Pac-Man

You need to program the Pac-Man game, and its screenshot is shown in Figure 1. The player controls Pac-Man, who must eat all the dots inside an enclosed maze while avoiding four colored ghosts. Eating large flashing dots called " Power Pellets" causes the ghosts to temporarily turn blue, allowing Pac-Man to eat them for bonus points.

Yor can search Pac-Man and try to play the game first.

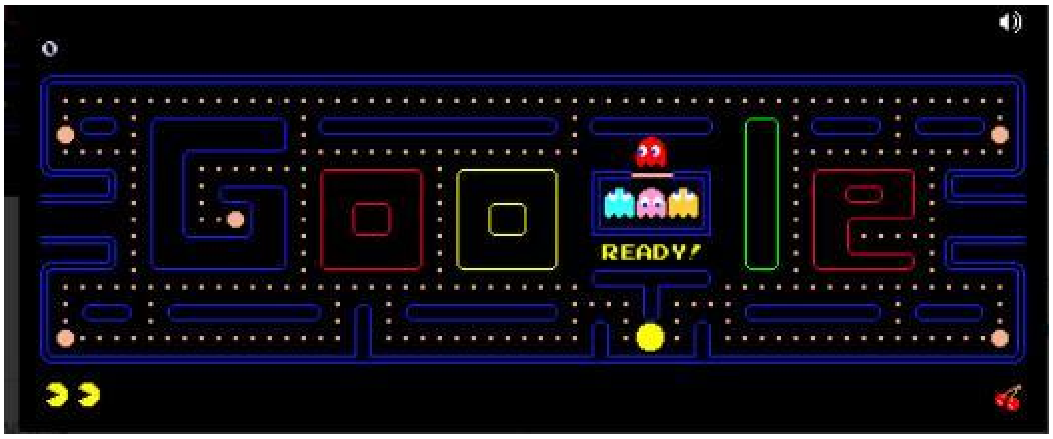

Figure 1 The screenshot of Pac-Man

Assessment Criteria:

The overall score 100 will be computed according to the program score (60%), the document score (30%) and the GUI (graphical user interface) score (10%). The program score will be assessed based on the score you get in the game.

The Pac-Man Game Document

Class： CS21-1

Student ID：211302104

Student Name：李嘉梁 (in Chinese)

Development Tool(s) and Language：PyCharm Python

1. Algorithm(s)

Use pseudo code to illustrate the algorithms(s) and try to analyze the complexity of the algorithm(s).

a) Greedy-BFS:

```
Input: cur_position.
Output: next_direction.
1.If pacman has already won the game  returns -1.
2. Initialize two 2D arrays:
   i. bfs_visitmap: a map of visited cells, initialized to infinity
   ii. bfs_dire: a map of directions to take to reach each cell from the starting position, initialized to -1
3. Find the shortest path from the current position to the first dot found by the bfs algorithm.
4. Return the result.
```

Complexity:

The time complexity of it depends on BFS which is O(V+E).

The space complexity of it depends on BFS is which is O(V+E).

V：Number of accessible cells. E: Number of accessible paths among cells.

b) Greedy-DFS:

```
Input: cur_position.
Output: next_direction.
1.If pacman has already won the game  returns -1.
2.  If the current goal is None or pacman has reached the current goal.
```

i. Initializes a 2D-array dfs_visitmap with the size of the game board with 0.

ii. Calculates the shortest path to the nearest dot using the dfs algorithm and updates the dfs_goal to the position of the nearest dot.

iii. Calculates the shortest path from pacman's current position to the first dot found by the dfs algorithm.

iv. Returns the direction to move to reach the next cell in the path.

3. If the current goal is not None and pacman has not reached the current goal

i. Calculates the shortest path from pacman's current position to the current goal using the dfs algorithm.

ii.Returns the direction to move to reach the next cell in the path.

Complexity:

The time complexity of it depends on DFS which is O(V+E).

The space complexity of it depends on DFS which is O(V+E).

V：Number of accessible cells. E: Number of accessible paths among cells.

c) DP:

Input: cur_position.

Output: next_direction.

1. If pacman has already won the game returns -1.
2. If the current optimal path is None.

i. Initializes an empty list tempobj to store the positions of all the dots.

ii. Initializes a 2D-array to store the optimal distances and next dot positions for each subset of dots. Fills in it using bit manipulation, and backtracks to find the optimal path to all the dots.

3. If the current optimal path is not None.

i.Finds the next dot in the path that has not been eaten, and returns the direction to move in to reach the next cell in the path.

4. If all dots have been eaten, returns -1.

Complexity:

The time complexity of it is O(V^2*2^V).

The space complexity of it is O(V*2^V).

V：Number of accessible cells.

d) Heuristic:

Input: cur_position.

Output: next_direction.

1.If pacman has already won the game returns -1.

2. If the goal has already been set or if the current position is the goal position:

a. Find the nearest possible goal position using the bfs algorithm.

b. Iterate through all possible goal positions and select the one with the shortest Manhattan distance to pacman.

c. Set the goal and path to the selected goal position and return the second move in the path.

3. If the goal has already been set and the current position is not the goal position:

i. Iterate through the path to the goal and return the next move in the path.

Complexity:

The time complexity of it depends on BFS which is O(V+E).

The space complexity of it depends on BFS which is O(V+E).

V：Number of accessible cells. E: Number of accessible paths among cells.

e) Naive:

Input: cur_position.

Output: next_direction.

1. If pacman has already won the game returns -1.
2. Finding the distances and paths to all the ghosts using the bfs algorithm.
3. If all the ghosts are far away from pacman (mindis >= 6),

   i. Moves pacman towards the nearest dot using the bfs algorithm.

4. If any ghost is nearby (mindis < 6)

   i. Finds the direction that moves pacman farthest away from the closest ghost by checking all four directions and calculates the Manhattan distance between pacman's position and each neighboring cell that is valid on the game board.

   ii. Chooses the direction that maximizes the minimum distance between Pac-Man's neighboring cells and all the ghosts.

Complexity:

The time complexity of it depends on BFS which is O(V+E).

The space complexity of it depends on BFS which is O(V+E).

V：Number of accessible cells. E: Number of accessible paths among cells.

f) Sophisticated:

Input: cur_position.

Output: next_direction.

1. If pacman has already won the game returns -1.
2. If pacman is not invincible.
   i. If any ghost is nearby (mindis <= 4), the function chooses a move that will take pacman away from the closest ghost.
   ii. If pacman is not surrounded by multiple ghosts, it will make a further judgement according to the situation.
   iii. If there are no nearby ghosts, pacman will move towards the nearest dot.
3. If pacman is invincible, it will always move towards the nearest dot, regardless of the proximity of the ghosts.

Complexity:

The time complexity of it depends on BFS which is O(V+E).

The space complexity of it depends on BFS which is O(V+E).

V：Number of accessible cells. E: Number of accessible paths among cells.

2. The Development Procedure

1) Introduce the procedure of developing the project.

a) Fundamental Infrastructure Implement: Collect images, design the GUI (including map & buttons) and implement it based on pygame library.

b) Game Logic Implement: Design the classes & functions based on game logic. e.g. main.py is designed as main function for handling events, agent.py is designed for defining agent classes (including Pacman & Ghost) and the actions of them, painting is designed for updating the frame, srcload.py is designed for loading resources (including map & images) and initializing the game settings (including window & font & buttons etc.), parameter.py is designed for defining constant. util.py is designed for defining utility classes (including Map & Button & Direction) and the methods of them.

c) Ghost Search Algorithm: Design including Random & Predict algorithm (agentctrl.py).

d) Pacman Search Algorithm Implement: Design including Greeedy-BFS &Greedy-DFS &DP & Heuristic & Naive & Sophisticated algorithm (agentctrl.py & pathctrl.py).

e) Synthesis & Optimize & Write README.md.

2) Besides, you need to describe what problems you have encountered during the procedure and the corresponding solutions.

a) Problem: How to let the pacman move continuously?

Solution: Design the relative coordinate and let the frame repaint frequently.

b) Problem: How to let pacman move automatically?

Solution: Let pacman move to the first dot found uneaten to eat continuously until all the dots are eaten.

c) Problem: How to let pacman move intelligently?

Solution: Add judgement logic into the search algorithm (Naive & Sophisticated algorithm).

3) You also need to summarize what you have learned by developing the project.

a) Ability of Algorithm Analyze: I enhanced it while I was analyzing and adjusting my code to make it more efficient and exquisite.

b) Ability of Algorithm Design: I improved it while I was thinking how to use the original algorithms in the new situations and designing new algorithms.

c) Ability of Algorithm Implement: I extended it while I was translating abstract pseudo code into concrete real code.

d) Ability of Debugging: I acquired how to use PyCharm to debug Python program.

e) Ability of Solving Problems with Complex Process: I learned how to design the whole frame, how to divide the entire project into specific module, and then divided into classes and functions, which known as "Top-Down Approach".

f) Ability of Literature Searching: I mastered how to search information efficiently and effectively via the internet.

g) Ability of Innovation: I designed two new algorithms based on original algorithms and game rules (Naive & Sophisticated algorithm).

3. References

Must have English references.

[1]https://blog.csdn.net/dfs0002/article/details/127147240?ops_request_misc=&request_id=&biz_id=102&utm_term=%E5%90%83%E8%B1%86%E4%BA%BA%20python&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduweb~default-7-127147240.142^v88^control,239^v2^insert_chatgpt&spm=1018.2226.3001.4187

[2]https://blog.csdn.net/junruitian/article/details/79055577?ops_request_misc=&request_id=&biz_id=102&utm_term=%E5%90%83%E8%B1%86%E4%BA%BApython%20%E4%BA%BA%E5%B7%A5%E6%99%BA%E8%83%BD&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduweb~default-1-79055577.142^v88^control,239^v2^insert_chatgpt&spm=1018.2226.3001.4187

[3]https://blog.csdn.net/weixin_42062229/article/details/94637972?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522168904612416800222896948%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=168904612416800222896948&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-2-94637972-null-null.142^v88^control,239^v2^insert_chatgpt&utm_term=%E5%90%83%E8%B1%86%E4%BA%BApython%20%E4%BA%BA%E5%B7%A5%E6%99%BA%E8%83%BD&spm=1018.2226.3001.4187

[4]https://www.google.com.hk/search?q=pacman&newwindow=1&ei=eh6yZO_HI6Ob2roPueeYkAI&ved=0ahUKEwjv7MeM7o-AAxWjjVYBHbkzBiIQ4dUDCBA&uact=5&oq=pacman&gs_lp=Egxnd3Mtd2l6LXNlcnAiBnBhY21hbjIHEC4YigUYQzIFEAAYgAQyBRAAGIAEMgUQLhiABDIFEAAYgAQyBRAAGIAEMgUQABiABDIFEAAYgAQyBRAAGIAEMgUQABiABDIWEC4YigUYQxiXBRjcBBjeBBjgBNgBAUjWPlD1G1jIMnADeAGQAQGYAawDoAGdDaoBCTAuNC4xLjAuMrgBA8gBAPgBAcICChAAGEcY1gQYsAPCAgcQLhiABBgKwgIHEAAYgAQYCsICFhAuGIAEGAoYlwUY3AQY3gQY4ATYAQHCAgsQLhiABBjHARivAcICBxAAGIoFGEPiAwQYACBB4gMFEgExIECIBgGQBgq6BgYIARABGBQ&sclient=gws-wiz-serp

[5]https://en.wikipedia.org/wiki/Pac-Man

大作业提交方式:
2023 年 6 月 30 日之前把源代码(文件夹命名为“源代码”) 和大作业报告(单个文件, 模板参见“The Pac-Man Game Project.docx”文件的第 2 页，报告另存为 pdf 格式, 命名为“学号_姓名.pdf”) 打包为压缩文件“学号姓名.rar”(比如“180101309 石国林.rar”), 上传至ftp://211.71.149.53/徐艳艳/课程作业/Algorithm Design and Analysis (Project)文件夹中。如果上传后需要修改，则新传文件命名为“学号姓名_修改 1.rar”, 以此类推，重新上传即可(以最后上传的文件为准)。

请注意: 打包交上来的.exe 务必是在 win10 上能直接运行的(不用再安装或配置其他环境)