Organização de Sistemas de Computação

Projeto Final

Engenharia da Computação - 2º período

Ricardo Pannain

Enzo Eiki Yasuda – 24000880

Vinícius Bertozzi da Cruz – 24003728

MANUAL DE USUÁRIO

Resumo:

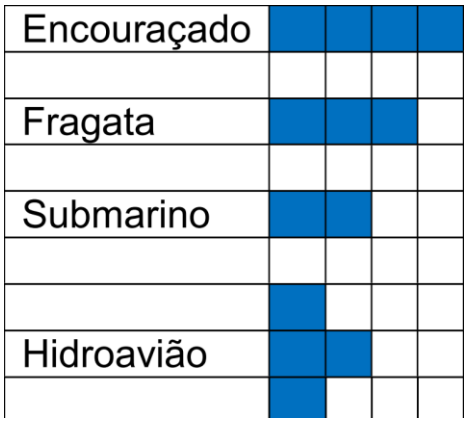
O programa tem como objetivo simular uma batalha naval usando uma matriz 20x20. Nele, irá ter uma escolha aleatória dentre 5 mapas, com posições das peças diferentes uma da outra. Após a escolha, aparecerá uma matriz em branco na tela (feitas apenas com 0), o que significa que o jogador deverá adivinhar as posições das peças para ganhar. Se acertar a posição da peça, aparecerá um “V” na posição escolhida, entretanto, se errar, aparecerá um “X”. Desse modo, o intúito do jogador é acertar todas as peças para assim vencer. Caso queira sair, a qualquer momento, poderá apertar apenas “ESC”, terminando o programa.

Para poder jogar no programa, será preciso instalar o VSCODE e rodar nele. Outro requisito é baixar a extenção: TASM dentro do VS. Após todo o procedimento, será preciso apenas apertar o botão direito na tela do programa e escolher a opção: Run ASM code. Todas as instruções estão dentro do programa.

REGRAS:

- O jogador não poderá escolher um número entre 1 e 20.
- Há 6 peças no total, sendo eles:
    - 1 Encouraçado
    - 1 Fragata
    - 2 Submarinos
    - 2 Hidroaviões

Aqui segue o tamanho e formato das peças:



- O jogador não poderá repetir posições já escolhidas.

INTERFACE:

- **Início:**

```
BEM VINDO AO JOGO DE BATALHA NAVAL!!!
Pressione ENTER para continuar.          Quer sair? Pressione ESC _
```

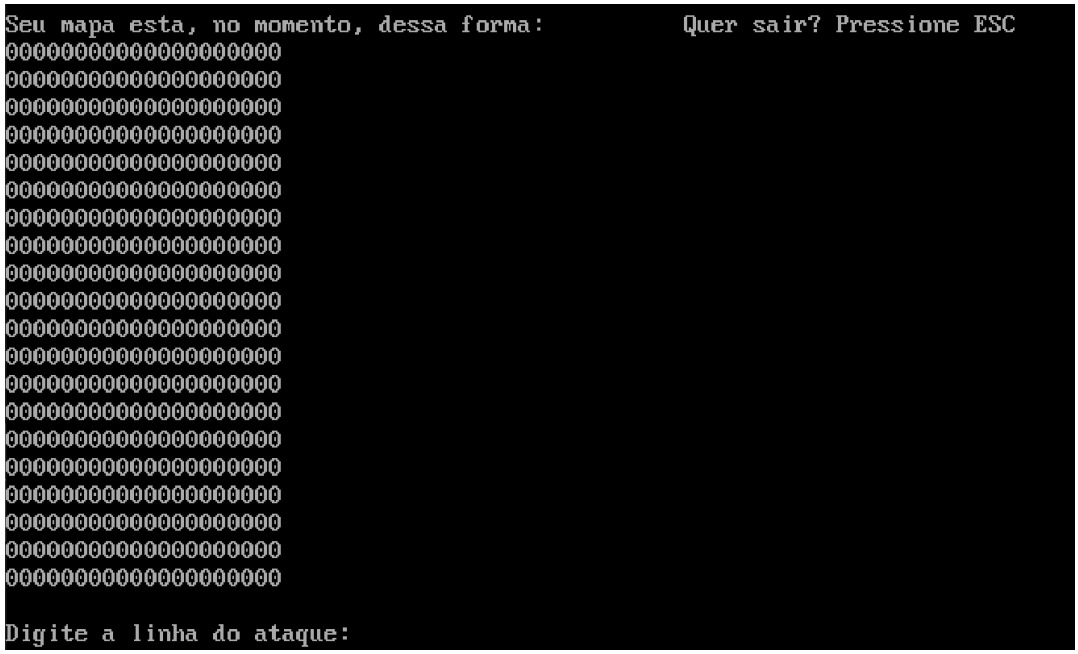
Aparecerá uma mensagem de bem vindo e a opção entre continuar e sair. A opção de sair estará em todo o jogo.

- **Instrução:**

```
O jogo escolheu um mapa aleatorio. O jogador deve escrever a linha e coluna do m
apa para o ataque.
Ao instruir corretamente, o lugar recebera um X, caso tenha ERRADO, ou um V, cas
o tenha ACERTADO.
Pressione ENTER para continuar.          Quer sair? Pressione ESC _
```

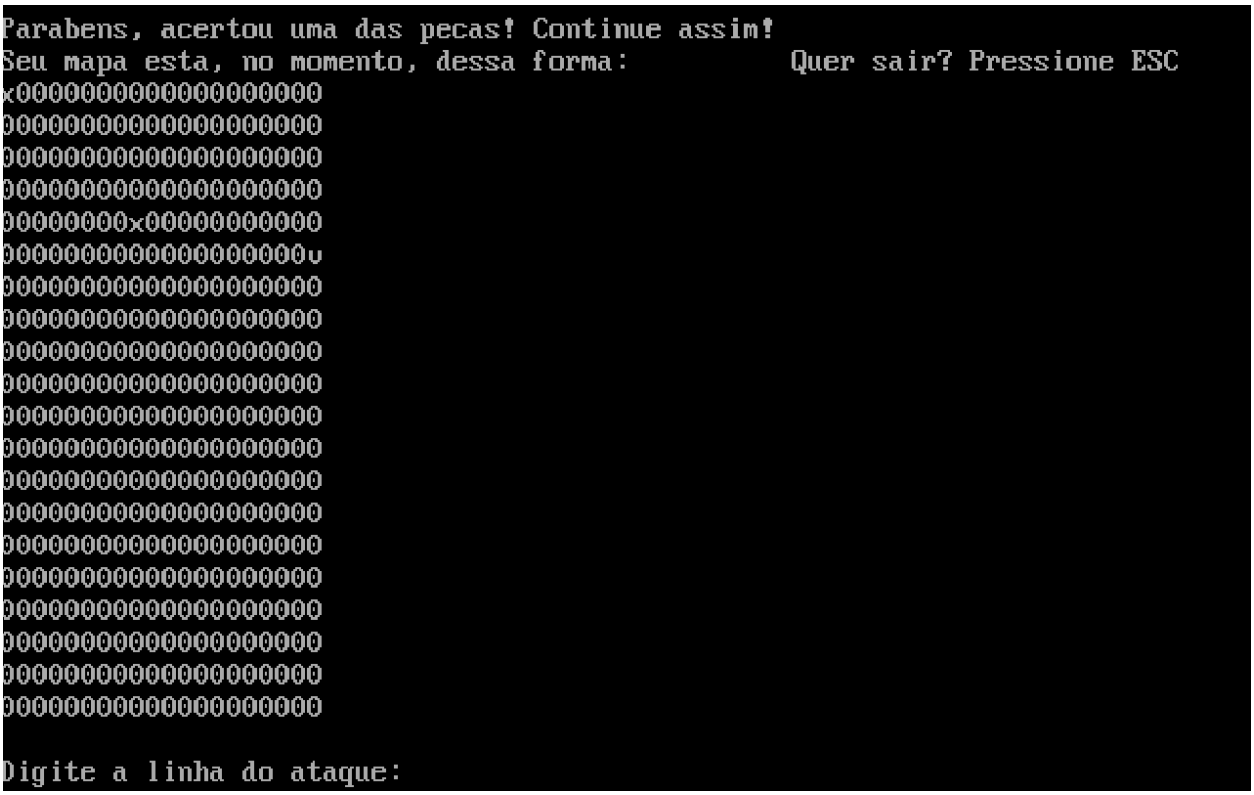
Aqui esta um breve resumo do que acontecerá no jogo e como ele funciona

- **BATALHA:**



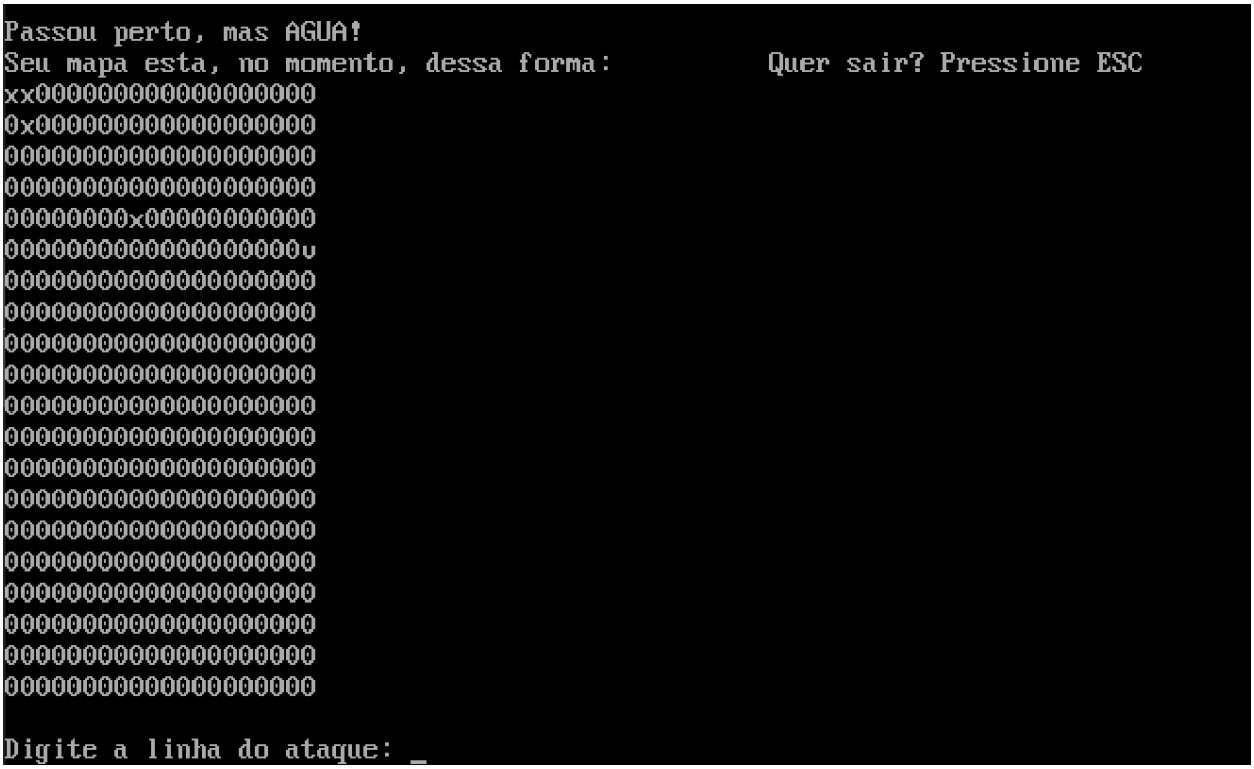
Irá imprimir a matriz em branco onde o jogador poderá pensar nas posições e afins. Ela começará a mudar com o decorrer do jogo.

- **Acertou:**



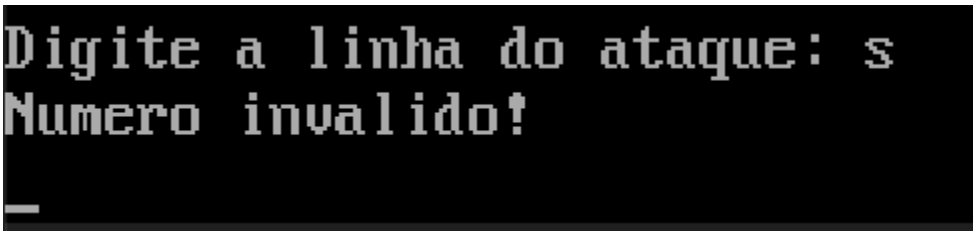
Caso o jogador tenha acertado, aparecerá uma mensagem logo acima e um “V” no lugar escolhido.

- **Errou:**



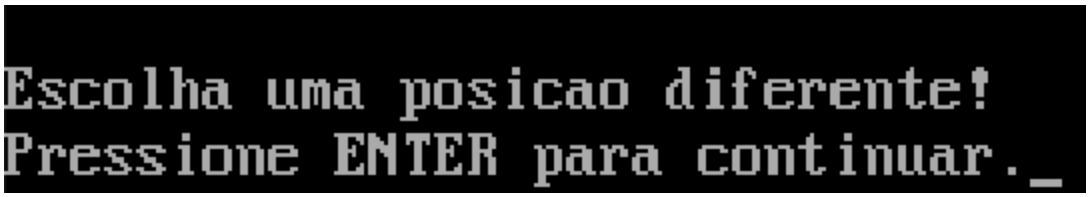
Caso o jogador tenha errado, aparecerá esta mensagem acima e um “X” na matriz na posição escolhida.

- **Inválido:**



Caso o jogador tenha colocado um caracter sem ser o autorizado, aparecerá uma mensagem de número invalido e pedirá para recolocar um novo caracter.

- **Repetiu:**



Caso o jogador repetir a posição, aparecerá uma mensagem alertando que o número já fora usado.

- **GANHOU:**

Caso o jogador ganhe o jogo, a interface de BATALHA irpa fechar e aparecerá uma mensagem motivadora.

- **Término**

Caso o jogador aperte “ESC” em qualquer momento do jogo, ou aperte ENTER depois de ganhar, aparecerá esta mensagem e o programa encerrará.

RESPOSTAS:

Aqui está a posição das peças nos diferentes mapas:

- **MAPA1**

```
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1
DW 0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0
DW 0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

- **MAPA2**

```
DW 1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,1,0,0,0
DW 0,0,0,0,0,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

- **MAPA3**

```
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
DW 0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

- **MAPA4**

```
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0
DW 1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0
DW 0,0,1,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0
```

- **MAPA5**

```
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0
DW 0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DW 0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Algumas observações sobre o Programa:

1) **MAIN PROC**

No main, irá ter todas as principais chamadas de procedimentos e MACROs, assim deixando o programa principal mais curto e entendível. Nele, se encontra as mensagens de recepção, instrução e de despedida, assim como a chamada dos procedimentos de batalha.

2) **BATALHA PROC**

Neste procedimento, está toda a parte de ação do programa, tendo a impressão do mapa em branco para o jogador e a escolha do MAPA que irá ser atacado, a leitura das posições do mapa e o ataque em si.

3) DECIMAL PROC

Este procedimento foi inspirado, majoritariamente, pelo Lab11 da aula prática de Organização de Sistemas de Computação, o qual tem um programa de entrada decimal, usado, portanto, para a entrada das posições, visando que há mais de uma casa decimal. Ex: 16,18,20 (max). Desse modo, ele foi adaptado para este programa para o valor de BX e SI.

4) RANDOM PROC e DEFINIR_MAPA PROC

Estes dois procedimentos não podem ser separados, uma vez que o PROC DEFINIR_MAPA só existe por conta do RANDOM. O procedimento RANDOM serve para escolher “aleatoriamente” o número do mapa. Ele funciona pegando os segundos e divindo até chegar em um número de 1 a 5. Para o PROC DEFINIR_MAPA, nele pega o valor escolhido em RANDOM e filtra até chegar no número específico dentre 1 a 5. Assim, ele começa a escrever os valores da matriz escolhida em uma cópia que será usada para o resto do programa de BATALHA.