# AI Project
# Meichu 2.0

1. Project Objective

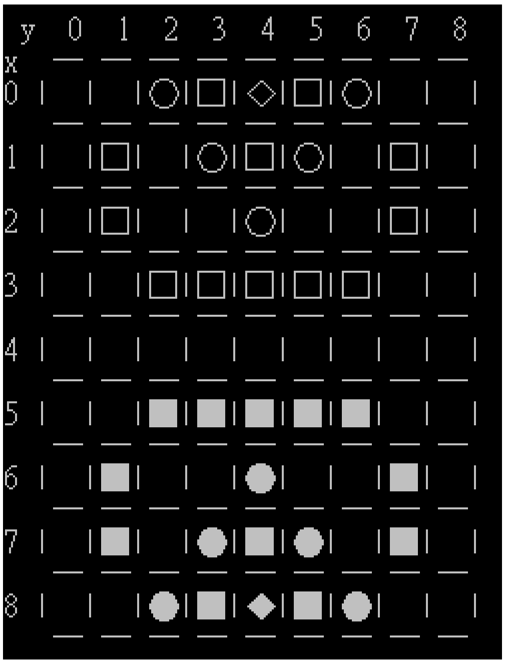
   With what you learned in class, design a AI program to play the game MeiChu well.

2. Programming Language

   C/C++ or Python. Please contact TA if you would like to use the other language.

3. Game Rules

   I. The 2-player game is played on a 9x9 checkerboard. Every player has 1 flag, 5 pieces (soldiers), 12 barriers and 150 budgets, all of which are initially placed as the diagram shown below.



| y<br>x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| 0 |  |  | ○ | □ | ◇ | □ | ○ |  |  |
| 1 |  | □ |  | ○ | □ | ○ |  | □ |  |
| 2 |  | □ |  |  | ○ |  |  | □ |  |
| 3 |  |  | □ | □ | □ | □ | □ |  |  |
| 4 |  |  |  |  |  |  |  |  |  |
| 5 |  |  | ■ | ■ | ■ | ■ | ■ |  |  |
| 6 |  | ■ |  |  | ● |  |  | ■ |  |
| 7 |  | ■ |  | ● | ■ | ● |  | ■ |  |
| 8 |  |  | ● | ■ | ◆ | ■ | ● |  |  |

The upper side is WHITE pieces.
The lower side is BLACK pieces.

Flags are shown in diamonds.
Soldiers are shown in circles.
Barriers are shown in squares.

   II. BLACK go first. At the very beginning, both of the players bid for the color to play by means of Vickrey auction. Both player have only one chance to tell which color they are willing to play with and how many budgets to bid for the choice. Then there will be 3 circumstances:

   - Both players bid for the different colors. Then both of them can take their color for free.
   - Both players bid for the same color but with different budgets. Then the player with more budgets win the bid. But they only need to pay the opponent’s bid plus one.

- Both players bid for the same color and the same budgets. Then TA will assign the color randomly. However, the one who gets what they need still has to pay the bid plus one.

III. When it’s your turn, you can choose one of your soldiers to move. The way it moves is like rooks in Chess, which move along a straight line and cannot leap over any pieces/barriers/flags. You should pay the budgets to move a soldier.

- Every move costs 1 base budget.
- You should pay for the number of squares you go through.

  For example, you go through 5 empty squares.

  You should pay 1+5 budgets.
- You should pay 3 extra budgets to break an opponent barrier.
- You can gain 3 budgets as you break your own barrier.

IV. The time limit for each turn is 30 seconds. There will be penalty as you spend more time. However, your program will be considered runtime error if it spends too long to finish the game.

V. Once any flag is occupied, the game terminates simultaneously.

VI. If one of the player exhausts their budgets with both flags surviving, the other player can make moves consecutively until they exhausts the budgets.

VII. When both players exhaust the budgets with both flags surviving, the game result will be determined as following rules in order:

i. The player having more soldiers wins.

ii. The player having more barriers wins.

iii. Draw

## 4. Submission

I. Report (student_ID.pdf) Ex. 0856705.pdf

II. Source code (student_ID.xx) Ex. 0856705.py

Please zip these files into student_ID.zip

DEADLINE: 23:55 05/01/2021

## 5. Grading Police

I. You get base score as your program is capable of finishing a game, and you are qualified to the tournament.

II. We will hold a Swiss-system tournament, and your extra score is determined by the rank you achieve in this tournament.

III.

| Penalty rules | Project Score |
|---|---|
| Time limit exceeded | -5% |
| Runtime error | -5% |
| Wrong file name | -10% |
| Delay per day | -10% |
| Plagiarism | -100% |

6. Due to the time limit, some of you may consider under what the environment will the code be run. Here is the environment we will run your code in the tournament. We will only run on CPU.
   - Windows 10
   - CPU: AMD R5 2600
   - Memory: 32 Gb
   - Python 3.7.6
   - `$ g++ -std=c++14 -O3 student_ID.cpp -o student_ID.exe`

7. Feel free to contact TA if you have any further questions.