Assignment 2:
The X-pattern Game

Set by CS4386 TA Team

Early Testing Deadline:

Thursday 28th 23:59 March 2024

Sunday 7th 23:59 April 2024

Final Submission Deadline:

Sunday 21th 23:59 April 2024

This assignment is worth

20 % (**twenty** percent)

of the overall course mark

Overview

1 Introduction

2 Marking scheme

3 The X-pattern Game

4 Online Judgment platform

5 Requirements

6 Q&A

1 Introduction

You are tasked with writing an AI program that can run the X-pattern game, you can implement your AI program using algorithms including but not limited to those learned in class. You can use Python, C or C++ to implement your AI program. An online AI game judgment platform is provided to submit and test your code.

2 Marking scheme

Item	Due	Mark
Beat preset AI	21th April 23:59	30
Tournament1	28th March 23:59	0
Tournament2	7th April 23:59	20
Tournament3	21th April 23:59	20
Report	21th April 23:59	30
	Total	100

Note: Your AI performance depends on:

- Whether you beat our three preset AIs. If you beat all our three preset AIs, you will get 30 points (10 points for each AI).
- Your ranking in tournaments. We will schedule three tournaments. After each tournament, you'll be able to see where your AI stands in the rankings. These rankings are an integral part of your overall assessment.

In addition to the score from your AI performance, **the quality of your algorithm and** report will also affect the final score. Take some time to make it look good.

3 The X-pattern Game

The X-Pattern Game is a strategic board game where two artificial intelligence (AI) players compete to form 'X' patterns on a 10x10 grid. Each player takes turns to place their pieces, aiming to create as many 'X' patterns as possible to get higher points. The game ends when the grid is filled(no more than 100 moves) and the player who creates more X-pattern on the grid will be the winner. You are encouraged to come up with your own strategies to make your AI make better decisions in each turn. We are excited to see your brilliant ideas and how they challenge each other !

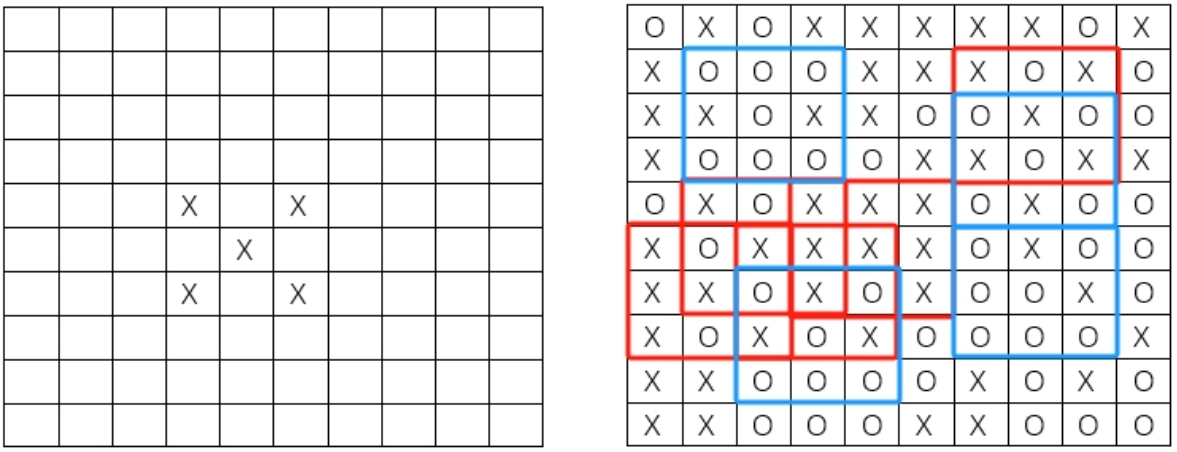


Figure 1. Score shape examples

When the game ends, we will calculate your score by counting the number of **full** 'X' patterns in the grid. The player with the higher score will be the winner, and the final score and game log will be provided. For example, in figure1, player O gains 4 points and player X gains 5 points, so player X is the winner.

4 Online Judgment platform

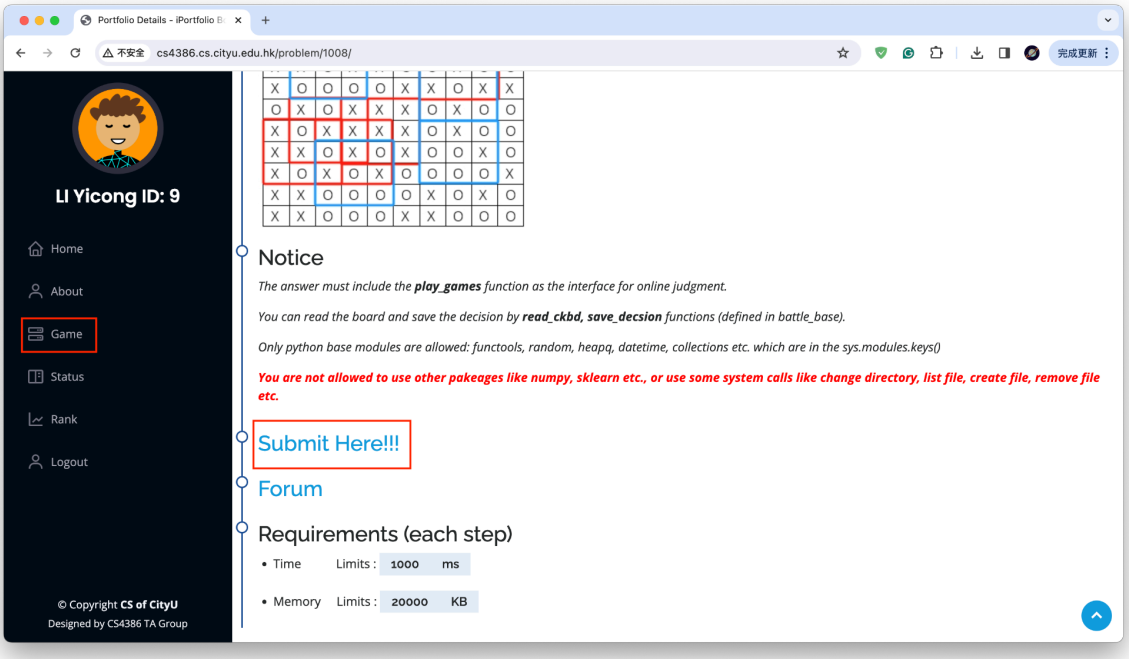
Website URL: http://cs4386.cs.cityu.edu.hk/user/login/
This URL can **only** be accessed within the cityu campus network. If you want to use this website outside the campus, please use cityu vpn first. The platform supports Python, C and C++ as your program language.

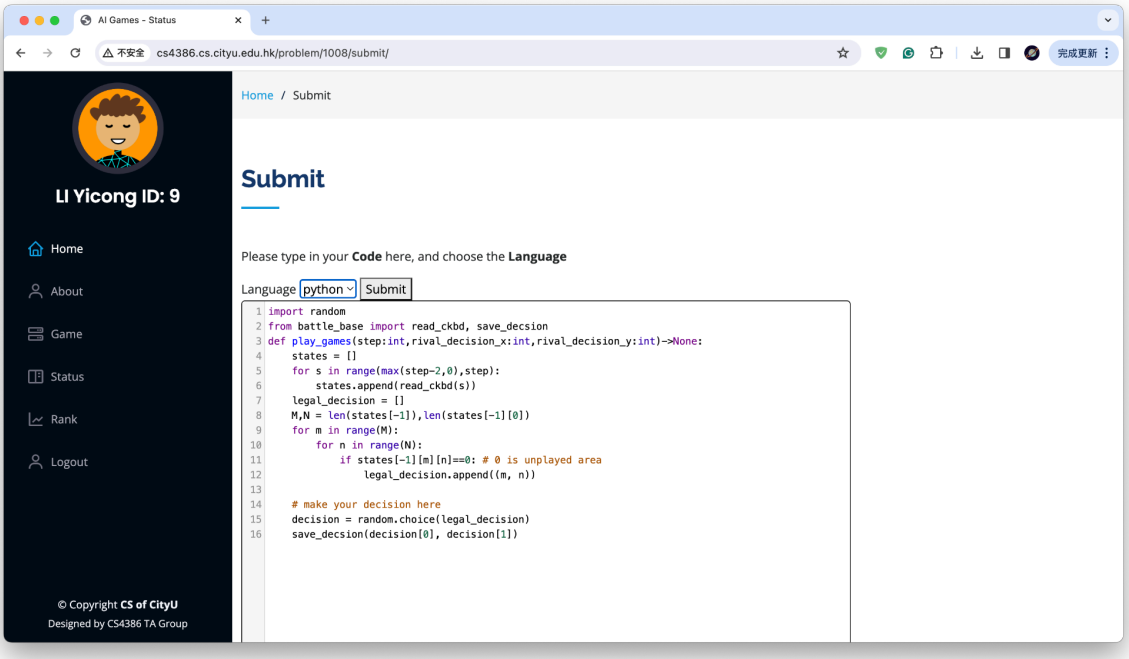
Register and log in

- You should register using your Cityu email.
- You should register using your real name.
- Upon registration, you will be arranged a useID. Only your ID will be showed in the ranking.

Submit your code: play against preset AI

Click on "Game" in the left column, then you can see the X-pattern game. Click "X-pattern" game then click "Submit Here" to go to the submission page. Fill in your code and click "Submit" to submit. You can choose a language you are familiar with.


Portfolio Details - iPortfolio
cs4386.cs.cityu.edu.hk/problem/1008/
LI Yicong ID: 9
Home
About
Game
Status
Rank
Logout
© Copyright CS of CityU
Designed by CS4386 TA Group
Notice
The answer must include the play_games function as the interface for online judgment.
You can read the board and save the decision by read_ckbd, save_decsion functions (defined in battle_base).
Only python base modules are allowed: functools, random, heapq, datetime, collections etc. which are in the sys.modules.keys()
You are not allowed to use other pakeages like numpy, sklearn etc., or use some system calls like change directory, list file, create file, remove file etc.
Submit Here!!!
Forum
Requirements (each step)
Time Limits : 1000 ms
Memory Limits : 20000 KB


AI Games - Status
cs4386.cs.cityu.edu.hk/problem/1008/submit/
LI Yicong ID: 9
Home
About
Game
Status
Rank
Logout
© Copyright CS of CityU
Designed by CS4386 TA Group
Home / Submit
Submit
Please type in your Code here, and choose the Language
Language python Submit
import random
from battle_base import read_ckbd, save_decsion
def play_games(step:int,rival_decision_x:int,rival_decision_y:int)->None:
states = []
for s in range(max(step-2,0),step):
states.append(read_ckbd(s))
legal_decision = []
M,N = len(states[-1]),len(states[-1][0])
for m in range(M):
for n in range(N):
if states[-1][m][n]==0: # 0 is unplayed area
legal_decision.append((m, n))
# make your decision here
decision = random.choice(legal_decision)
save_decsion(decision[0], decision[1])

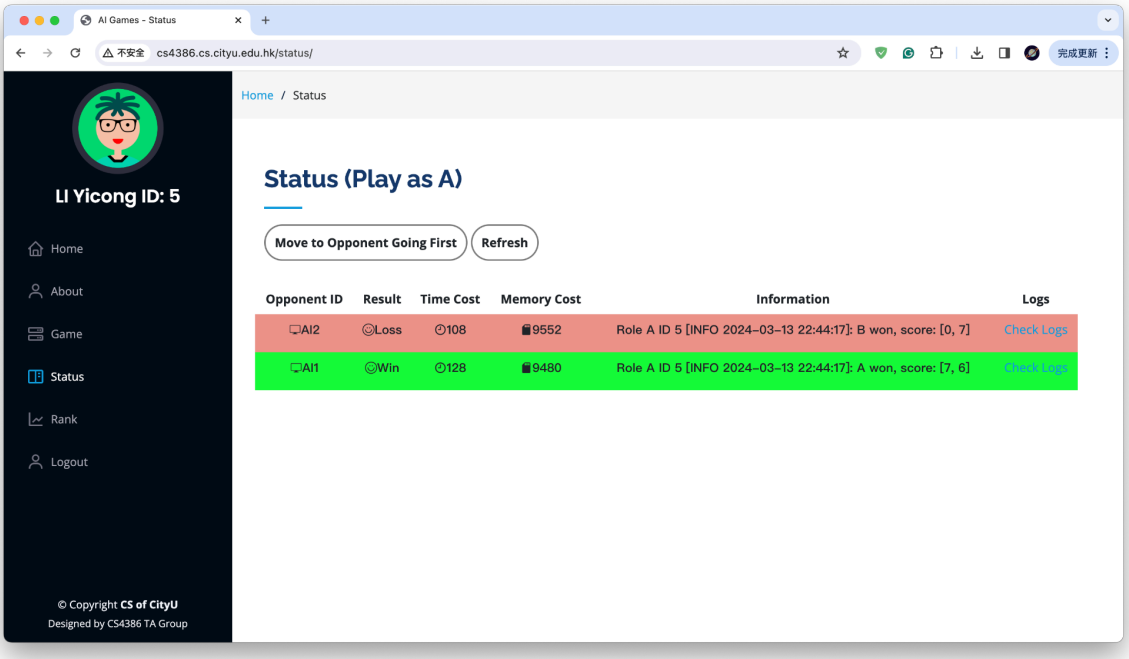


Figure2. Submit your code and get result

We've set up three predetermined AI opponents for you to challenge: "Random," "Defensive," and "Offensive," designated by user IDs 1, 2, and 3, respectively. Upon submitting your code, it will automatically be matched against the Random (AI1) and Defensive (AI2). You'll need to periodically refresh the page to view the outcomes of these matches.

To ensure fairness, considering the advantage or disadvantage of starting first, we'll conduct two matches for each AI pairing. This includes one game where your AI going first, and another where it plays second(opponent going first). The results of these matches, along with the time and memory usage statistics, will be updated following each submission.

To earn the "Beat preset AI" score, your most recent AI submission must outperform our preset AIs in both match configurations—once your AI going first and once opponent going first. Please note, the matches against the "Offensive AI" will not be visible during your testing phase.

Check log:
You can download the log to view the state of the chessboard at each step after getting the results.

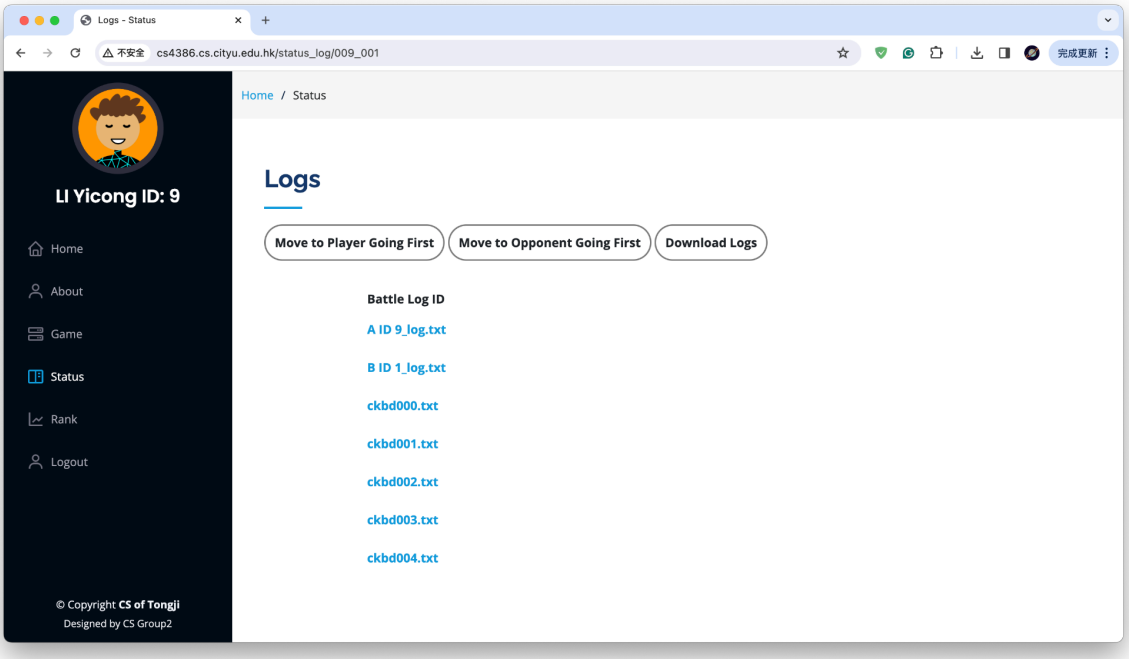


Figure 3. For each submission, you can click the download logs file(format: userid1_userid2 , userid1 is always the player going first). You can also view the logs directly on the website.

View ranking
The system will save the most recent version of your code submitted prior to each tournament deadline. The tournament itself takes place the day following the deadline. On this "competition day," the server will be offline. Your AI's performance will be evaluated based on its win rate across the matches(If there is a draw it will not count as a win).

In instances where two participants' AIs have identical win rates, the AI that used less time to achieve its wins will receive a higher ranking. While you're free to check the rankings at any moment, it's important to remember that these interim rankings only compare your AI against our preset AI and not against the entire field of competitors. Therefore, these preliminary standings do not represent the final rankings for the tournament. The definitive rankings will be announced once all matches are concluded and thoroughly evaluated.

5 Requirement

Format of the code:

- The answer must include the ***play_games*** function as the interface for online judgment, which is defined as,
  C/C++: void play_games(int step, int rival_decision1,int rival_decision2);
  Python: play_games(step:int,rival_decision_x:int,rival_decision_y:int)->None

    - *step* indicates the move number in the current game. It starts from Player A, step=1, then turns to player B, step=2, and then turns to player A, step=3, ..., end of the game.
    - That is, player A's step is always odd, player B's step is even. Step=0 represents an empty checkerboard for initialization.
    - *rival_decision_x,y* are the opponent's decisions made in the previous step, which can help you make better responses.

- You can read the board and save the decision by ***read_ckbd***, ***save_decsion*** functions. MAX_M, MAX_N are the size of the board, (equal 10 in this game). They are defined in **battle_base**.
    - C/C++: void read_ckbd(int step, int array[MAX_M][MAX_N]);
      void save_decision(int x, int y);
    - Python: read_ckbd(step:int)->List[List[int]]
      save_decision(x:int, y:int)->None
    - In the array/list returned by ***read_ckbd***, 0 represents the unplayed area and you can make a decision here. 1 represents player A's decisions, 2 represents player B's decisions.
    - Note that you must save your decision by calling ***save_decision*** in ***play_games*** function, don't return anything. **Do not** try to write those functions, you just call them. You can read any step of chess before the current step.
- The ***main*** function is not allowed in the answer (especially for C/C++).
- You can use print/printf/cout to print logs to help you debugging. They will be printed in the battle_log.txt.
- The details of the ***read_ckbd***, ***save_decsion*** functions are in the appendix. Just for helping you run the battle on your computer. Please do not include them in your answer.

Submission requirements:

- For Python users: python base modules (in sys.modules.keys()): functools, random, sorted, heapq, collections etc. are allowed. You are **not allowed** to use other packages like numpy, sklearn etc.
- For C/C++ users: most header files are allowed, even including <bits/stdc++.h>, <iostream> and <algorithm>.
- You can not use some system calls like change directory, list file, create file, remove file etc.
- For all AI programs, we limit the total running time and the total memory usage, you can find it on the website.
- Violation of any of the above will raise an error and let the game be lost.

Requirement for the report:

You should write a report to explain your AI.
Describe your algorithm as clearly as possible. Feel free to use examples and add screenshots or other figures if it can help better illustrate your method. If you adopt some part of your code from somewhere, you **must** fully acknowledge this and provide a reference to where you obtain the code. You **must** declare how much of your submitted code is obtained from someone/somewhere else and how much is indeed written by you.
At the end of your report, include the related references from where you have gathered useful information in working on your assignment.
You need to submit your source code to the online judgment system and submit your report to Canvas.
Name your files according to the following format, where XXXXXXXX is the 8 digits of your student ID number:
CS4386-2324B-A2-XXXXXXXX-Report.pdf (should be in PDF format)

6 Q&A

For any questions, please write an email to: yicongli4-c@my.cityu.edu.hk, We will also create a discussion on Canvas. You can ask questions there and the TAs will check it everyday and answer your questions as soon as possible.

Appendix: Sample submission code

Python:

```
import random
from battle_base import read_ckbd, save_decsion
def
play_games(step:int,rival_decision_x:int,rival_decision_y:int)->No
ne:
    states = []
    for s in range(max(step-2,0),step):
        states.append(read_ckbd(s))
    legal_decision = []
    M,N = len(states[-1]),len(states[-1][0])
    for m in range(M):
        for n in range(N):
            if states[-1][m][n]==0: # 0 is unplayed area
                legal_decision.append((m, n))

    # make your decision here
    decision = random.choice(legal_decision)
    save_decsion(decision[0], decision[1])
```

C:

```c
#include <stdio.h>
#include <stdlib.h>
#include "battle_base.h"
#define max(a,b) ((a) >= (b) ? (a) : (b))
void play_games(int step, int rival_decision_x,int
rival_decision_y) {
    int states[2][MAX_M][MAX_N]; // Assuming MAX_M and MAX_N are
defined board size
    int states_count = 0;
    for (int s = max(step - 2, 0); s < step; s++) {
        read_ckbd(s, states[states_count]);
        states_count++;
    }

    int legal_decision[MAX_M*MAX_N][2];
    int legal_count = 0;
    for (int m = 0; m < MAX_M; m++) {
        for (int n = 0; n < MAX_N; n++) {
            if (states[states_count - 1][m][n] == 0) // 0 is
unplayed area
            {
               {
                    legal_decision[legal_count][0] = m;
                    legal_decision[legal_count][1] = n;
                    legal_count++;
                }
            }
        }
    }

    // Make your decision here
    int random_index = rand() % legal_count;
    int* decision = legal_decision[random_index]; // Example of
random strategy

    save_decision(decision[0], decision[1]);
}
```

C++:

```
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include "battle_base.h"
void play_games(int step, int rival_decision1,int rival_decision2)
{
    int states[2][MAX_M][MAX_N]; // Assuming MAX_M and MAX_N are
defined board size
    int states_count = 0;
    for (int s = std::max(step - 2, 0); s < step; s++) {
        read_ckbd(s, states[states_count]);
        states_count++;
    }

    int legal_decision[MAX_M*MAX_N][2];
    int legal_count = 0;
    for (int m = 0; m < MAX_M; m++) {
        for (int n = 0; n < MAX_N; n++) {
            if (states[states_count - 1][m][n] == 0) // 0 is
unplayed area
            {
                {
                    legal_decision[legal_count][0] = m;
                    legal_decision[legal_count][1] = n;
                    legal_count++;
                }
            }
        }
    }

    // Make your decision here
    int random_index = rand() % legal_count;
    int* decision = legal_decision[random_index]; // Example of
random strategy

    save_decision(decision[0], decision[1]);
}
```

The following code is for reference only, do not include them in your answer.

Battle_base.py

```
import os
DESCISION_TMP = os.path.join('decision.txt')
BATTLE_LOG = 'logs'
MAX_M = 10
MAX_N = 10
def read_ckbd(step):
    try:
        array =[]
        txt_path = '{}/ckbd{:0>3d}.txt'.format(BATTLE_LOG,step)
        with open(txt_path,'r') as f:
            data_lists = f.readlines()
        for lines in data_lists:
            array.append([int(x) for x in lines.strip().split('
')])
        return array
    except:
        raise IndexError('Invalid read')

def save_decsion(a,b):
    with open(DESCISION_TMP,'w') as f:
        f.write('{:d} {:d}'.format(a,b))
```

Battle_base.h

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_M 10
#define MAX_N 10
#define BATTLE_LOG "logs"
#define DESCISION_TMP "decision.txt"

#ifndef __BATTLE_h__
#define __BATTLE_h__

inline int read_ckbd(int step, int array[MAX_M][MAX_N]);
inline void save_decision(int a, int b);
void play_games(int step, int rival_decision1,int
rival_decision2);
int read_ckbd(int step, int array[MAX_M][MAX_N]) {
    char txt_path[100];
    sprintf(txt_path, "%s/ckbd%03d.txt", BATTLE_LOG, step);

    FILE *f = fopen(txt_path, "r");
    if (f == NULL) {
        printf("Error opening file\n");
        return 1;
    }
    int array_index = 0;
    char line[100];

    while (fgets(line, sizeof(line), f)) {
        char *token = strtok(line, " ");
        int index = 0;
        while (token != NULL) {
            array[array_index][index] = atoi(token);
            token = strtok(NULL, " ");
            index++;
        }
        array_index++;
    }

    fclose(f);
```

```
    return 0;
}

void save_decision(int a, int b) {
    FILE *f = fopen(DESCISION_TMP, "w");
    if (f == NULL) {
        printf("Error opening file\n");
        return;
    }
    fprintf(f, "%d %d", a, b);
    fclose(f);
}
#endif
```