**Lista 1 Humbertinho**

Nome: João César

**Questão 1:** Primeiramente iria configurar a IA para conseguir entender que tipo de estratégia iria fazer, se fosse eu para fazer, eu iria treina-la para fazer o modo aleatório, assim ela sempre irá jogar aleatoriamente após a minha jogada, ela pode muito bem ganhar ou perder, dependendo de como for a jogada, com o tempo a IA irá ter sua evolução assim podendo armazenar as jogadas que levaram a sua vitória, e evitar as que ela perdeu, e mesmo assim continuaria com a sua aleatoriedade de jogadas, mas sendo um pouco mais esperta e tendo um placar só de vitorias.

**Questão 2:** Eu iria fazer a abordagem aleatória, na qual a máquina seleciona aleatoriamente entre “pedra”, “papel” ou “tesoura”. É simples e imprevisível, mas não inteligente. Também iria tentar a abordagem de frequência, onde a máquina acompanha as escolhas passadas do oponente e escolhe a jogada que tem mais chances de vencer a opção mais frequente do adversário. A abordagem baseada em padrões, tentando identificar padrões nos movimentos do oponente para prever e contra-atacar na próxima jogada. Finalmente, eu iria fazer a abordagem de aprendizado de máquina, utilizando técnicas como árvores de decisão para que a máquina aprenda com jogos históricos e melhore sua estratégia ao longo do tempo. Cada abordagem oferece um nível crescente de sofisticação, desde a aleatoriedade básica até a inteligência adaptar.

**Questão 3:** O algoritmo Minimax é uma técnica comum em jogos de tabuleiro de dois jogadores, como o Connect 4 (ou Lig 4). A ideia é simular todas as possíveis jogadas futuras e avaliar os resultados, assumindo que ambos os jogadores jogam da melhor maneira possível. Vamos explicar como isso funciona em termos de árvore de decisão, semelhante ao que ocorre no jogo da velha.

Representação e Análise de Jogadas com Minimax

Estrutura da Árvore de Decisão:

No Connect 4, uma árvore de decisão é construída com base em todas as jogadas possíveis a partir de um determinado estado do tabuleiro.

Raiz da árvore (estado atual do tabuleiro): A raiz representa o estado atual do tabuleiro.

Nível 1 (jogadas do jogador atual): A partir da raiz, criam-se ramos representando as possíveis jogadas que o jogador atual (quem está para jogar) pode fazer. Em Connect 4, isso significa escolher uma das colunas ainda disponíveis.

Nível 2 (jogadas do adversário): Em seguida, cada um desses ramos leva a outro conjunto de ramos, representando as jogadas que o adversário pode fazer em resposta a cada movimento do jogador atual.

Nível 3 (jogadas do jogador atual): O processo continua alternando jogadas entre o jogador e o adversário até atingir um estado terminal, como a vitória, derrota ou empate.

Funcionamento e Lógica do Minimax:

Simulação das Jogadas: O algoritmo simula todos os possíveis movimentos do jogador e, em seguida, todos os possíveis movimentos do adversário, explorando todos os caminhos possíveis até que a vitória, derrota ou empate seja alcançada.

Valoração dos Estados Finais: No final de cada caminho (nó folha da árvore), o algoritmo atribui um valor de utilidade (ou avaliação) para cada estado final:

**Vitória: Se o jogador vence, o estado recebe uma pontuação positiva (geralmente +1).**

**Derrota: Se o jogador perde, o estado recebe uma pontuação negativa (geralmente -1).**

**Empate: Se o jogo termina em empate, o estado recebe uma pontuação neutra (geralmente 0).**

A partir das folhas, o algoritmo retropropaga esses valores para cima na árvore. Nos níveis onde é a vez do jogador, ele escolhe o valor máximo (tenta maximizar a sua chance de ganhar). Nos níveis onde é a vez do adversário, o adversário tentará minimizar a chance de vitória do jogador, ou seja, ele escolherá o valor mínimo.

Escolha da Melhor Jogada: No final da análise da árvore, o jogador seleciona o movimento que leva ao estado com o maior valor, ou seja, a melhor jogada possível levando em consideração que o adversário também jogará da melhor forma.

**Funcionamento das questões 4 e 5 no program.cs**

**Questão 4:**

Conceito do Algoritmo Minimax

O Minimax é baseado na criação de uma árvore de jogadas possíveis, onde cada nó representa um estado do jogo (um tabuleiro, no caso do jogo da velha). A ideia é simular todas as jogadas possíveis até o fim do jogo (ou até um estado terminal), atribuindo valores de utilidade a cada estado final (vitória, empate ou derrota) e, em seguida, propagar esses valores de volta pela árvore para determinar a melhor jogada no estado atual.

O Minimax alterna entre os dois jogadores, onde:

Maximizer: Um jogador que tenta maximizar o valor do estado (assumido como o jogador "X" no jogo da velha, por exemplo).

Minimizer: Um jogador que tenta minimizar o valor do estado (assumido como o jogador "O" no jogo da velha, por exemplo).

Passos do Minimax

Construção da Árvore de Decisão:

A árvore começa com o estado atual do tabuleiro no nó raiz.

Cada nó filho representa uma jogada possível (colocar um "X" ou "O" em uma célula vazia).

A árvore continua a se expandir recursivamente até atingir nós folha, que representam estados terminais do jogo (vitória, empate ou derrota).

Avaliação dos Estados Finais:

Quando o estado do jogo é terminal (o jogo terminou), uma pontuação é atribuída ao estado:

Vitória para "X": +1

Empate: 0

Vitória para "O": -1

Propagação dos Valores:

O valor de cada nó terminal é propagado para cima pela árvore.

Se é o turno do Maximizer, o valor máximo entre os nós filhos é escolhido.

Se é o turno do Minimizer, o valor mínimo entre os nós filhos é escolhido.

Escolha da Melhor Jogada:

No nível da raiz, o algoritmo escolhe a jogada que leva ao nó com o valor que maximiza o ganho para o jogador "X" (ou minimiza a perda para o jogador "O", dependendo do turno).

Regras de Pontuação

Central: Adicionar 2 pontos se a posição for a central (posição [1,1]).

Cantos: Adicionar 1 ponto se a posição estiver nos quatro cantos da matriz (posições [0,0], [0,2], [2,0], [2,2]).

Adversário: Subtrair 2 pontos se houver uma ou mais peças do adversário na mesma linha, coluna ou diagonal.

Impedir Vitória: Adicionar 4 pontos se a posição impedir a vitória do adversário.

Vitória: Adicionar 4 pontos se a posição levar a uma vitória para o computador.

**Questão 5:**

A estratégia está no modo aleatório ir tendo uma evolução a cada jogada

**No arquivo program.cs**