Student’s Name: Phạm Lê Thanh Nhàn

Student's ID: ITDSIU20073

**LAB 1**

**Exercise 1:** The agent’s behavior

|  |  |  |
| --- | --- | --- |
| **Behavior** | **Agent in tinyMaze** | **Agent in mediumMaze** |
| Does it get to the food? | Agent can get the food in a short time, but not always. | Agent may get the food. |
| Does it crash? | Yes, it cannot recognize the walls to ignore. | |
| Does it get stuck? | No, there is no case in which it is ended up in a position where all possible moves are illegal. | |
| Performance | It takes less actions to reach the food. | It takes more actions to reach the food 🡪 more complexity. |
| Efficiency | The RandomAgent does not find the food effectively. It makes several unnecessary or redundant moves before reaching the goal state. | |

**Exercise 2:**

A screenshot of a video game

Description automatically generated

Figure 1. myLayout environment

A screen shot of a game

Description automatically generated

Figure 2. openSearch environment

In openSearch environment, the average score I have got is 464.

**Exercise 3:**

Random agent without stopping, it takes less time to reach food as it does not have to do the “STOP” action, which is not helpful for the agent’s goal (getting more food) but also wasting the agent’s time.

**Exercise 4:**

Pac-Man’s percepts:

• His position: pacman\_position = state.getPacmanPosition()

• The position of all the ghosts: ghost\_positions = state.getGhostPositions()

• The locations of the walls: walls = state.getWalls().asList()

• The positions of the capsules: capsule\_positions = state.getCapsules()

• The positions of each food pellet: food\_positions = state.getFood().asList()

• The total number of food pellets still available: state.getNumFood()

• Whether it has won or lost the game: state.isWin(), state.isLose()

• His current score in the game: score += state.getScore()