* 1º) Devem ser utilizadas pelo menos duas classes, contemplando conceitos como encapsulamento, variáveis estáticas, atributos e métodos;

\* class Forca;

->Atributos:

- palavra;

- qtdLetras;

->Métodos:

- set Palavra;

- set qtdLetras;

- get Palavra;

- get qtdLetras;

- get verificaLetra;

\* class Player;

->Atributos:

- Nome;

- Score;

- palavraAtual;

- letrasDigitadas;

- tempoCorrido;

->Métodos:

- set Nome;

- set Score;

- get Nome;

- get Score;

* 2º) Ter um input com espaço para o usuário chutar a palavra de uma vez só; // Pode ser o mesmo input e verificar se é palavra ou letra e tratar diferente;
* 3º) Utilizar pelo menos uma função de getElementBy ou querySelector para buscar algum elemento no DOM; // Terá que ser utilizada de qualquer forma para pegar os elementos do DOM
* 4º) Utilizar o event listener para reconhecer SOMENTE LETRAS do alfabeto; //
* 5º) A palavra deve ir sendo completada no DOM caso o usuário acerte; //
* 6º) O boneco na forca deve ir sendo desenhado a cada novo erro do usuário; //
* 7º) Caso o usuário perca, deve aparecer a mensagem GAME OVER (ou algo parecido), e caso vença, deve aparecer YOU WIN (ou algo semelhante); //
* 8º) Permitir ao usuário jogar novamente uma nova partida. //

9º) LocalStorage e/ou SessionStorage para armazenar highscore (podem considerar o que quiserem como highscore, mas pensei aqui no tempo que o jogador levou pra acertar; quanto menor, melhor); - Cauã

EXTRA (300XP)

* - Acessar o conjunto de palavras a partir de um JSON e de forma ASSÍNCRONA; - +- ok

- LocalStorage deve ser usado para salvar o estado atual do jogo (se o jogador parar no meio e quiser voltar depois);

* - Ter um teclado virtual com as letras e funcionando de acordo. – Verificar API JS