



Game of TSI



Jogo desenvolvido como desafio da VI Semana Acadêmica de TSI.



Informações do Arquivo do Jogo

- Versão: 1
- Desenvolvido em Java 8 sem nenhuma dependência externa.
- Compatível com Windows, Linux e Mac.
- Tamanho: ~700kb.
- Para rodar o jogo, clique 2 vezes no arquivo JAR ou execute-o pelo terminal “java -jar jogo.jar”.

Como Jogar













- O jogo sempre é jogado entre 2 jogadores, sendo um contra o outro.
- Cada jogador possui um castelo e o primeiro que destruir o castelo inimigo vence. Caso o tempo acabe, o jogador com o castelo com mais vida vence. No caso de empate, os dois jogadores perdem.
- Para atacar o inimigo, o jogador deverá comprar unidades para a **Guerra de TSI**. O jogador deverá decidir somente quando e qual unidade comprar. Depois de comprar uma unidade, ela se moverá e atacará sem interferência do jogador. O jogador com a melhor estratégia vence.

- Os jogadores ganham peças de ouro conforme o tempo passa. Elas devem ser usadas para comprar unidades.

Unidades

Abaixo estão unidades disponíveis no jogo. Crédito das imagens:

<https://orkimides.deviantart.com/art/Fight-of-Thrones-410299344>

Personagem	Unidade	Características
 Tyrion Lannister		Preço: 3 Vida: 10 Ataque: 3 (perto)
 Melisandre		Preço: 4 Vida: 7 Ataque: 2 (longe)
 Hodor		Preço: 5 Vida: 30 Ataque: 3 (perto)
 Jon Snow		Preço: 10 Vida: 21 Ataque: 10 (perto)
 Daenerys Targaryen		Preço: 17 Vida: 14 Ataque: 3 (muito longe) Especial: ataca todos os inimigos no alcance.
 Gregor Clegane		Preço: 17 Vida: 100 Ataque: 4 (perto)

Programação do Jogador

Abaixo está a classe de exemplo da jogadora Maria.

```
public class Maria {
    // ----- AÇÕES -----
    private static final int WAIT      = -1; // NÃO COMPRA NENHUMA UNIDADE
    private static final int TYRION    = 0;
    private static final int MELISANDRE = 1;
    private static final int HODOR     = 2;
    private static final int JON_SNOW  = 3;
    private static final int DAENERYS  = 4;
    private static final int CLEGANE   = 5;
    // -----

    public int play(int gold, // OURO
        int[] units, // UNIDADES ALIADAS.
        // units[TYRION] é a quantidade de TYRION aliados
        int[] enemies, // UNIDADES INIMIGAS
        // enemies[TYRION] é a quantidade de TYRION inimigos
        int castle, // VIDA DO CASTELO ALIADO
        int castleEnemy // VIDA DO CASTELO INIMIGO
    ) {

        return TYRION; // retorna a AÇÃO
    }

    public String getName() {
        return "Maria";
    }
}
```

- O exemplo possui o mínimo necessário para jogar.
- As constantes servem para auxiliar a programação. Elas representam o resultado do método “play”.
- Coloque o seu nome no método “getName”. Ele será exibido no jogo.
- O método “play” é onde ficará a estratégia do jogador. Ele recebe por parâmetro as informações da guerra e, com base nelas, deve decidir se compra alguma unidade ou se espera o próximo turno. Para não esperar, basta retornar “WAIT”.
- O valor retornado pelo “play” é a unidade que se deseja comprar. Ela só será comprada caso o valor seja válido e o jogador tenha o ouro suficiente.
- O método “play” será chamado várias vezes por segundo.
- O jogo instancia um objeto da sua classe automaticamente, portanto você poderá criar métodos e atributos na sua classe. O estado do objeto é preservado entre as chamadas do método “play”.
- O código-fonte dos jogadores devem estar na pasta “players”.

Proibições

- Não use o nome da sua classe no seu código, pois quando o jogo carregar a sua classe, ele alterará o nome da classe. Exemplo do que **não** se deve fazer no código: `Maria.atributo = 3;`
- Não pode manipular processos ou threads.
- Não pode manipular arquivos ou acessar a rede.
- Não pode depender de bibliotecas externas. Use somente o que tem disponível no JDK.
- Não pode tentar obter informações do jogo ou de outros jogadores. Todas as informações necessárias para o jogador serão passadas por parâmetro para o método “play”.
- Caso o código do aluno não compile, ele perderá a guerra.
- Caso o método “play” gere uma exceção (exemplo: `NullPointerException`), a ação do jogador será desconsiderada.

Competição

- Os alunos poderão enviar o seu código-fonte pelo site da semana acadêmica. Podem ser enviadas várias vezes por dia. Será enviado somente um arquivo Java.
- O jogo é executado uma vez por dia no nosso servidor. Depois de executado, é gerado um ranking com a classificação de cada aluno.
- Todos os alunos jogarão contra todos os alunos. Quem ganhar um duelo, ganha um ponto na classificação. A partir da pontuação é que será gerado o ranking.
- Não será exibida a classificação dos 4 primeiros alunos, assim ninguém saberá quem está ganhando. Durante a semana acadêmica será exibida uma disputa entre os 4 primeiros para decidir o vencedor do desafio.