**VIETNAM NATIONAL UNIVERSITY – HO CHI MINH CITY**

**INTERNATIONAL UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**Ảnh có chứa vòng tròn, biểu tượng, Nhãn hiệu, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.**

**ARTIFICIAL INTELLIGENCE**

**IT159IU**

**REPORT LAB 1**

**Instructor:**

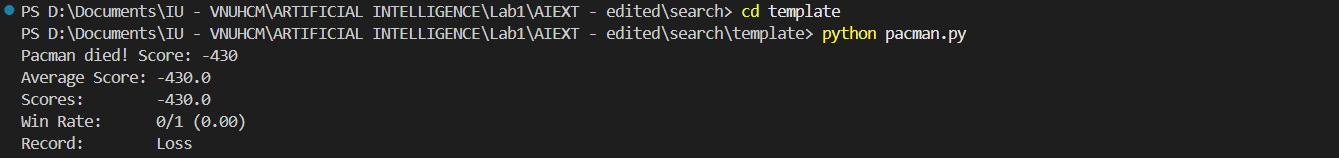
**Dr. Nguyen Trung Ky**

**Dr. Ly Tu Nga**

**Nguyen Huynh Ngan Anh - ITDSIU23003**

1. **Setup:**

Tried running the below command to run the game



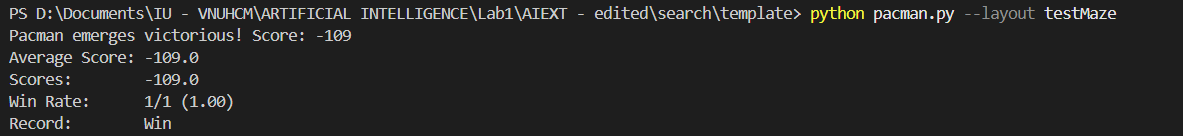
The interface of the game

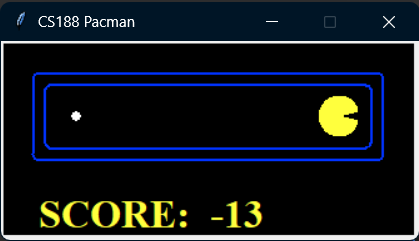
Ảnh có chứa ảnh chụp màn hình, Phần mềm đa phương tiện, văn bản, phần mềm

Nội dung do AI tạo ra có thể không chính xác.

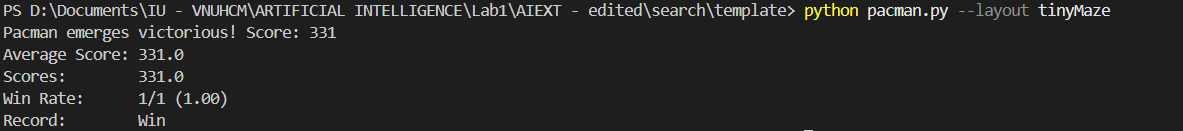
Tried running different layout of the maze to view their interface:

* testMaze:





* tinyMaze:



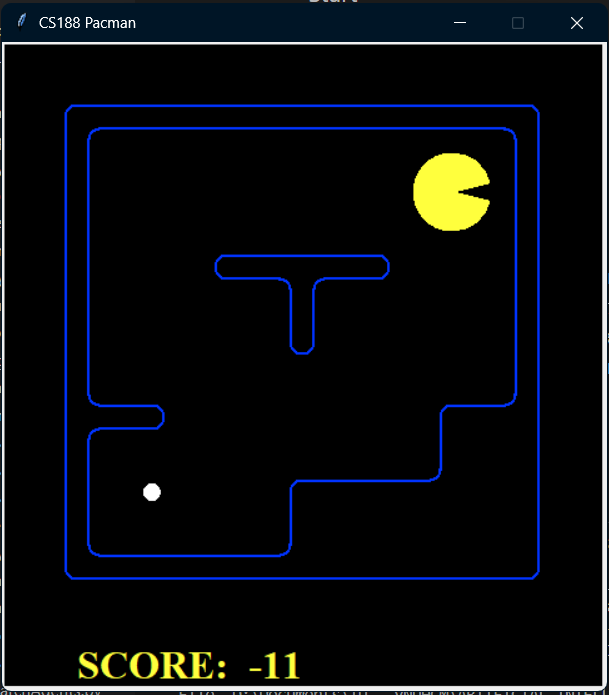
Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, biểu đồ

Nội dung do AI tạo ra có thể không chính xác.

* tinyMaze that zoom x2:

Ảnh có chứa văn bản, Phần mềm đa phương tiện, phần mềm, ảnh chụp màn hình

Nội dung do AI tạo ra có thể không chính xác.

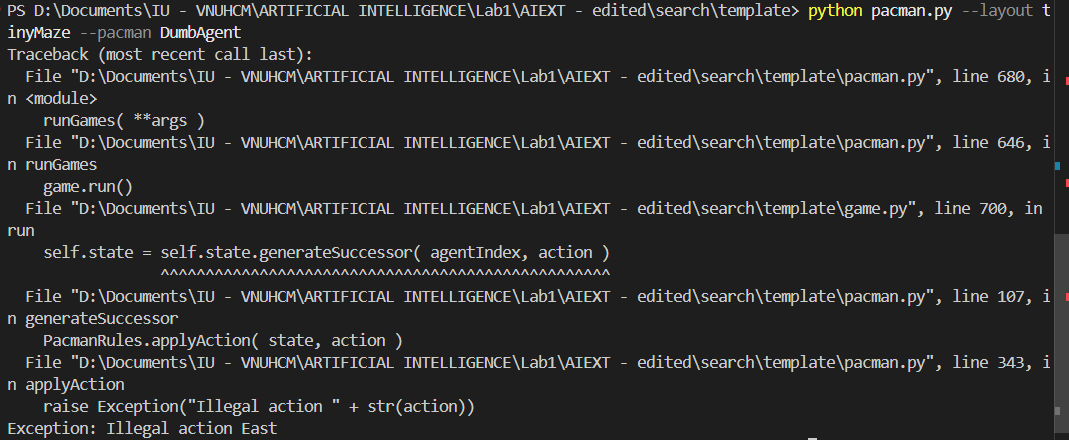


Tried creating DumbAgent and running it in tinyMaze to test pacman’s move

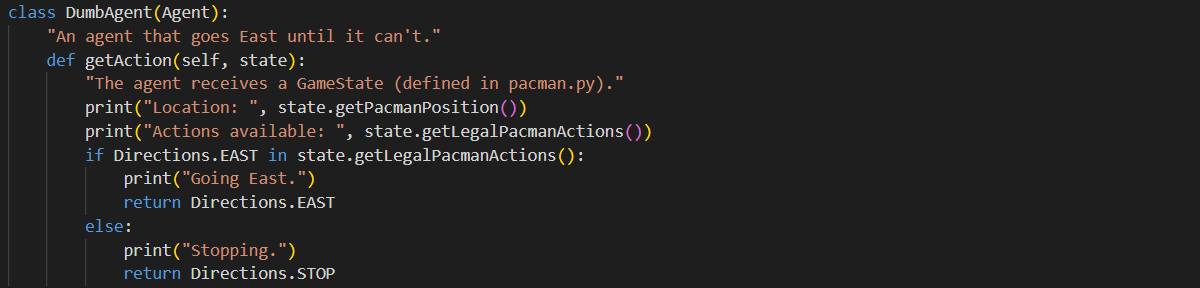
Ảnh có chứa văn bản, Phần mềm đa phương tiện, Phông chữ, ảnh chụp màn hình

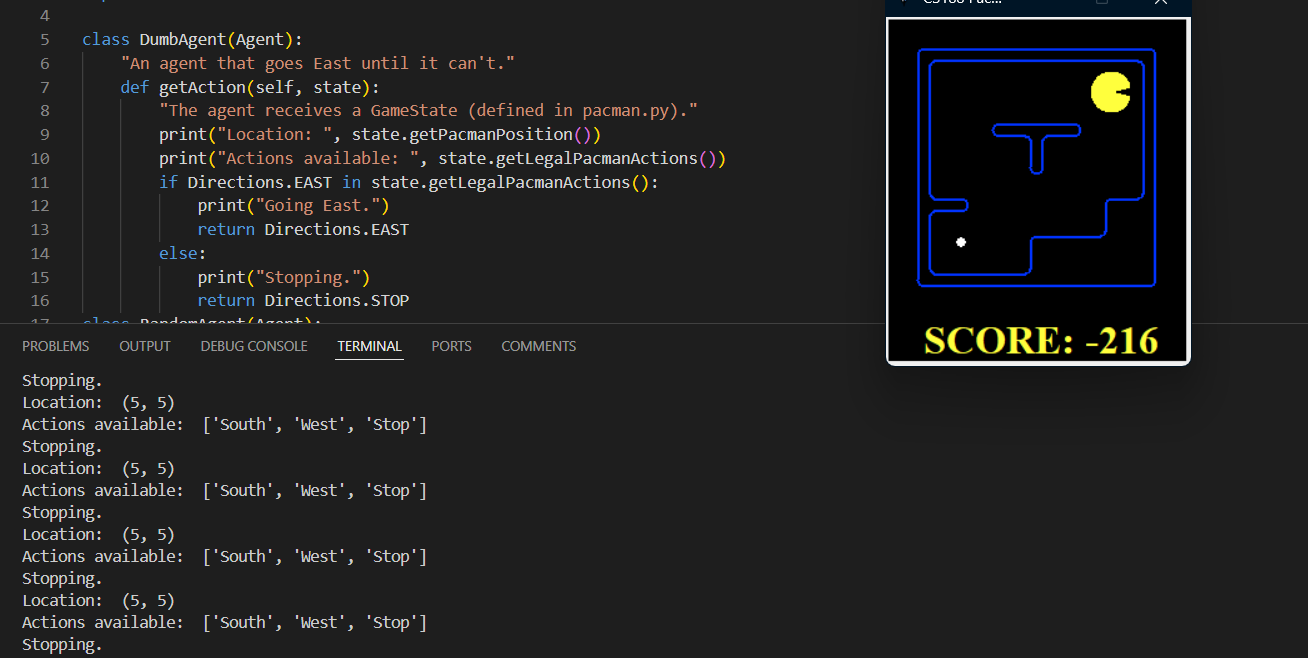
Nội dung do AI tạo ra có thể không chính xác.

The game crashed with an ‘Illegal Action East’. The error message indicated that the [DumbAgent](vscode-file://vscode-app/c:/Users/ASUS/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o ") was trying to perform an illegal action, specifically moving East when it was not allowed.



Tried modified the DumbAgent to prevent the game from crashing





1. **Exercise 1:**

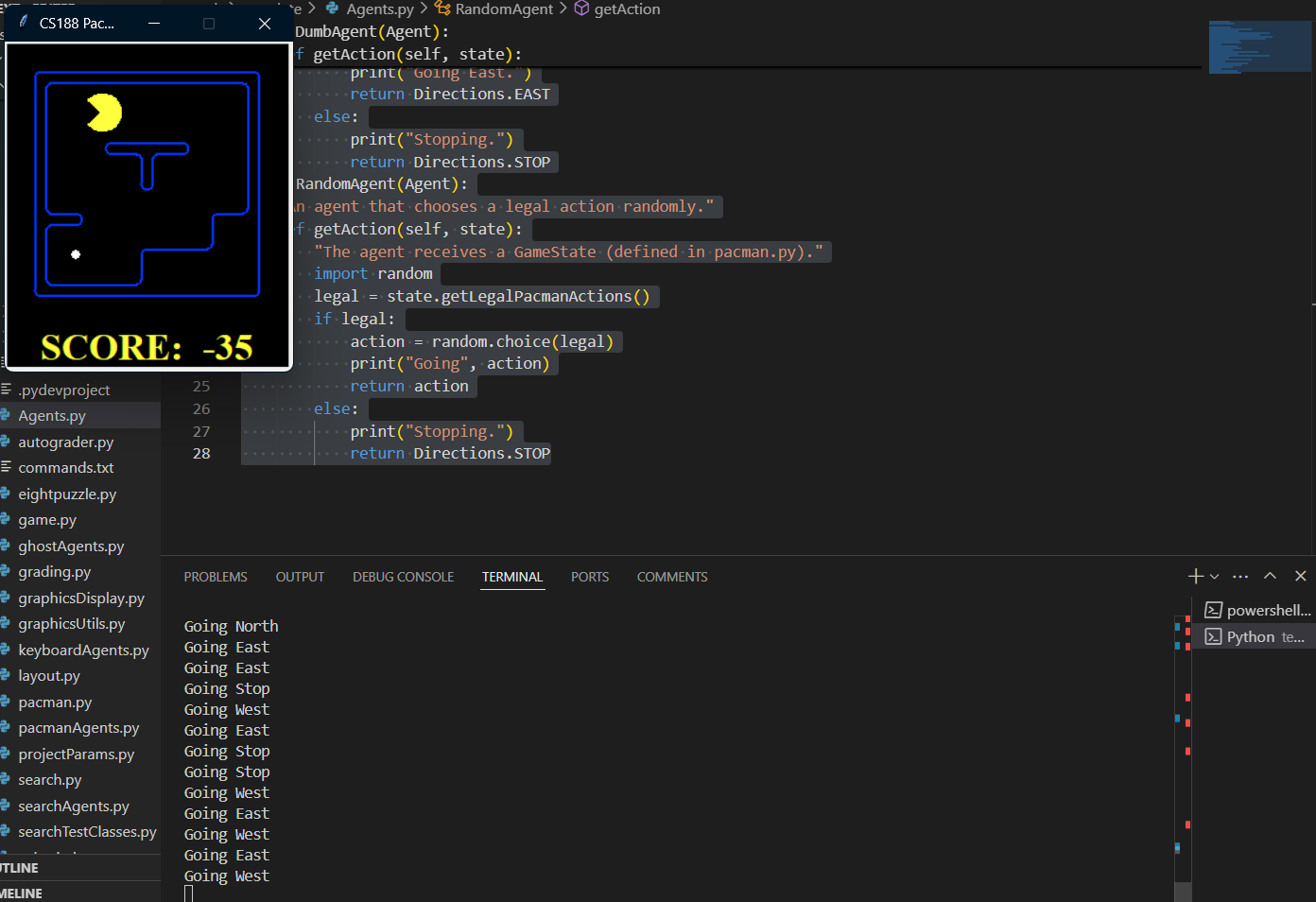
Created a RandomAgent class in Agents.py

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Nội dung do AI tạo ra có thể không chính xác.

Since it was random, it always got the food ultimately, whether it did so right away or took a while. Because it only pursued legal action, it didn't collapse. The agent moved in numerous directions in a short time; it could be looped again with the previous step or go to the new position. Perhaps, it took time for an agent to reach the food position, so the result was not quite good, and if there were enemies, there was a high probability of losing because the agent could not shirk the danger. RandomAgent moved in a random direction, including stopping. It sometimes stopped on the wall.

Running the RandomAgent in tinyMaze





Running the RandomAgent in mediumMaze

Ảnh có chứa ảnh chụp màn hình, Phần mềm đa phương tiện, văn bản, Phần mềm đồ họa

Nội dung do AI tạo ra có thể không chính xác.

1. **Exercise 2:**

Created myLayout.lay

Ảnh có chứa văn bản, ảnh chụp màn hình

Nội dung do AI tạo ra có thể không chính xác.

* For the myLayout.lay, each "%" defined the wall for the game rendering. They could have many different shapes, but the prerequisite requirement was that the map had to be wrapped by "%". According to this, the game system worked smoothly without error.
* Each "." defined the food that the agent could eat to win the game. If there was no "." inside the layout, the map described it as a space, which the agent could move across.
* The "P" defined the initial position of the agent (Pac-Man).

Running RandomAgent in myLayout.lay

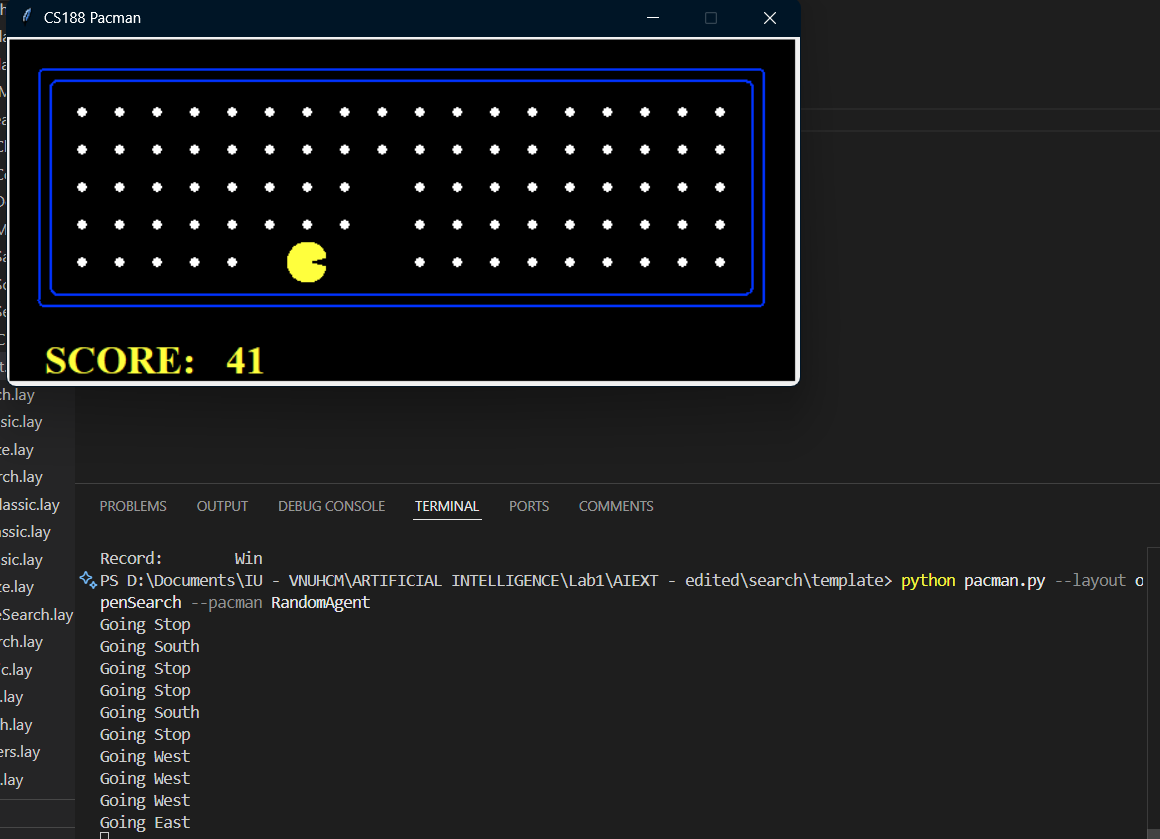
Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, màu đen, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

Running RandomAgent in openSearch.lay



While running RandomAgent in openSearch.lay:

Attempt 1 - 200 in 11.07s  
Attempt 2 - 115 in 11.00s  
Attempt 3 - 160 in 11.01s  
Attempt 4 - 210 in 10.97s  
Attempt 5 - 99 in 11.10s  
Attempt 6 - 190 in 11.03s  
=> Average 162.3 pts in 11s with RandomAgent in openSearch environment.  
It typically consumed everything in the center and was less likely to consume food on the edges.

1. **Exercise 3:**

Created BetterRandomAgentẢnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Nội dung do AI tạo ra có thể không chính xác.

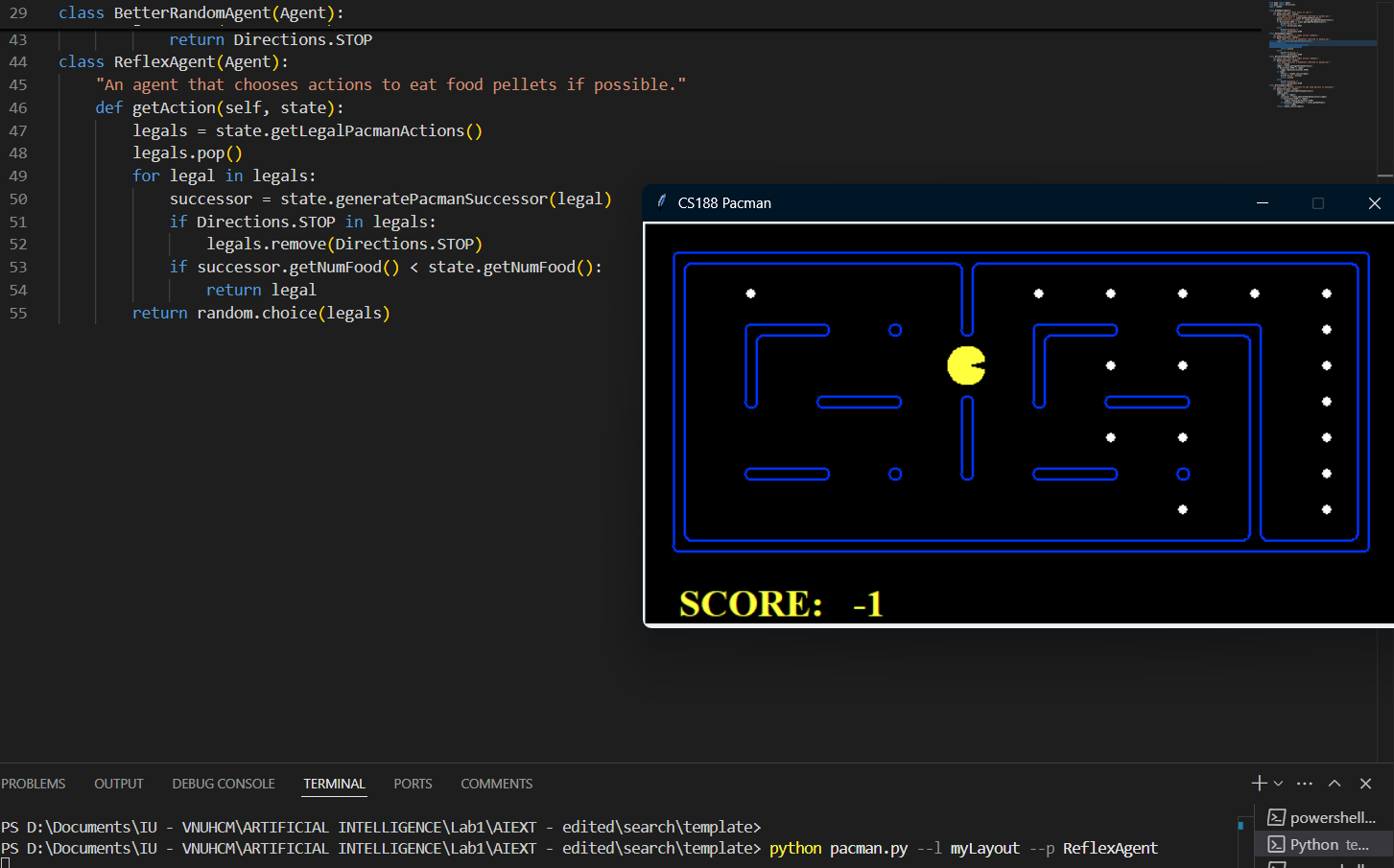
Running BetterRandomAgent in openSearch.lay:

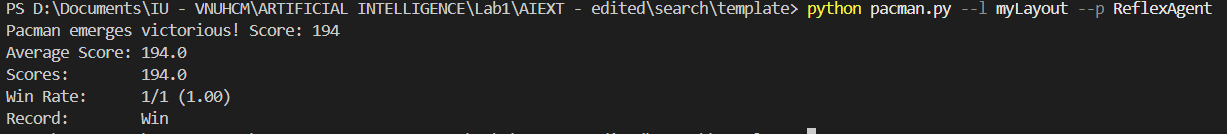
Attempt 1 - 225 in 11.07s  
Attempt 2 - 212 in 11.15s  
Attempt 3 - 160 in 11.00s  
Attempt 4 - 256 in 11.01s  
Attempt 5 - 145 in 11.12s

Attempt 6 - 155 in 11.10s  
=> Average 192.17 points in 11 seconds with BetterRandomAgent in the openSearch environment.  
BetterRandomAgent traveled without halting in a random path. RandomAgent moved more slowly than BetterRandomAgent. As a result, it was penalized with fewer points. The agent moved in the direction of the food path, and when there were spaces, it went randomly until it found the food in the openSearch layout.

1. **Exercise 4:**

Created ReflexAgent and tried running it in myLayout.lay





Running ReflexAgent in openSearch.lay

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Nội dung do AI tạo ra có thể không chính xác.

The agent always used the zig-zag technique to consume the entire right side of the map in an openSearch environment. Then, at random, it entered the other side of the map to locate food pellets, which it then consumed entirely. Of course, for the remaining part, it continued to use the zig-zag technique to consume all the food. The time for the game to finish in the openSearch layout using the ReflexAgent class was quite higher than in the BetterRandomAgent one. However, overall, they were similar to each other.

Pac-man’s position: **gameState.getPacmanPosition()**All gosts’ position: **gameState.getGhostPositions()**Walls’ location: **gameState.getWalls()**Capsules’ position: **gameState.getCapsules()**Each food pellet’s position: **gameState.getFood()**The total number of food pellets still available: **gameState.getNumFood()**Whether it has won or lost the game: **gameState.isLose()**, **gameState.isWin()**Pac-man’s current score in the game: **gameState.getScore()**