**INFO 6205 Spring 2022 Project**

***Menace***

**Group Member**: Peizhen Wu, Yutong Gao, Guangyu Wang, Chuntao Cai

**Introduction**:

**Aim**:

Implement “The Menace” by replacing matchboxes with values in a hash table.

Train the Menace by running games played against “human” strategy, which is based upon optimal strategy.

Implement logging: Log each training run with date/time, win/loss/draw.

Unit tests: run unit tests before we start training. Show the date/time of

our most recent unit test run.

Tests must have good coverage (each method must be covered).

Implement a GUI that shows the progress of a game of tic-tac-toe.

Implement a game server\* that can accept a challenge from a Menace.

**Approach**:

Menace: Record every move and the cheese state of every match in a HashMap, according the result of the match change the probability of taking same move of specific cheese state.

HumanStrategy: If the state of the cheese have two same pieces in one line, the human will choose the third place of this line to defend or win. Otherwise, human will choose a random place to move.

**Program**:

**Data Structures**: 1-D Array; 2-D Array; HashMap; ArrayList.

**Classes**: GameClient; GameClientDemo; GameServer; Player; Human; Player1-fixed; RandomHuman; Menace; Move; MoveDictionary; Train; TrainRandom; menaceTest; modelTest; randomHumanTest; Inter; MyOXGame; OXGamePVC; Start.

**Algorithm**:

We used random algorithm to train Menace and compare the win rate of the offensive position and the defensive position. We design the algorithm to put each move randomly and before it starts its next move, we must analyze the situation. For example, when Menace finished its second move, there’s chance for Menace to win the game in next step, so the random algorithm must stop Menace from winning. By the way, the game is over no matter which player wins the game, or the game ends in a tie.

**Invariants**:

Array, Boolean and Int: char[] tic = new char[9]; int n = tic.length; boolean gameOverFlag=false;

**Flow Charts (inc. UI Flow)**:

PVC:

Diagram

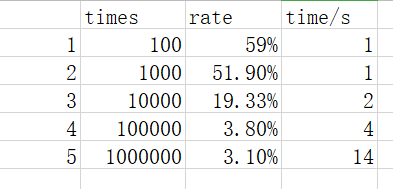
Description automatically generated

PVP: In order to run GameClientDemo twice, we should edit configuration -> click on “Modify options” and select “Allow multiple instances”.

图示

描述已自动生成

**Observations & Graphical Analysis**:



For the graph and sheet, we can conclude that the best choose of training times is 100,000 times. The lower times cannot get the good train result, the rate of lose is much higher than 3.8%. While we when increase the number, the train result is still stay around 3%. This will lead a waste of time and space.

**Results & Mathematical Analysis**:

In the process of training Menace, we were surprised to find that the probability of winning between the offensive position and the defensive position is quite similar. If Menace in the offensive position, then it has approximately more than 97 percent of not losing. if Menace in the defensive position, it at most has about 95 percent of not losing.

Human First:



Menace First:



**Testcases**:

menaceTest: test Menace’s next step and whether Menace wins, loses or draw the game.

文本

描述已自动生成

modelTest: test every move and if any player wins or finally draw. 文本

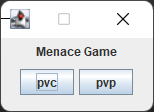
描述已自动生成

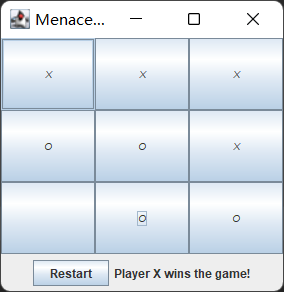
randomHumanTest: test whether the human strategy works.

文本

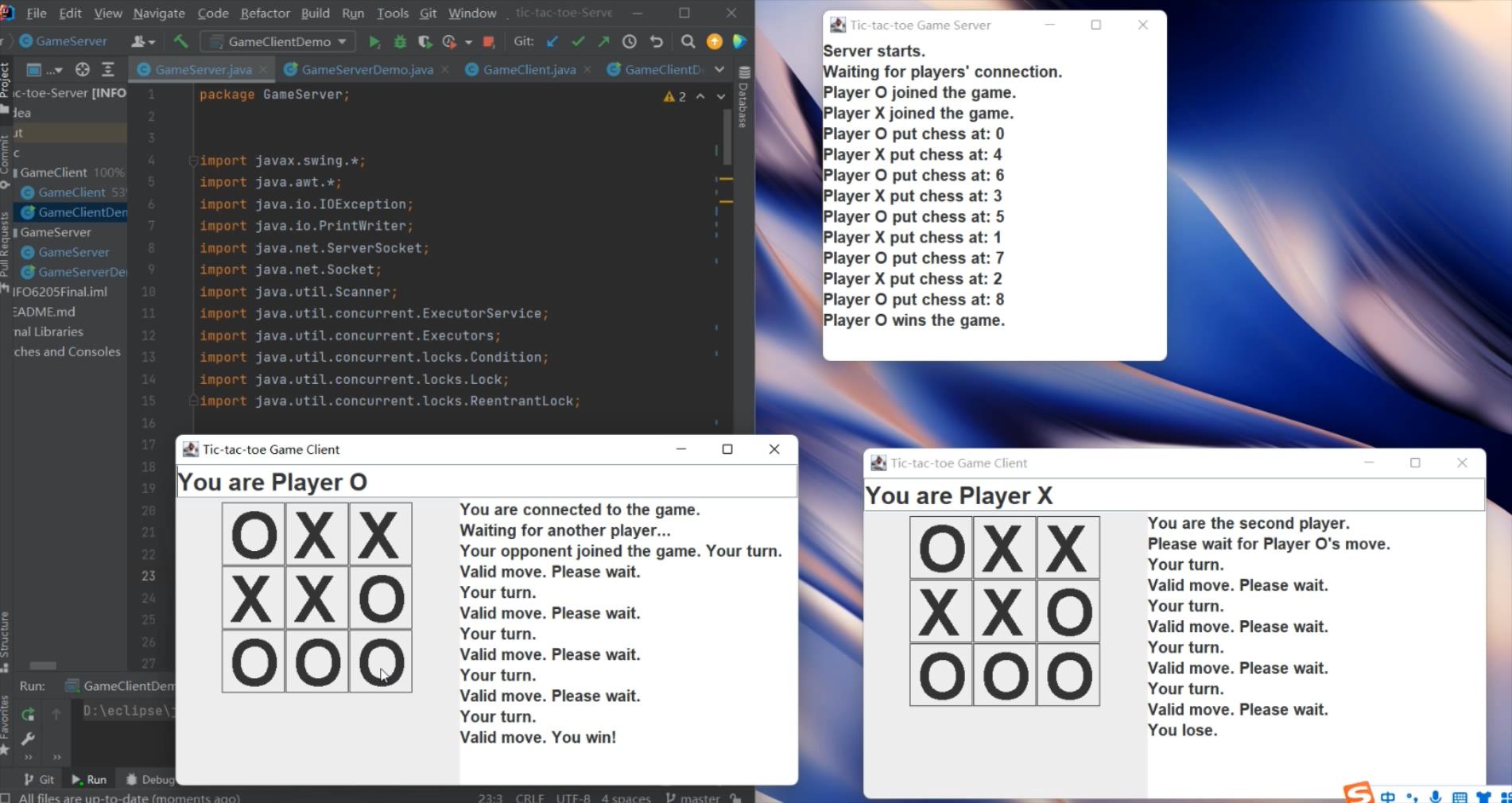
描述已自动生成

**UI of the Menace:**





**UI of the GameServer and GameClient:**



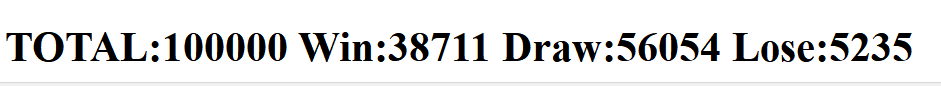
**Conclusion**:

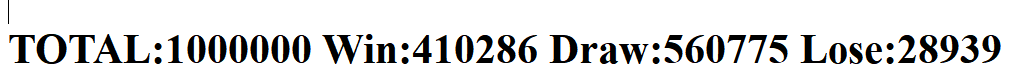
With the increase of the train times, the Menace becomes more ‘clever’, which means the probability of lose will be lower.

We can view pictures below, when the total training times equal to 1,000, Menace’s winning percentage equals to 27.8%. When the total training times equal to 100,000, Menace’s winning percentage equals to 38.7%. When the total training times equal to 1,000,000, Menace’s winning percentage equals to 41.0%.

画着卡通图案

中度可信度描述已自动生成





**References**:

Tic-tac-toe. From Wikipedia, the free encyclopedia.

Learn How to Build a Multiplayer Tic Tac Toe. Maximilian Kürschner, Programonaut.

AI based Tic Tac Toe (Java code). codeproject.