**VIETNAM NATIONAL UNIVERSITY**

**UNIVERSITY OF SCIENCE**



**TOPIC**

**Project 01**

MEMBERS

Lê Thanh Bình 18127041

Lư Ngọc Liên 18127046

Nguyễn Thị Anh Đào 18127272

Vũ Công Thành 18127218

**Course: Artificial Intelligence**

**Ho Chi Minh City– 2020**

# Contents

[Contents 4](#_Toc48389192)

[Assignment Plan 5](#_Toc48389193)

[Environment to compile and run the program 7](#_Toc48389194)

[Estimating the degree of completion level for each requirement 8](#_Toc48389195)

[1. Level 1 and level 2 8](#_Toc48389196)

[2. Level 3 10](#_Toc48389197)

[3. Level 4 10](#_Toc48389198)

[4. Graphical demonstration 10](#_Toc48389199)

[5. Report 11](#_Toc48389200)

[6. Data 11](#_Toc48389201)

[Instruction 13](#_Toc48389202)

# Assignment Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tasks | Lê Thanh Bình | Lư Ngọc Liên | Nguyễn Thị Anh Đào | Vũ Công Thành |
| Create text files | No | No | 5 maps for each levels (level 3 and level 4)  5% | 5 maps for each levels (level 1 and level 2)  5% |
| Read files | No | Function to read maze from file  1% | No | No |
| Graphic | Draw map (monster, food and pacman) from file.  5% | Pacman move follow path 4% | No | No |
| Level 1 and level 2 | No | No | A\* function to find the path for pacman 15% | BFS function to find the path for pacman  15% |
| Level 3 | Monsters move randomly  5% | No | Find path for pacman  10% | No |
| Level 4 | Monster change path when pacman move.  1% | Using A\* to find path for Monsters  4% |  |
| Report | Write report 15% for each member | | No | No |

***Self rating:***

Project: 100% overall.

For each member:

+ Lê Thanh Bình: 26%

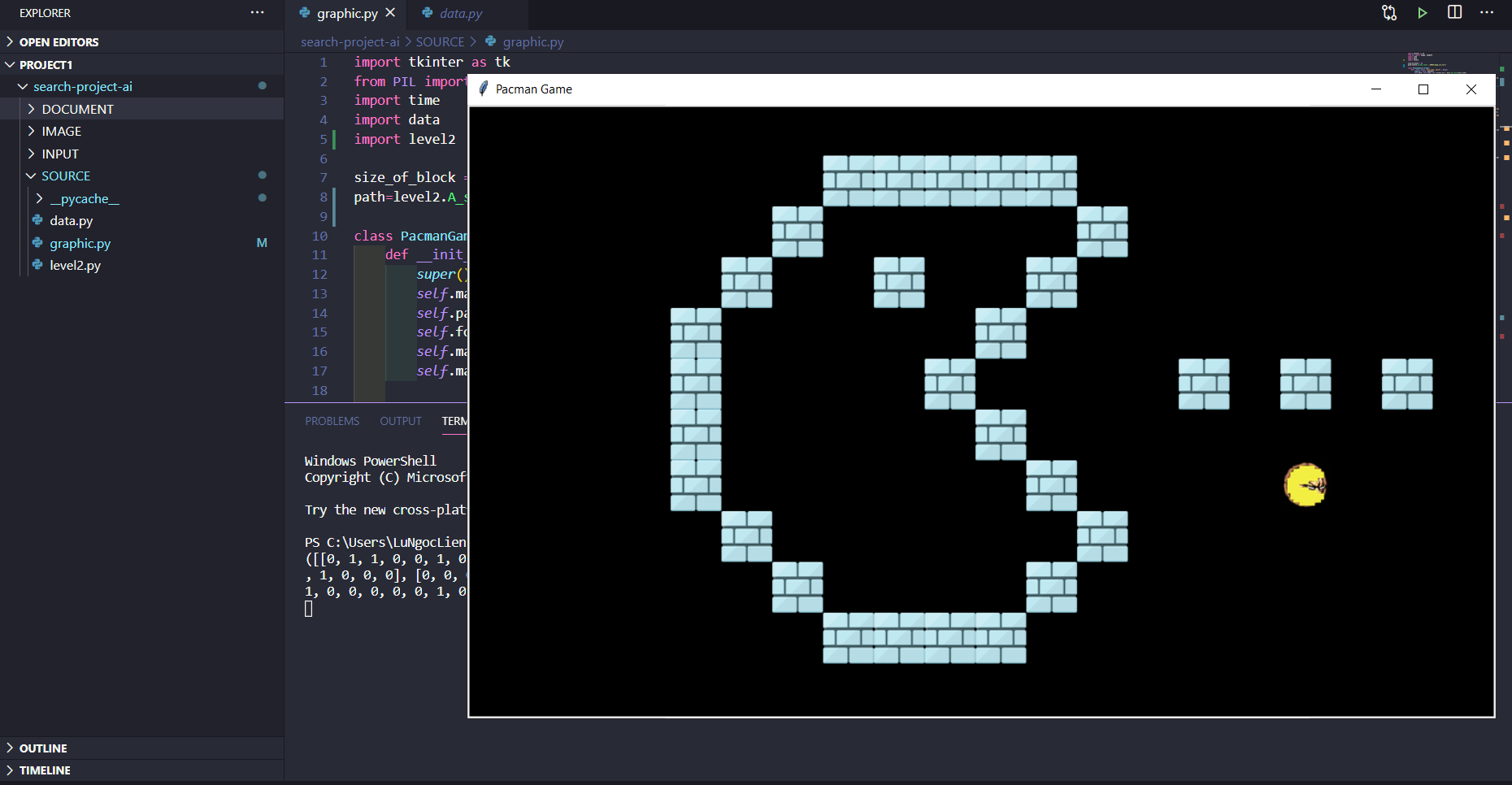
+ Lư Ngọc Liên: 24%

+ Nguyễn Thị Anh Đào: 30%

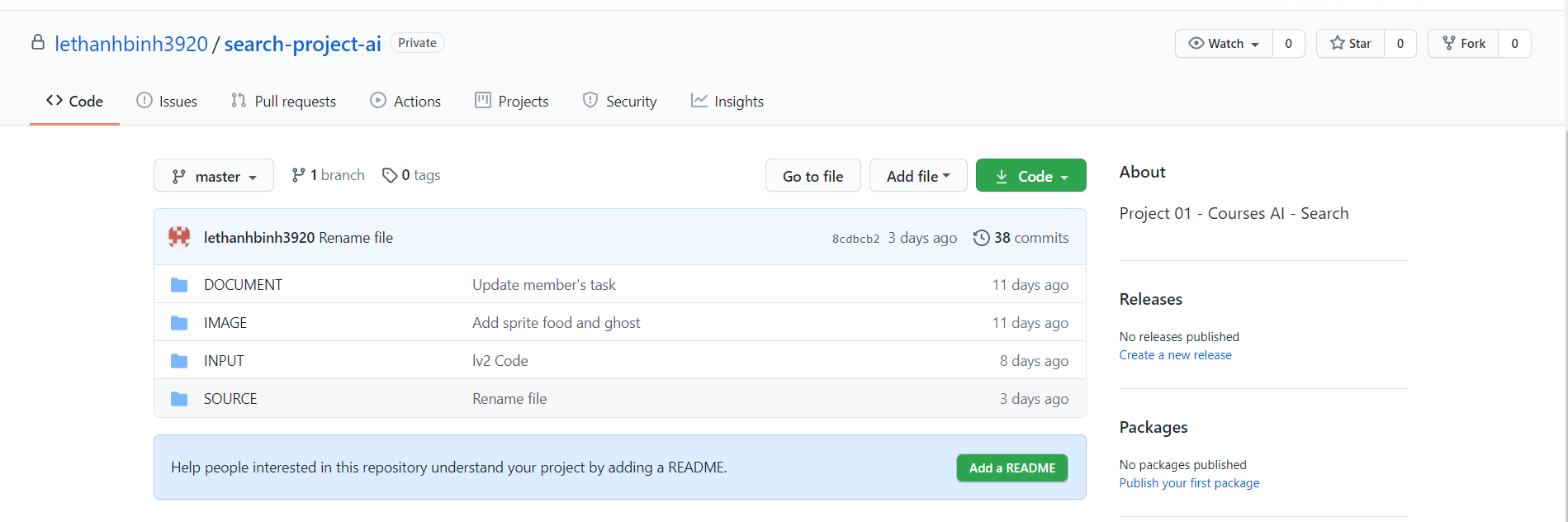
+ Vũ Công Thành: 20%

# Environment to compile and run the program

Environment to compile and run the program: Visual Code



Version control: GitHub

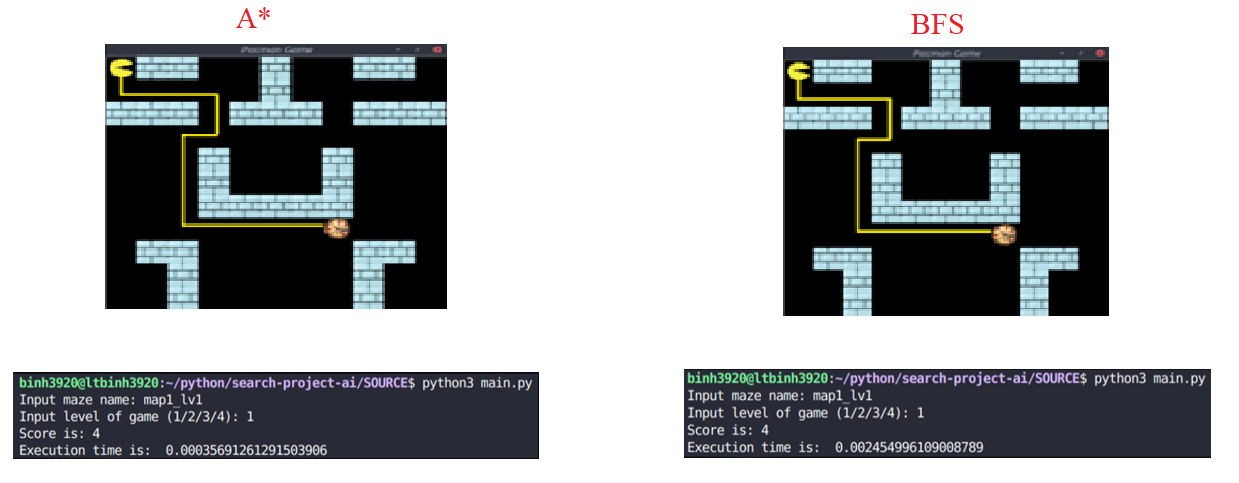
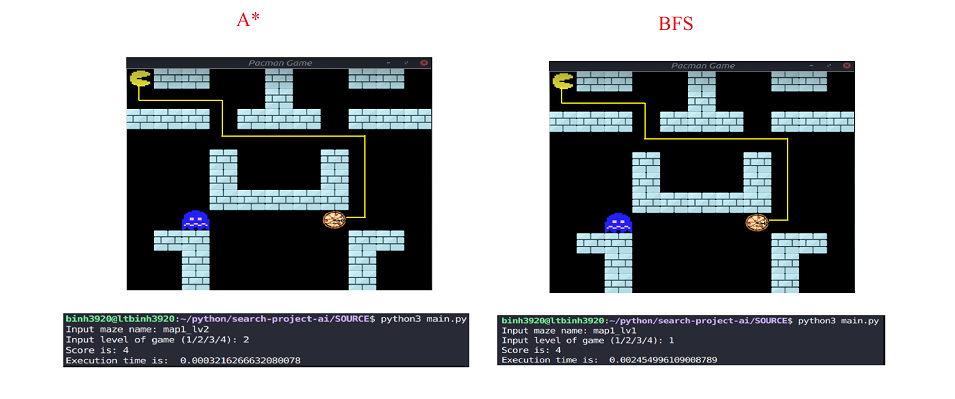
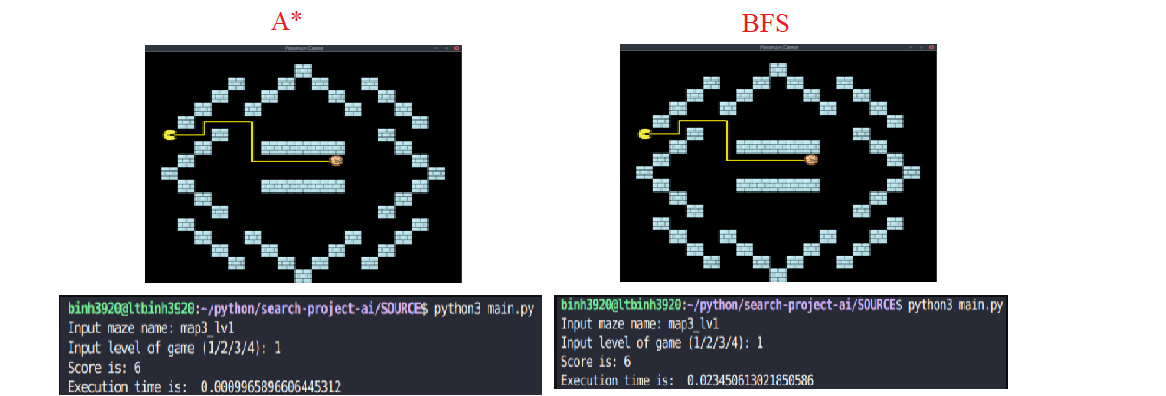


# Estimating the degree of completion level for each requirement

## Level 1 and level 2

The idea of level 1 and level 2 is the same because at level 2 the monster does not move, so we consider the monster as a wall. The mission is pacman finds the food.

We use two searching algorithms:

* A\* search
* BFS.
* MAP 1
* LEVEL 1:
* LEVEL 2:
* MAP 2
* LEVEL 1:
* LEVEL 2:



Summary:

In runtime implement, A\* is better than BFS. But in some special situations (level 2 map 2) A\* takes long time to get the path than BFS.

## Level 3

The idea:

* Pacman: In area 7x7, if the pacman sees the food, moves directly to it and eats by using A\*. Based on heuristic (move follow the step which has the smallest heuristic between pacman position and food position). However, in some situations, the pacman will stop to wait for monster’s move to get a better heuristic. If not, pacman moves randomly to the next step. This action repeats until there is no more food in maze or pacman touch the monster.
* Monster: when pacman moves, the monster moves randomly around the its position (up, down, left, right) until pacman eats all food in maze or catches pacman.

## Level 4

The idea:

* Pacman is the same as level 3.
* Monster: monsters move directly to the pacman position by using A\* heuristic between pacman position and its position until the pacman eats all food in maze or catches pacman. Each monster will wait for a different time before it scans for pacman again.

## Graphical demonstration

We use graphical library included:

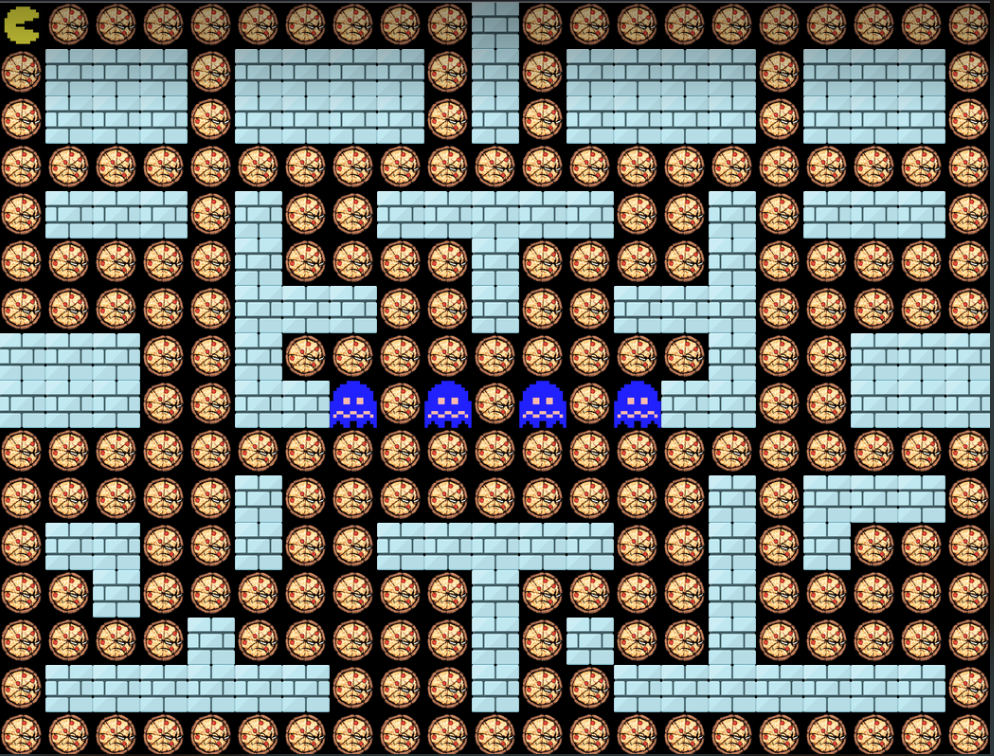
+ Canvas of Tkinter

+ Image of PIL

Images:

|  |  |  |  |
| --- | --- | --- | --- |
| Food | Monster | Pacman | Block |
|  |  |  |  |

Some example maps:



## Report

We use Microsoft Word to write down what, how we have done the project by using text, image to illustrate.

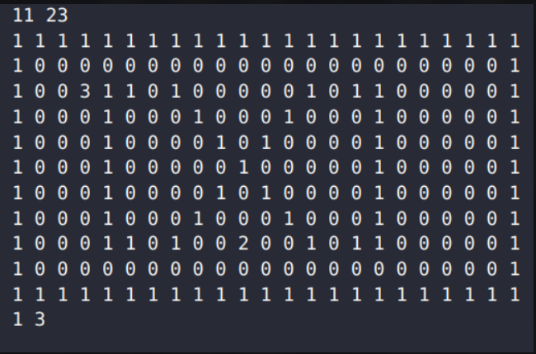
## Data

We have 5 different maps for level 1 and level 2. These are quite the same where we put the food, block and pacman. The different is in level 2, the map has one monster.

In level 3 and level 4, we have 5 completely different maps.

The first line is the size of maze, the last line is the start position of pacman, number 1 is block, number 2 is food and number 3 is monster

Some example maps:



# Instruction

To play the game, you should follow steps below.

1. Run the main.py file.
2. Input maze you want to play by entering the syntax *“map[1]\_lv[2]”*

* [1] is the maze number (from 1 to 5)
* [2] is the level number (from 1 to 4)

1. Input the level you want to play by entering the number from 1 to 4
2. Choose the movement speed of pacman. The higher number, the slower pacman moves. For example: *250* means 250 milliseconds.
3. Enjoy the game.

During your time game, the score will calculate follow some rules:

1. Each step pacman moves, your score decrease 1 point.
2. When you get food, your score increase 20 point.
3. You can see the score on the right side on the game screen.
4. Console will print real time execution. This is time for algorithms return path for pacman.