o tiro e reporta o acerto do cliente:

```
O Jogo Começou!
24/05/2020 14:35:19: Cliente CLIENTE enviou -> SHOT 0,0
CLIENTE acertou o tiro na posição 0, 0
Rodada do Cliente, novamente!
5 8 8 8
[['X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['C' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['C' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'C' 'C']
 [' ' ' ' 'N' ' ' 'S' 'S' 'S' ' ' 'C' 'C']
 ['A' 'A' 'A' 'A' 'A' ' ' 'S' ' ' 'C' 'C']
 [' ' ' ' 'S' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'S' ' ' ' ' 'N' 'N' 'N' 'N' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' 'S' 'S' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']]
```

- Cliente atira em 1,0 e acerta novamente:



- Servidor reporta o acerto:



- Cliente atira em 0,1 e erra:

```
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar: 0,1
Atirando na posição 0, 1...
---
Você errou, rodada do oponente!
---
Servidor errou o tiro na posicao 8, 1
---
Sua vez!
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar:
```

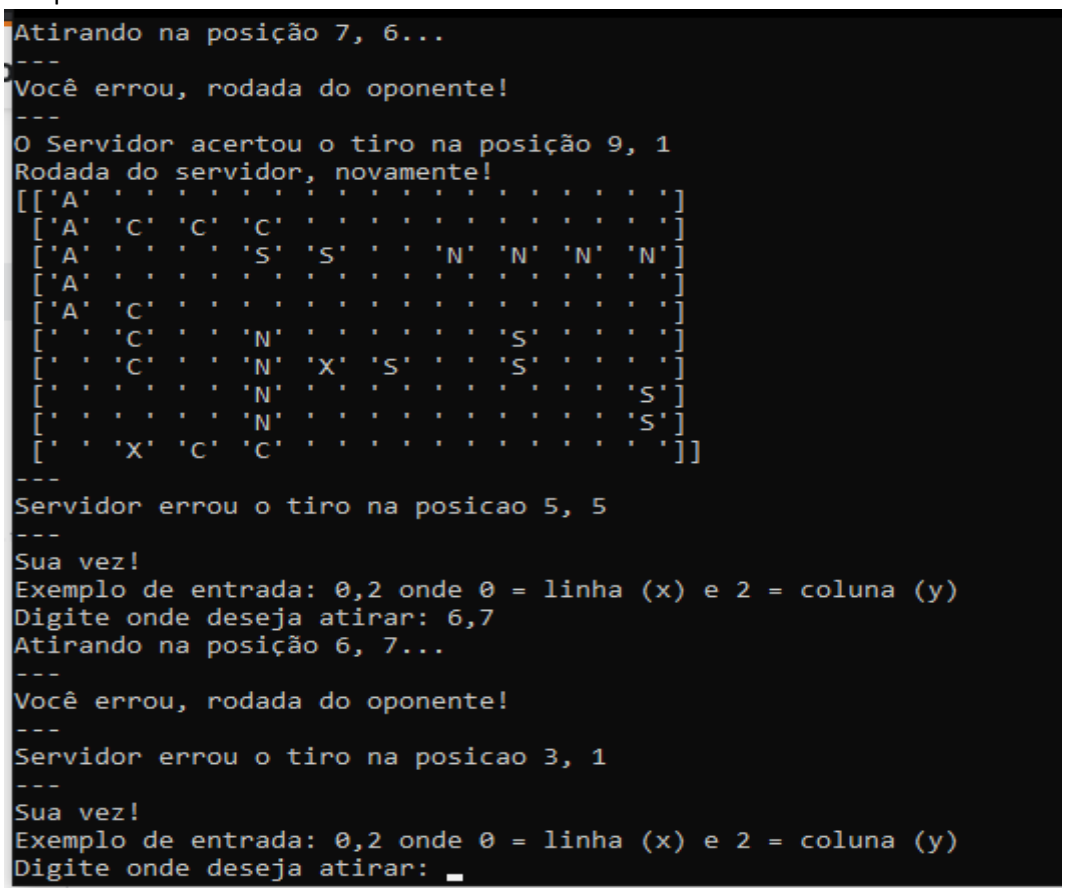
- Servidor recebe o erro e realiza sua jogada:

```
24/05/2020 14:37:38: Cliente CLIENTE enviou -> SHOT 0,1
CLIENTE errou o tiro na posição 0, 1
Rodada do Servidor!
Atirando na posição 8, 1...
[['X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['X' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['C' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' 'C' 'C']
 [' ' ' ' 'N' ' ' 'S' 'S' 'S' ' ' 'C' 'C']
 ['A' 'A' 'A' 'A' 'A' ' ' 'S' ' ' 'C' 'C']
 [' ' ' ' 'S' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'S' ' ' ' ' 'N' 'N' 'N' 'N' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' 'S' 'S' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']]
24/05/2020 14:37:38: Cliente CLIENTE enviou -> MISS
Você errou, rodada do oponente!
[['X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['X' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['C' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' ' 'C' 'C']
 [' ' ' ' 'N' ' ' 'S' 'S' 'S' ' ' 'C' 'C']
 ['A' 'A' 'A' 'A' 'A' ' ' 'S' ' ' 'C' 'C']
 [' ' ' ' 'S' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'S' ' ' ' ' 'N' 'N' 'N' 'N' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' 'S' 'S' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']]
```

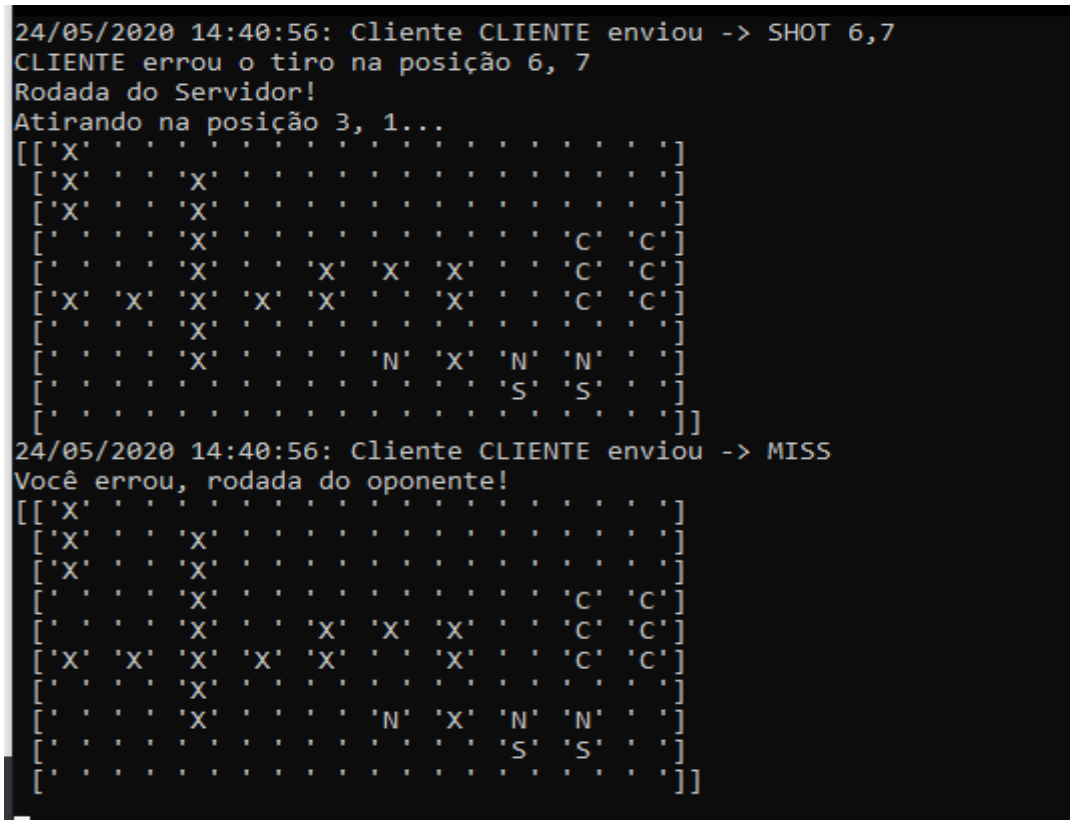
- Servidor erra e passa a rodada para o cliente

- A partida continua ...

- Depois de vários tiros o lado do cliente:



- O lado do servidor:



- Cliente burro (eu) sem querer fecha o cmd do cliente e finaliza a conexão com o servidor:

```
[['X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['X' ' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['X' ' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' 'C' 'C']
 [' ' ' ' 'X' ' ' 'X' 'X' 'X' ' ' ' ' 'C' 'C']
 ['X' 'X' 'X' 'X' 'X' ' ' 'X' ' ' ' ' 'C' 'C']
 [' ' ' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' 'X' ' ' ' ' 'X' 'X' 'X' 'X' ' ' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X' 'X' ' ']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']]
24/05/2020 14:43:12: Cliente CLIENTE finalizou a conexão.
Finalizando conexão com CLIENTE
```

- Novo cliente se conecta:

```
Para começar digite:
cliente <ip> <porta>

cliente 192.168.100.29 8080
[['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' 'C' 'C' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' ' ' ' ' 'S' 'S' ' ' 'N' 'N' 'N' 'N']
 ['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'C' ' ' 'N' ' ' ' ' ' ' 'S' ' ' ' ' ']
 [' ' 'C' ' ' 'N' 'S' 'S' ' ' 'S' ' ' ' ' ']
 [' ' ' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' ' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' 'C' 'C' 'C' ' ' ' ' ' ' ' ' ' ' ' ']]
Enviado -> StartConnection CLIENTE
O Jogo Começou!
Sua vez de atirar!
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar:
```

```
Finalizando conexão com CLIENTE
Connected by ('192.168.100.29', 59275)
MensagemRecebida = StartConnection CLIENTE
[[' ' 'N' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'N' 'A' 'A' 'A' 'A' 'A' ' ' ' ' ' ' ']
 [' ' 'N' ' ' 'C' 'S' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'N' ' ' 'C' 'S' ' ' 'S' 'S' ' ' ' ' ']
 [' ' ' ' ' ' 'C' ' ' 'C' 'C' 'C' ' ' ' ' ']
 ['S' 'S' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' 'C' 'C' 'C' ' ' ' ' ']
 [' ' ' ' ' ' 'N' 'N' 'N' 'N' ' ' ' ' ' ']]
24/05/2020 14:45:10: Server conectado por CLIENTE
O Jogo Começou!
```

- Uma longa batalha se inicia...

- Falta um tiro para o cliente vencer:

```
 [' ' ' ' ' ' 'X' 'X' 'X' 'X' ' ' ' ' ' ' ']]
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar: 4,6
Atirando na posição 4, 6...
---
Você acertou! Sua vez novamente!
Voce:
[['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' 'C' 'C' ' ' ' ' ' ' ' ' ' ' ' ']
 ['X' ' ' ' ' 'X' 'S' ' ' 'N' 'N' 'N' 'N']
 ['A' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 ['A' 'C' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'C' ' ' 'N' ' ' ' ' ' ' 'S' ' ' ' ']
 [' ' 'C' ' ' 'N' 'S' 'S' ' ' 'S' ' ' ' ']
 [' ' ' ' ' ' 'N' ' ' ' ' ' ' ' ' ' ' 'S']
 [' ' ' ' ' ' 'X' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' 'C' 'C' 'X' ' ' ' ' ' ' ' ' ' ' ' ']]
Imigio:
[[' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' 'X' 'X' 'X' 'X' 'X' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' 'X' 'X' ' ' ' ']
 [' ' ' ' ' ' 'X' ' ' 'X' 'X' ' ' ' ' ' ']
 ['X' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' 'X' 'X' 'X' ' ' ' ' ']
 [' ' ' ' ' ' 'X' 'X' 'X' 'X' ' ' ' ' ' ']]
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar:
```

```
C:\Windows\system32\cmd.exe
Rodada do Cliente, novamente!
0 0 2 0
[[' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' 'X' 'X' 'X' 'X' 'X' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' 'X' 'X' ' ' ' ']
 [' ' ' ' ' ' 'X' ' ' 'X' 'C' 'C' ' ' ' ']
 ['X' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' 'X' 'X' 'X' ' ' ' ' ']
 [' ' ' ' ' ' 'X' 'X' 'X' 'X' ' ' ' ' ' ']]
24/05/2020 14:48:37: Cliente CLIENTE enviou -> SHOT 4,6
CLIENTE acertou o tiro na posição 4, 6
Rodada do Cliente, novamente!
0 0 1 0
[[' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' 'X' 'X' 'X' 'X' 'X' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' 'X' 'X' ' ' ' ']
 [' ' ' ' ' ' 'X' ' ' 'X' 'X' 'C' ' ' ' ']
 ['X' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' 'X' 'X' 'X' ' ' ' ' ']
 [' ' ' ' ' ' 'X' 'X' 'X' 'X' ' ' ' ' ' ']]
```

- O cliente acerta o disparo:

```
---
Sua vez!
Exemplo de entrada: 0,2 onde 0 = linha (x) e 2 = coluna (y)
Digite onde deseja atirar: 4,7
Atirando na posição 4, 7...
Voce venceu!!
Closing connection
```

```
[[' ' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' 'X' 'X' 'X' 'X' 'X' ' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' ' ' ' ' ' ' ' ' ']
 [' ' 'X' ' ' 'X' 'X' ' ' 'X' 'X' ' ' ' ' ']
 [' ' ' ' ' ' 'X' ' ' 'X' 'X' 'C' ' ' ' ' ']
 ['X' 'X' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' 'X']
 [' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ']
 [' ' ' ' ' ' ' ' ' 'X' 'X' 'X' ' ' ' ' ']
 [' ' ' ' ' ' 'X' 'X' 'X' 'X' ' ' ' ' ' ']]
24/05/2020 14:49:56: Cliente CLIENTE enviou -> SHOT 4,7
CLIENTE acertou o tiro na posição 4, 7
Rodada do Cliente, novamente!
Finalizando conexão com CLIENTE
```

1.5 CONCLUSÃO

Para a realização do trabalho final da matéria foi necessário estudar a biblioteca socket e compreender como se realiza uma comunicação TCP entre um cliente e um servidor, durante a implementação vi a necessidade de sempre se confirmar se os dados recebidos ou enviados estavam de acordo com as normas pré-estabelecidas, pois o simples fato de substituir a , em uma coordenada pode acarretar em consequências inesperadas. Aprendi também que o fato de o servidor atirar aleatoriamente sem nenhuma estratégia demonstra exatamente como não se deve jogar batalha naval. Optei por não ter de pressionar a tecla P para mostrar o tabuleiro, exibindo seu status sempre que ocorre alguma mudança. O tabuleiro lido do servidor nada mais é que uma matriz vazia onde a cada acerto confirmado pelo servidor em alguma coordenada é realizada uma marcação com X no tabuleiro, o perdedor sempre reporta ao vencedor que ele venceu.

1.6 REFERÊNCIAS

https://kharisecario.wordpress.com/2017/03/25/create-nxn-matrix-in-pythonnumpy/

https://realpython.com/python-sockets/

https://wiki.python.org.br/SocketBasico