VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
UNIVERSE OF SCIENCE
FACULTY OF ADVANCED INFORMATION TECHNOLOGY

PRACTICAL PROJECT: LOGICAL AGENT
INTRODUCTION TO ARTIFICIAL INTELLIGENCE – CSC14003

PROJECT #2: WUMPUS WORLD

—o0o—

INSTRUCTOR(S)

MS. NGUYỄN NGỌC THẢO
MR. NGUYỄN THANH TÌNH
MS. HỒ THỊ THANH TUYỀN
—o0o—

GROUP'S INFORMATION

NO.	STUDENT ID	FULL NAME	EMAIL
1	22127174	NGÔ VĂN KHẢI	nvkhai22@clc.fitus.edu.vn
2	22127322	LÊ PHƯỚC PHÁT	lpphat22@clc.fitus.edu.vn
3	22127388	TÔ QUỐC THANH	tqthanh22@clc.fitus.edu.vn
4	22127441	THÁI HUYỄN TÙNG	thtung22@clc.fitus.edu.vn

HO CHI MINH CITY, AUGUST 2024

TABLE OF CONTENTS

TEACHER'S COMMENTS

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

……

Date: **August, …, 2024**

Graded and Commented Teacher(s)

PLEDGE

Our team declares that this research was conducted by the team, under the supervision and guidance of the teachers of the **Introduction to Artificial Intelligence – CSC14003** subject: **Ms. Nguyen Ngoc Thao**, **Mr. Nguyen Thanh Tinh**, and **Ms. Ho Thi Thanh Tuyen**. The results of this study are legal and have not been published in any form before. All documents used in this study were collected by the team and from various sources and are fully listed in the references section. In addition, we also use the results of several other authors and organizations. All are properly cited. In case of copyright infringement, we are responsible for such action. Therefore, **Ho Chi Minh City University of Science (HCMUS)** is not responsible for any copyright violations committed in our research.

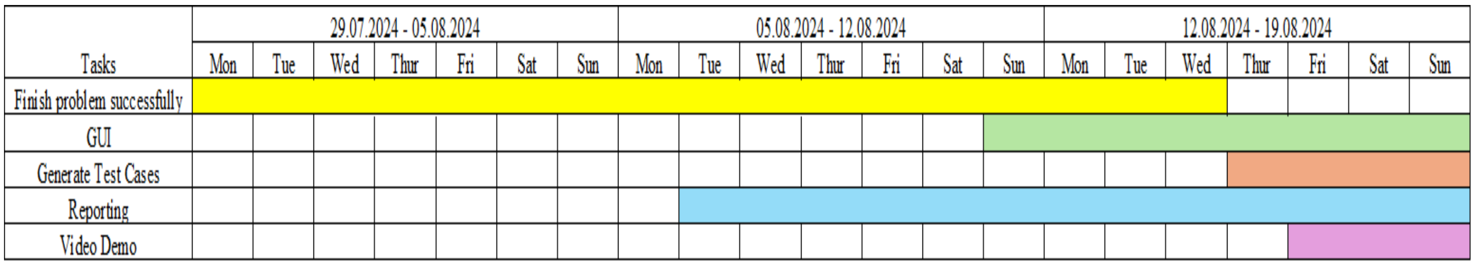
RESEARCH PROJECT

I. Project Evaluation

1. Work Assignment Table

StudentID	Full Name	General Tasks	Detailed Tasks	Completion
22127174	Ngô Văn Khải	GUI	Implementing the GUI (Coding).	100 %
			Describe in detail the code of GUI (Report).	100 %
		Test cases	Describe in detail the test cases (Report)	100 %
22127322	Lê Phước Phát	Algorithms	Describing in detail the algorithms of the problems successfully (Report).	100 %
		Test cases	Performance experiments with some reflection and comments (Report)	100 %
22127388	Tô Quốc Thanh	Algorithms	Implementing the algorithms of the problems (Coding).	100 %
		Test cases	Performance experiments with some reflection and comments (Report)	100 %
22127441	Thái Huyễn Tùng	GUI	Video demonstrating	100 %
			Game Play (Code + Report)	100 %
		Test cases	Generating test cases with some different attributes (position and number of Pit, Gold, and Wumpus) (Report)	100 %
			Describe in detail the test cases (Report).	100 %

2. Assignment Plan



Tasks	29.07.2024 - 05.08.2024							05.08.2024 - 12.08.2024							12.08.2024 - 19.08.2024						
	Mon	Tue	Wed	Thur	Fri	Sat	Sun	Mon	Tue	Wed	Thur	Fri	Sat	Sun	Mon	Tue	Wed	Thur	Fri	Sat	Sun
Finish problem successfully	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨				
GUI														🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Generate Test Cases																		🟧	🟧	🟧	🟧
Reporting									🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
Video Demo																			🟪	🟪	🟪

3. Self-evaluation of the completion rate at each level of the project and other requirements

No.	Details	Completion Rate
1	Finish the problem successfully.	100 %
2	Graphical demonstration of each step of the running process.	100 %
3	Generate at least 5 maps with different structures such as position and number of Pit, Gold, and Wumpus.	100 %
4	Report your algorithm, and experiment with some reflection or comments.	100 %
5	Video demonstrating	100 %

II. Detailed Program System Description

1. Environmental Requirements

- **Python version**: 3.10^{+} with Pygame graphical module.
- **Operating System**: the command may vary across platforms, so you need to run this program on Windows.

2. Settings

- Firstly, you need to set up the virtual environment in Python by using this command line below:

```
pip install virtualenv
python -m venv venv.
```

- Secondly, you need to activate the virtual environment in Python by using this command line below:

```
.venv\Script\activate
```

- Thirdly, we will install all libraries in the file "requirements.txt" using this command line below. Note that we need "cd" into the folder "Source".

```
pip install -r requirements.txt
```

- Finally, you need to run the application by executing the command "python main.py" in the console or terminal.

3. Supporting Classes and Functions

In our program, we designed some supporting classes and functions to prepare for the main algorithms to be implemented.

a. *Class cell (...)*

Implementation file: ***Source/algorithms/cell.py***

- This **"cell"** class is designed to represent a single cell on the map grid used for checking percepts and storing all elements in this cell.
- In this class, we will mention the coordinates of this cell with **y-coordinate** (row) and **x-coordinate** (column) of this cell, a list of **"elements"** that store elements present in this cell (such as gold, Wumpus, pit, healing potion, and poisonous gas, …). Then, this class will check that the cell has been visited by the agent or indicate the presence of a breeze (signal nearby pits), a stench (signal nearby Wumpus), a whiff (signal nearby poisonous gas), and a glow (signal nearby healing potion). Besides that, we all check the scream of Wumpus when it died by the agent's shooting, and check if it is a safe cell (that no pit or Wumpus in here).

b. *Function write_output (...)*

Implementation file: ***Source/utils/write_output.py***

- This function will write down the output of the agent's actions in the output file. It provides a clear and formatted summary of the game, including whether the agent won or lost, and details the agent's final score and health.

c. *Function a_star (...)*

Implementation file: ***Source/algorithms/a_star.py***

This function will help us determine the path to the desired cell with the lowest path cost along with the L1 distance heuristic.

This function will receive into the map marked which cells will be accessible by agent (1) and which cells will not be accessible by agent (-1) (through the create graph() function after running the dfs algorithm for agent finds possible paths), current point, destination point, and program.

First, we will initialize the visited list (nodes that have been visited), cost (path cost) and parents of the current node, as well as the list of nodes that the agent uses healing potion (tmp_heal).

After that, we will initialize the 2D array for the visited lists (corresponding to each cell having an initial False value meaning there are no visited cells), cost (the cost of each cell is infinite) and parent (corresponding to Each cell will have a parent node (-1, -1), which means the parent node of that cell does not exist).

We will assign the cost of start (the current cell to start browsing) equal to 0. Create a frontier with cost + heuristic = 100 if that cell definitely has poisonous gas, otherwise equal to 0, the current node, the agent's current health score and Current amount of healing potion.
First, we will take the cell with the lowest priority calculated by cost + heuristic along with the current health and number of healing potions. We will check to see if that cell has been approved. If not, we will mark the cell as approved. If the retrieved cell is different from the starting cell, we will perform an action to let that agent go to the new cell.
If that cell is a cell with poisonous gas, we will consider the current amount of health. If the current amount of health is 25, it means that if you are poisoned again, you will die. At this time, if the number of stored health potions (potions) is > 0, we will use it and the health will be restored by 25% and we will add this box. Go to the tmp_heal list to check the agent has performed a healing action here. Note that this is just the agent's prediction that this cell probably has poisonous gas, but in reality it is not certain, so we will check if this cell has "P_G" and if the agent accidentally passes through it, 25% of his current health will be deducted. and if current health <= 0, the agent will be marked as dead (lock block -1).
We will start finding the child nodes of the current node by looking at the four sides up, down, left and right. For each child node, we will check whether the child node exceeds the map limit. If not, we will check to see if this child node is not visited and not blocked, then we will update the child node's cost equal to the current node's cost to 1. If the cost of the new cost < the cost of the current child node, we will update the cost of the child node. The child node is new_cost and the child node's parent is the current node.
We will use L1 distance to calculate the heuristic from the child cell to the root node. If the agent turns too many 90 degrees, it will update the heuristic to 10 to reduce the frequency of performing actions. Finally, we will push this child node to the frontier and explore the set to find the path. When the goal node has been approved, we will update the current amount of health and the number of health potions, otherwise, we will return an empty string.
Then we will trace the path and return the path from the start node to the goal node and also update the coordinates of the nodes where the agent performs healing actions when finding the path.
positi

d. *Function create_graph (...)*
Implementation file: Source/algorithms/a_star.py

This function will create a new map based on the trace pathfinding by an agent. It initializes a 2D matrix where each cell is initially marked as inaccessible (-1). It then updates this matrix to mark the accessible cells as 1 based on the input list.

e. *Function main ()*

Implementation file: Source/main.py

The main() function will mainly link the backend and frontend. First, we will initialize agent = Agent () and program = Program (). Then let the player choose the designed map and implement the dfs algorithm to find a path without killing Wumpus, but only based on the available go-to cells on the original map. Then, we will check which cells may have Wumpus and pit.
We will create graph1 to find the way to the cells with Wumpus and shoot them and graph2 to find the way to return to the original cave.
We will call the Wumpus fire function and will save the new map to graph1. Then, we will traverse the agent's path and assign actions to the agent. At the same time, points will be calculated for the agent.

4. Graphical User Interface (GUI) and Gameplay

a. GUI

Function showGameBackground(…)

Implementation file: Source/ui/image.py

This function will display the background when the screen shows cave map. There are 5 different backgrounds for 10 input maps. The images are called from ***ui/assets*** folder.

Function showMenuBackground (…)

Implementation file: Source/ui/image.py

This function will display the menu background. The image is called from ***ui/assets*** folder.

Class ImageElement (…)

Implementation file: Source/ui/image.py

- **"ImageElement"** class is designed to store all images that could be shown on cave map.

Class Map (…)

Implementation file: Source/ui/image.py

- **"Map"** class inherits from **"ImageElement"** class.
- **"Map"** class is designed to call showing images functions from **"ImageElement"** class.

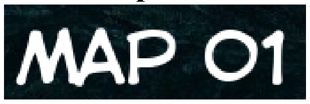
Class Text_Display (…)

Implementation file: Source/ui/text.py

- **"Text_Display"** class is designed to show text on the window.

Class Info (...)
Implementation file: Source/ui/image.py

- **"Info"** class inherits from **"Text_Display"** class.
- **"Info"** class is designed to show information and notification while the window is showing cave/Wumpus World map.
- **"showMapInfo"** function shows map index.


MAP 01

- **"showPoint"** function shows Agent's current point.

POINT: 10000

- **"showHP"** function shows Agent's current HP.

HP: 100

- **"showHealingPotion"** function shows the number of healing potions that Agent has.

HEALING POTION(S): 0

- **"showLeftBar"** function shows all mentioned lines.
- **"showNoti"** function shows the game notification. There are 5 notification lines:
    - Press Enter to run the problem.
    - Agent is moving. Press Enter to return to menu.
    - Agent exits the cave successfully. Press Enter to return to menu.
    - End game. Press Enter to return to menu
    - Agent dies. Press Enter to return to menu.

Class BackButton (...)
Implementation file: Source/ui/choice.py

- **"BackButton"** class stores the back button's logic.

Class NextButton (...)
Implementation file: Source/ui/choice.py

- **"NextButton"** class stores the back button's logic.

Class ChoiceList (...)

Implementation file: Source/ui/choice.py

- **"ChoiceList"** class stores the choice list's logic.

Class Choice (...)
Implementation file: Source/ui/choice.py

- **"Choice"** class stores the choice's logic.

Class Credit (...)
Implementation file: Source/ui/credit.py

- **"Credit"** class is designed to show credit page.

Function showMenu (...)
Implementation file: Source/ui/main_ui.py
This function will display the menu page by background and choice. Program user can choose the map he/she wants to execute, see the credit or exit the program.

Function showWumpusWorld (...)
Implementation file: Source/ui/main_ui.py
"showWumpusWorld" function will display the cave/Wumpus World map when the Agent has not been at starting point yet.

Function showAgentMove (...)
Implementation file: Source/ui/main_ui.py
After seeing the **"showWumpusWorld"** shown, user can press Enter so the program will find the Agent's list of coordinates, actions, points, HP and potions. Then **"showWumpusWorld"** function will display the Agent's action through the cave.

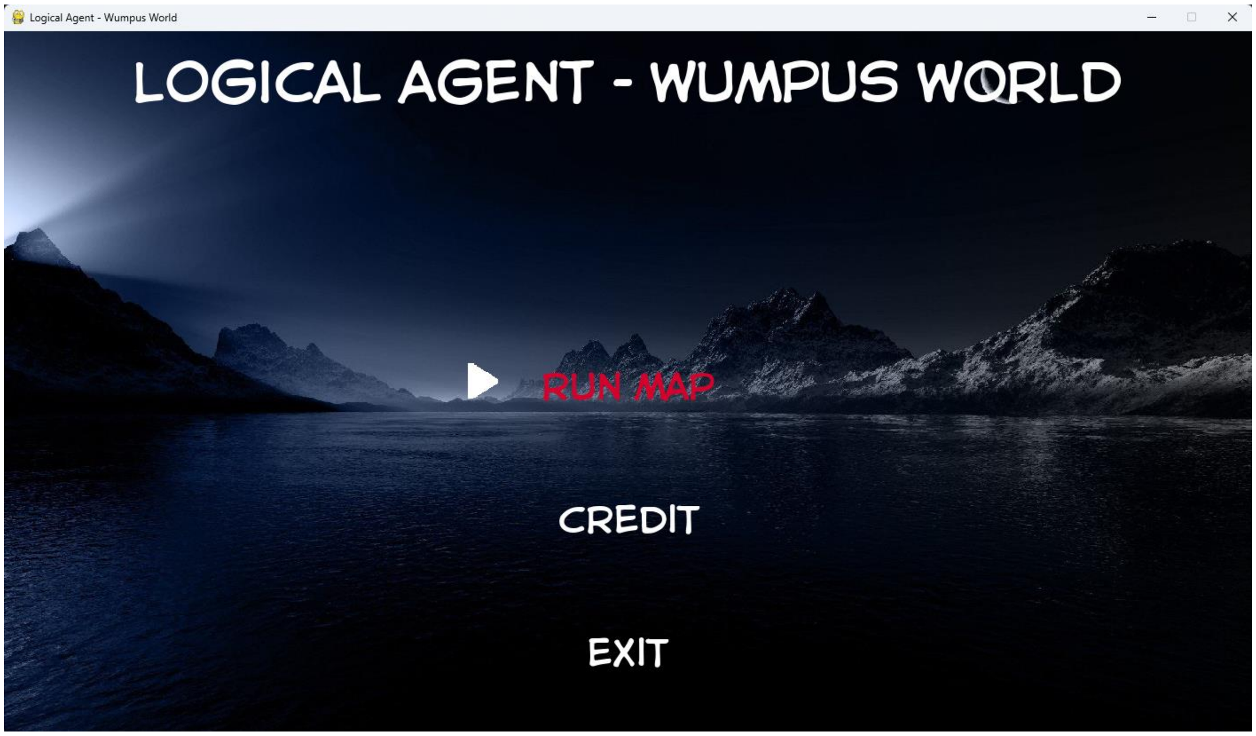
b. Gameplay

i. How to play

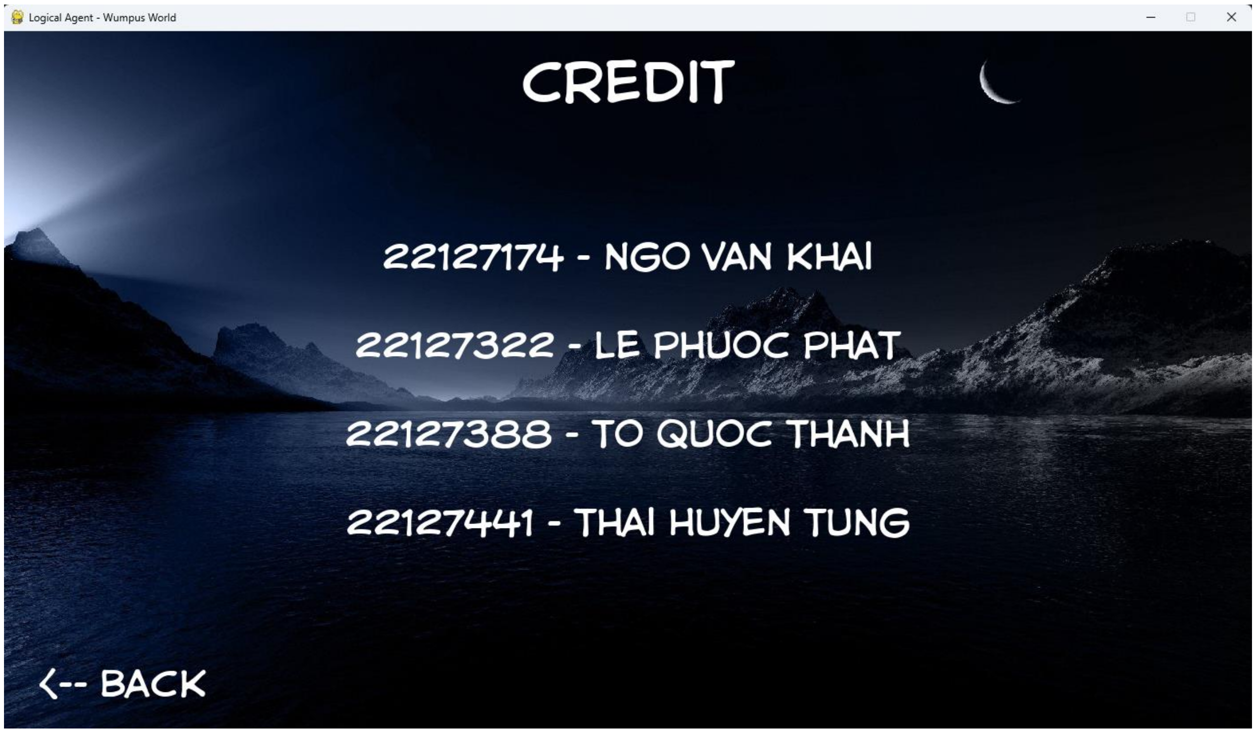
- We can choose an option in the menu of each screen by pressing the enter key on the keyboard.
- Also, we can move up or down to select an option by using the up-arrow key and down-arrow key on the keyboard respectively.

- Next, if we want to return to the previous page or move to the next, we must use the left and right arrow keys, respectively.

ii. GUI description



In the Main Menu, we have three options: RUN MAP which contains 10 maps for the demonstration, CREDIT which contains founders' information, including name and ID, EXIT which closes the program.



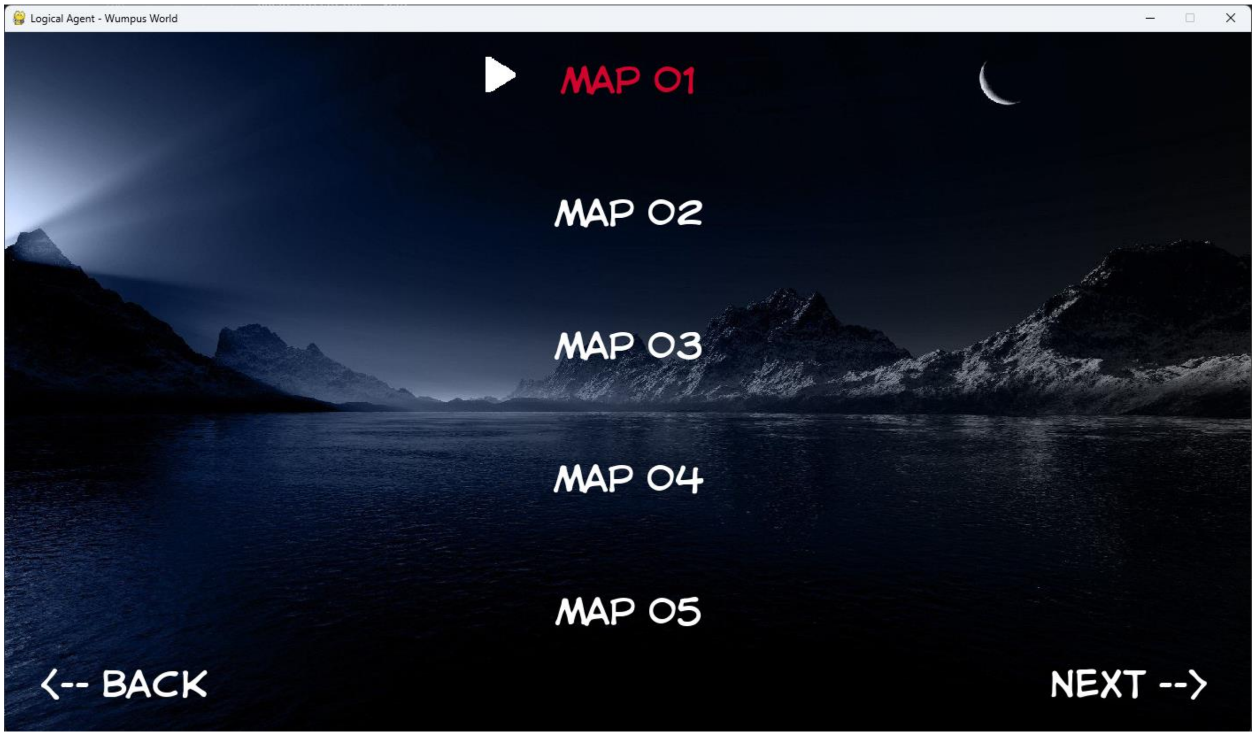
22127174 - NGO VAN KHAI

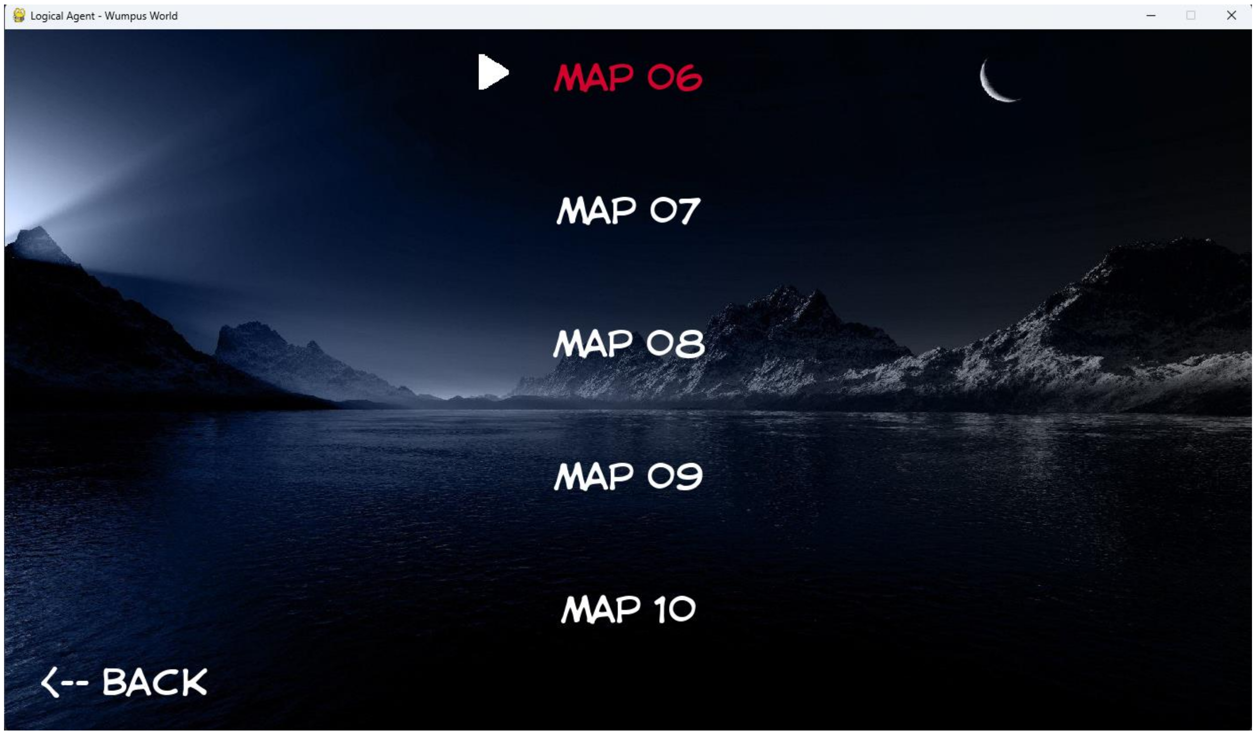
22127322 - LE PHUOC PHAT

22127388 - TO QUOC THANH

22127441 - THAI HUYEN TUNG

Here is the Credit screen, the project is developed by four members on the figure. We can click the left arrow key to return to the Main menu.





Here, we have 10 maps in total, each screen will display 5 maps. After choosing the first option (RUN MAP) in the Main menu, we will move to the Map screen on the first page (first 5 input maps). We can move up or move down as we describe in the How to play section. In the first page, we can move next to the second page, but we can only move back when we are in the second page because there are only 10 maps.

By pressing the enter key on each option in these 10 maps, we can view different UI for each map (the content of map and the background image). After the agent finishes his mission, we can press the enter key on the keyboard one more time to return the Main Menu and choose other maps.

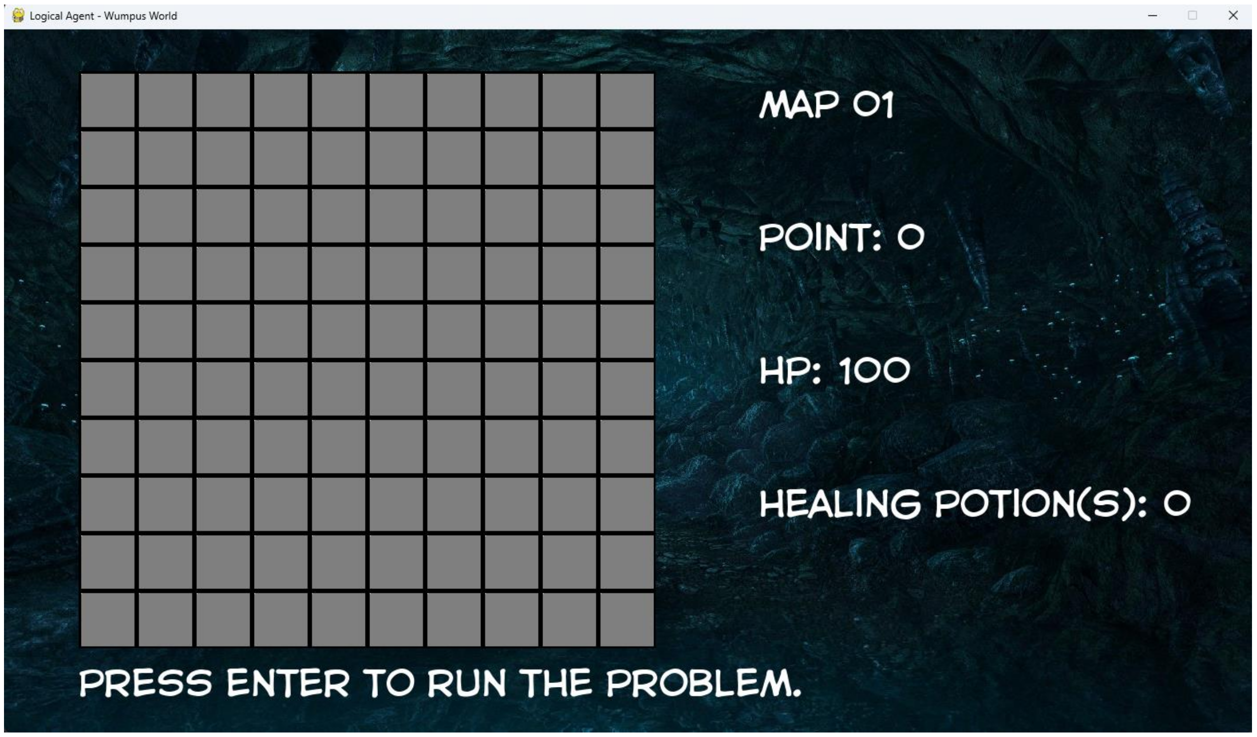


Figure 5. The UI in a specific map

We have 10 maps with the 10x10 size, after choosing one of these maps, we can press the enter key to run the map.

The left-hand side of the screen is the map, and the opposite side is the side bar which contains essential information such as the map's order, the point, the HP and the Healing Potion(s).

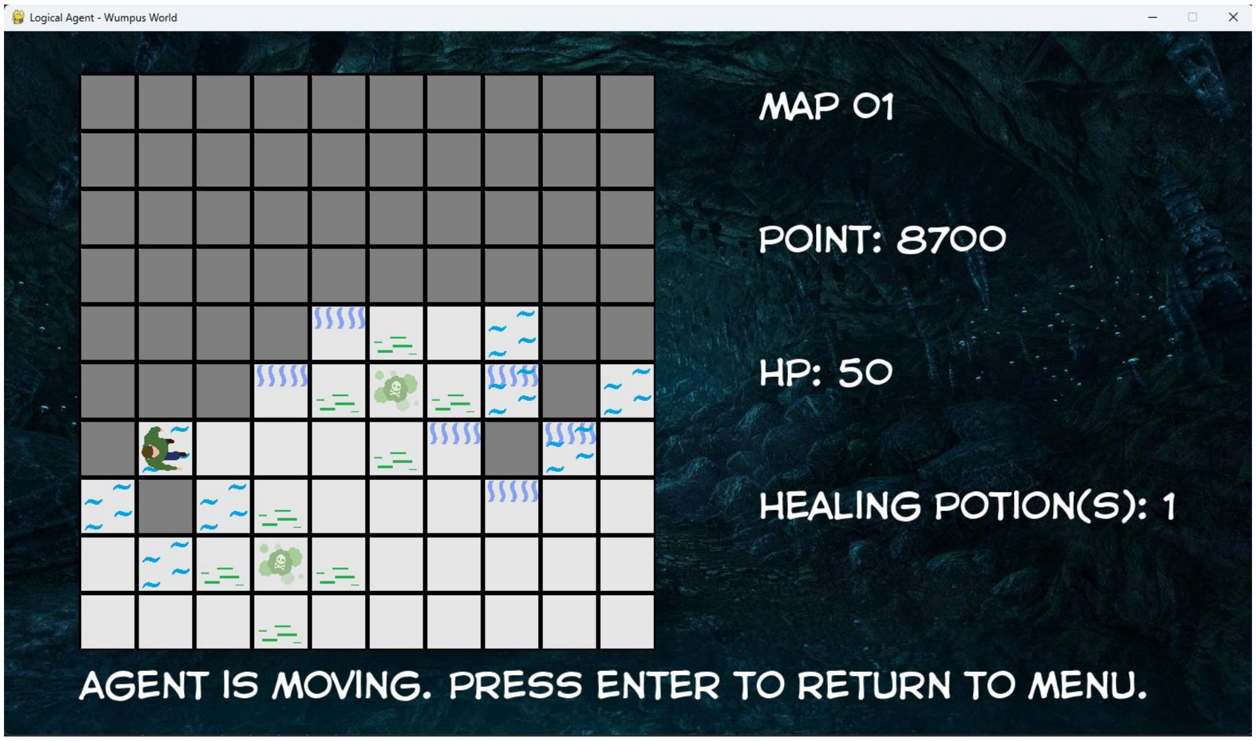

Logical Agent - Wumpus World
MAP 01
POINT: 8700
HP: 50
HEALING POTION(S): 1
AGENT IS MOVING. PRESS ENTER TO RETURN TO MENU.

Figure 6. The game is running in a specific map

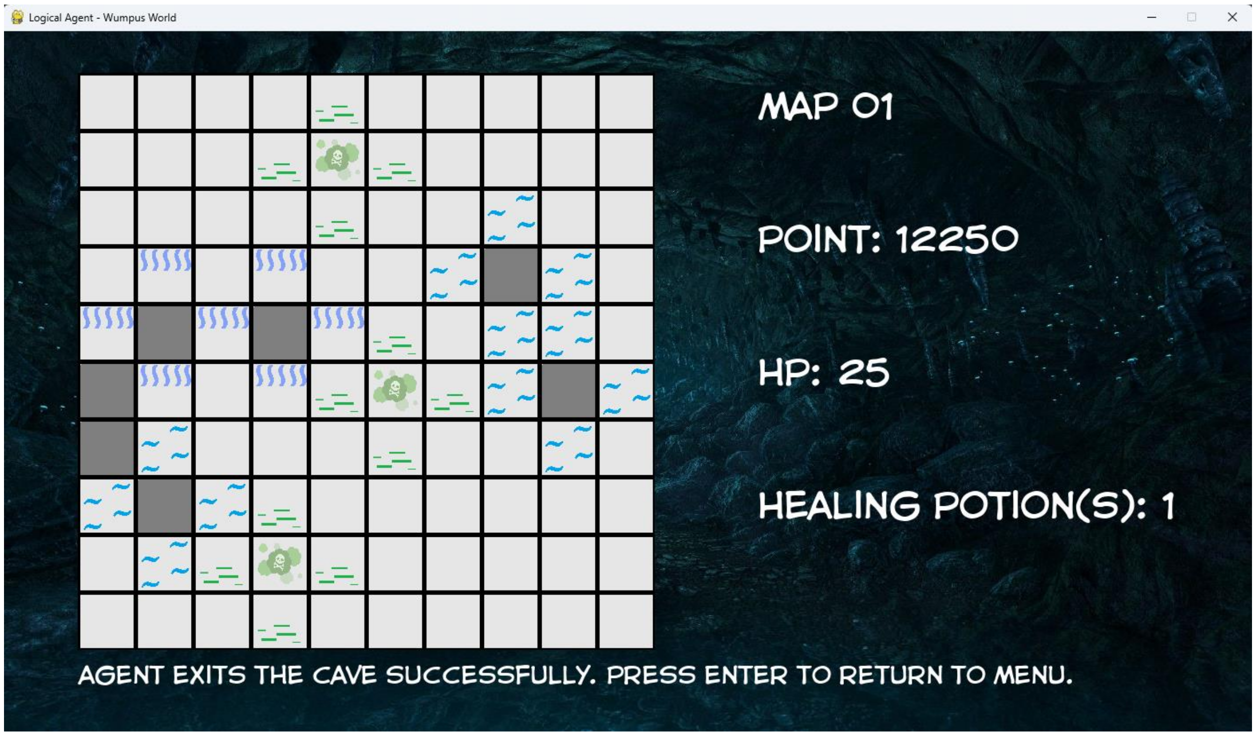

Logical Agent - Wumpus World
MAP 01
POINT: 12250
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

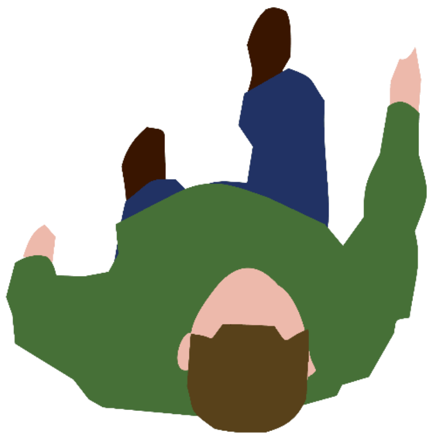
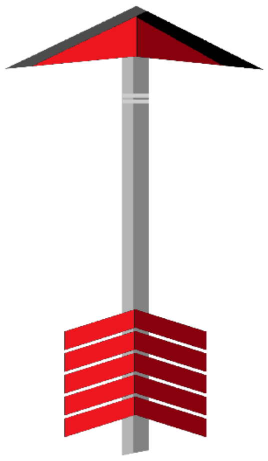
Figure 7. The game is completed in a specific map

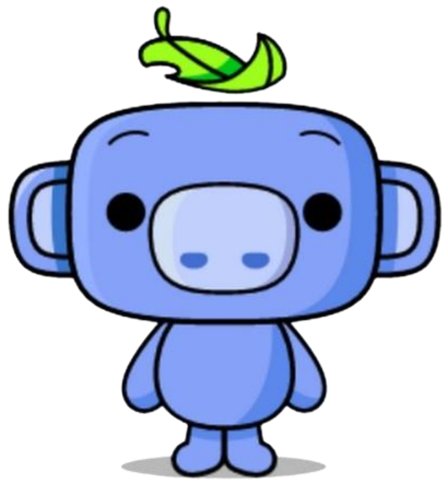
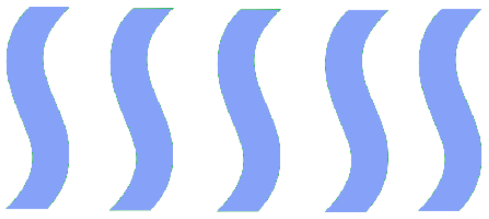
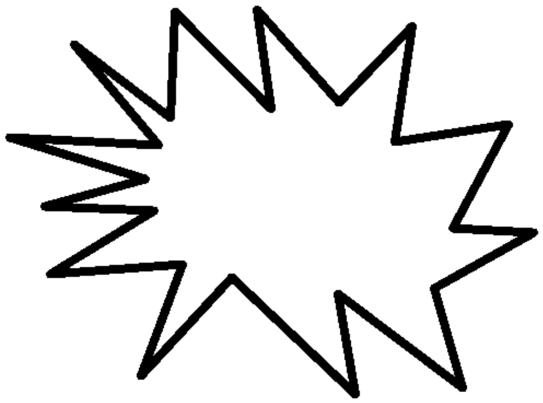
We can return to the Main Menu by pressing the enter key even while the game is running. In addition, we can also do that when the game is finished.

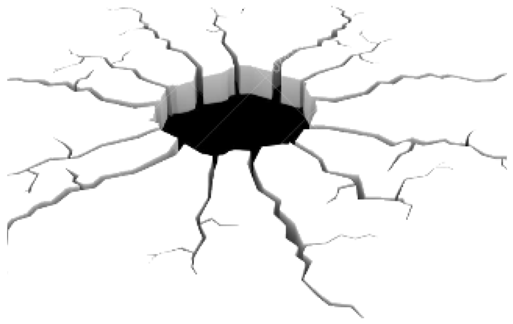
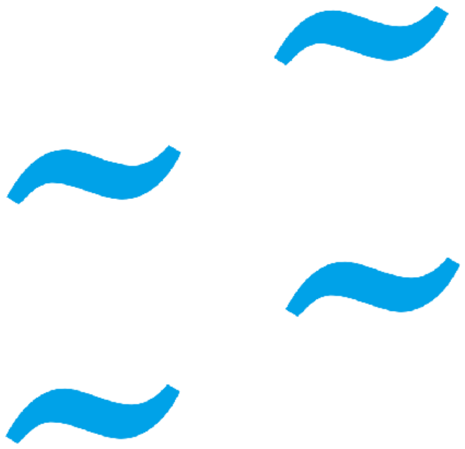
The data will be updated continuously and displayed on the side bar, so we can have a general view about what is happening while the game is running.

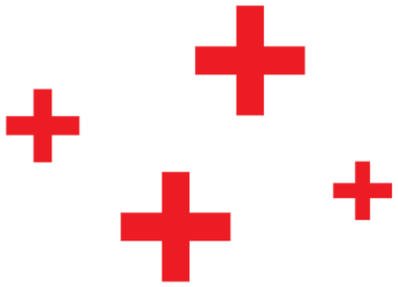
The figures above from figure 5 to figure 7 are taken from Map 01. We can have particular views for each map in the video demonstration later.

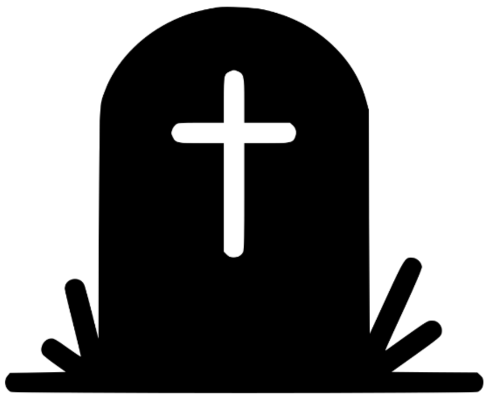
c. Graphic explanation

Agent	
Arrow - which is used to shoot monster	

Wumpus (monster)	
Stench	
Scream	

Pit	
Breeze	
Poisonous gas	

Whiff	
Healing potion	
Glow (the signal of nearly healing potion)	

Gold	
Die (when agent dies)	

III. Detailed Algorithms Description and Implementation

1. Detail Algorithms Description

For this project, we will use dfs and a_star algorithms and logic statements to implement. For dfs algorithm: Purpose: to help the agent find all possible paths based on the current knowledge base without finding and using specific inference methods. First we will create a frontier with the current cell coordinates, then we will find child nodes extending in 4 directions up, down, left and right. Then, we will retrieve the deepest node according to the LIFO mechanism. Then we will check to see if this cell has poisonous gas. If there is, we will use healing potion to update the health. Next, we will mark this cell as visited and add it to the agent's path. At the same time, update the current review node as the cell has been added to the path. Then, we will update the current cell's perceptions into the agent's knowledge base. At the same time, the agent will perform actions to pick up gold and health potions. Finally, the approved path will be returned according to each coordinate. The A* algorithm has been described in detail in the a_star() function.

2. Implementation

a. *Class Program (...)*

Implementation file: Source/algorithms/program.py

- The **"Program"** class is a crucial component in the **Wumpus World** project, responsible for managing the map, updating cells, and executing game logic. This class is designed to read data from a map file, initialize **"Cell"** objects, and update information related to effects like stench, breeze, whiff, and glow based on the elements present in each cell.
- Key Attributes
  - **"map_size"**: An integer representing the size of the map (NxN).
  - **"tmp_map"**: A temporary storage list for the map as a list of lists containing elements in each cell.
  - **"file_path"**: The file path to the map configuration file.
  - **"cells"**: A 2D list of Cell objects representing the Wumpus World grid.
  - **"MAPS"**: A list storing all the map states when they change.
- Methods
  - **__init__(self, file_path)**
    - Initializes the Program class with the provided map file path.
    - Reads the map from the file and initializes the cells in the grid.
  - **read_map(self)**
    - Reads the map data from the input file and initializes the cells with elements.
    - Parses the structure of the map and sets up the elements in each cell accordingly.
  - **reset_percepts(self, y, x)**
    - Resets the percepts for adjacent cells to the cell at coordinates (y, x).
    - Updates the percepts based on the current elements in the cell.
  - **update_percepts(self, y, x, elements)**
    - Updates percepts like stench, breeze, whiff, and glow for adjacent cells based on the elements in the current cell.
  - **init_map_info(self)**
    - Iterates through the map and updates cells with information about stench, breeze, whiff, and glow.
  - **add_stench(self, y, x)**

- Adds stench to adjacent cells of a cell containing a Wumpus.

- **add_breeze(self, y, x)**
  - Adds breeze to adjacent cells of a cell containing a Pit.
- **add_whiff(self, y, x)**
  - Adds whiff to adjacent cells of a cell containing Poisonous Gas.
- **add_glow(self, y, x)**
  - Adds glow to adjacent cells of a cell containing Healing Potions.
- **add_to_adjacent(self, y, x, object_name)**
  - Adds an effect (stench, breeze, whiff, glow) to adjacent cells based on the specified object (e.g., Wumpus, Pit, Poisonous Gas, or Healing Potions).
- **display_map_test(self)**
  - Displays the map with all information for testing purposes.
- **return_map_test(self)**
  - Returns the map with all information for testing purposes.
- **get_cell_info(self, y, x)**
  - Retrieves the information of the cell at position (y, x) from the temporary map.
- **copy(self)**
  - Creates and returns a deep copy of the Program object.

b. *Class Agent (...)*
Implementation file: Source/algorithms/agent.py

- The Agent class defines the behavior of an agent in the Wumpus World game. This agent will explore the map, gather information from the environment, make decisions based on existing knowledge, and perform actions such as moving, shooting, and using items.
- Attributes:
  - **x, y**: (int) Current position of the agent on the map (corresponding to the column and row).
  - **current_hp**: (int) Current health points of the agent (default is 100).
  - **map_size**: (int) Size of the map (default is 10x10).
  - **direction**: (tuple(int, int)) Direction in which the agent is facing (default is (1, 0)).
  - **knowledge_base_pit_percept**: (list) List of coordinates where the agent has detected breeze, indicating the potential presence of a pit.
  - **knowledge_base_wumpus_percept**: (list) List of coordinates where the agent has detected a stench, indicating the potential presence of a Wumpus.

- **knowledge_base_poison_percept**: (list) List of coordinates where the agent has detected a whiff, indicating the potential presence of poison gas.
- **knowledge_base_health_percept**: (list) List of coordinates where the agent has detected a glow, indicating the potential presence of a healing potion.
- **maybe_wumpus**: (list) List of coordinates that may contain a Wumpus.
- **maybe_poison**: (list) List of coordinates that may contain poison gas.
- **maybe_pit**: (list) List of coordinates that may contain a pit.
- **maybe_health**: (list) List of coordinates that may contain a healing potion.
- **sure_wumpus**: (list) List of coordinates that are confirmed to contain a Wumpus.
- **sure_poison**: (list) List of coordinates that are confirmed to contain poison gas.
- **sure_pit**: (list) List of coordinates that are confirmed to contain a pit.
- **sure_health**: (list) List of coordinates that are confirmed to contain a healing potion.
- **healing_potion**: (int) Number of healing potions the agent currently holds.
- **point**: (int) Current score of the agent.
- **path**: (list) Path taken by the agent during exploration of the map.
- **grab_heal**: (list) List of coordinates where the agent has collected a healing potion.
- **grab_gold**: (list) List of coordinates where the agent has collected gold.
- **heal**: (list) List of coordinates where the agent has used a healing potion.
- **shoot_act**: (list) List of shooting actions performed by the agent.

- Methods:
  - **__init__(self, map_size=10)**
    - Initializes the Agent object with default starting position at (0, 0) and other attributes.
  - **check_have_wumpus(self, y, x, cell=None)**
    - Checks if the location (y, x) potentially contains a Wumpus.
  - **check_have_pit(self, y, x, cell)**
    - Checks if the location (y, x) potentially contains a pit.
  - **check_have_poison(self, y, x, cell)**
    - Checks if the location (y, x) potentially contains poison gas.
  - **check_have_healing(self, y, x, cell)**

- Checks if the location (y, x) potentially contains a healing potion.

▪ **check_no_safe(self, y, x, cell)**
- Checks if the location (y, x) is safe (contains no pits or Wumpus).

▪ **dfs(self, program)**
- Performs Depth-First Search (DFS) to explore the map based on the agent's current knowledge.

▪ **go_to_shoot(self, i, program)**
- Determines the shortest path to reach a position where the agent can shoot a Wumpus.

▪ **shoot(self, ny, nx, program)**
- Executes the action of shooting a Wumpus at location (ny, nx)

▪ **shoot_process(self, program, graph)**
- Handles the shooting process, checking and confirming the Wumpus location, performing the shoot, and updating the map after the shooting.

IV. Experimental Description and Evaluation

1. The relationship between input text file and input in UI

The rows in the input file will be reversed upside-down in the program.

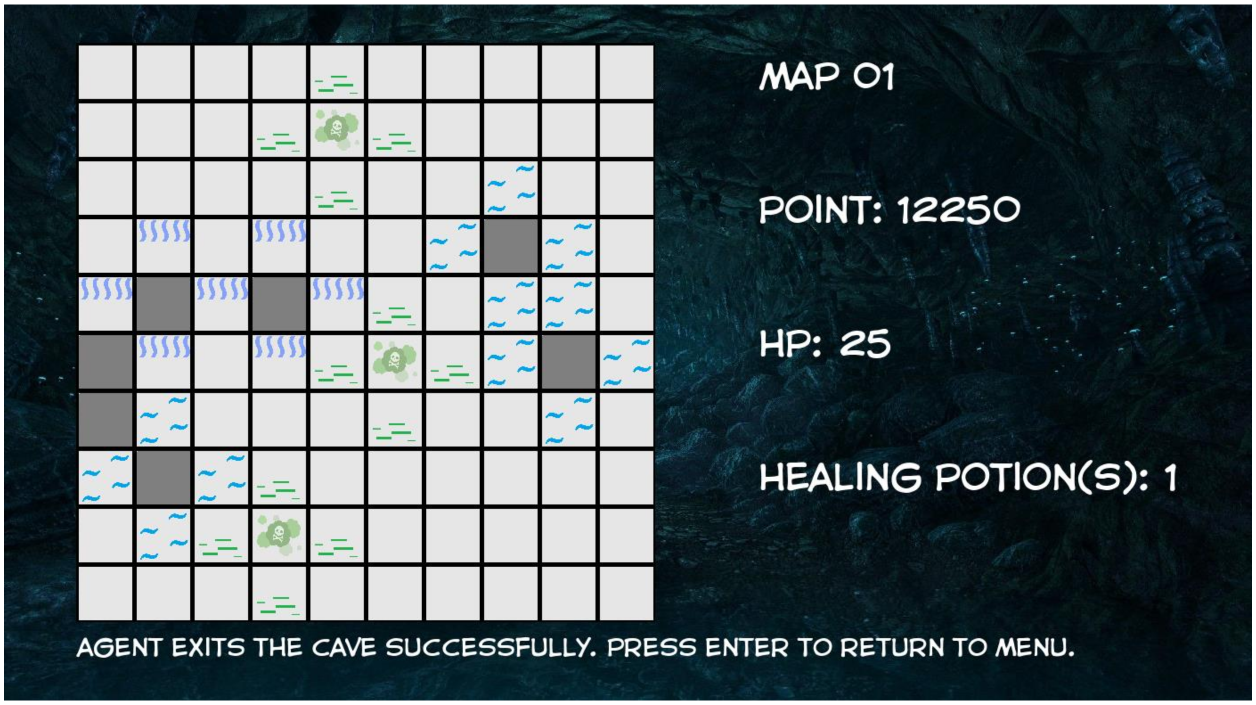
2. Experimental Description

Map 01:

Input Text File:

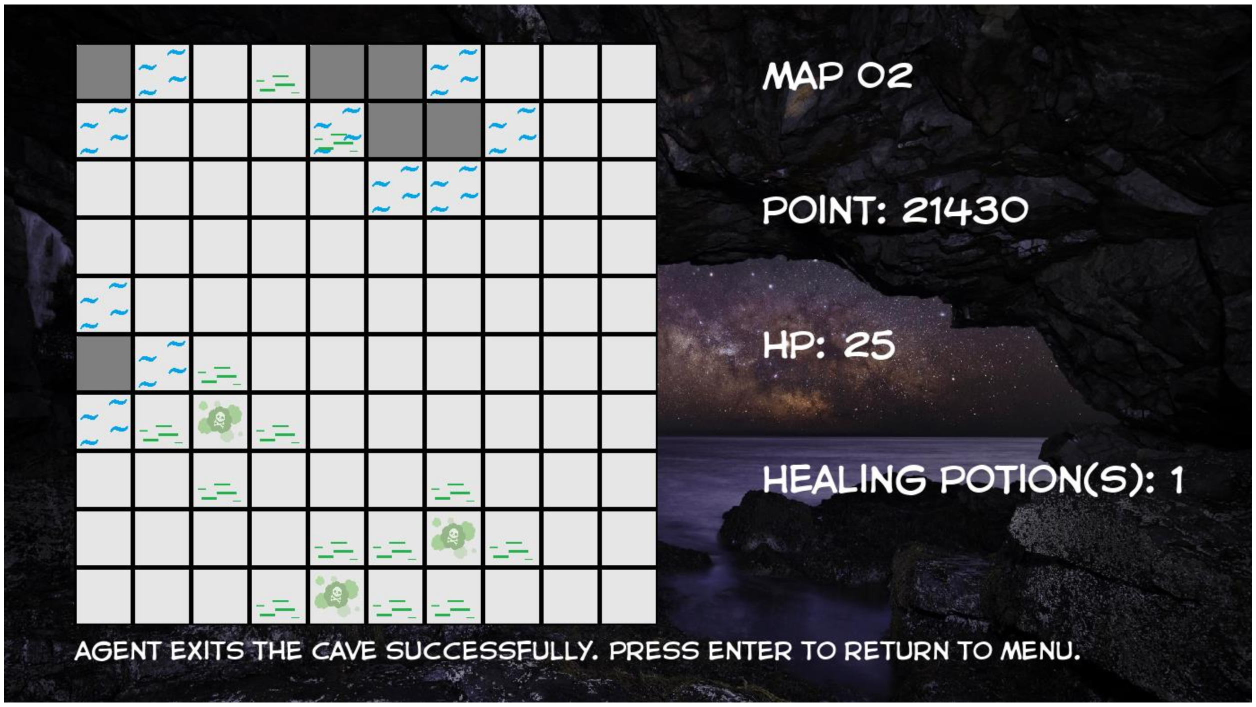
```
10
-.-.-.-.-.-.-.-.-.-
-.-.-.P_G.-.-.-.H_P.-.-
-.P.-.-.W.-.-.-.-.-
P.-.-.-.-.-.-.W.-.-
-.-.-.-.-.P_G.-.-.P.-
-.W.-.W.-.-.-.-.-.-
-.-.-.-.-.-.-.P.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.-.P_G.-.-.-.-.-
-.-.-.-.-.-.-.-.-.G
```

UI View:


MAP 01
POINT: 12250
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

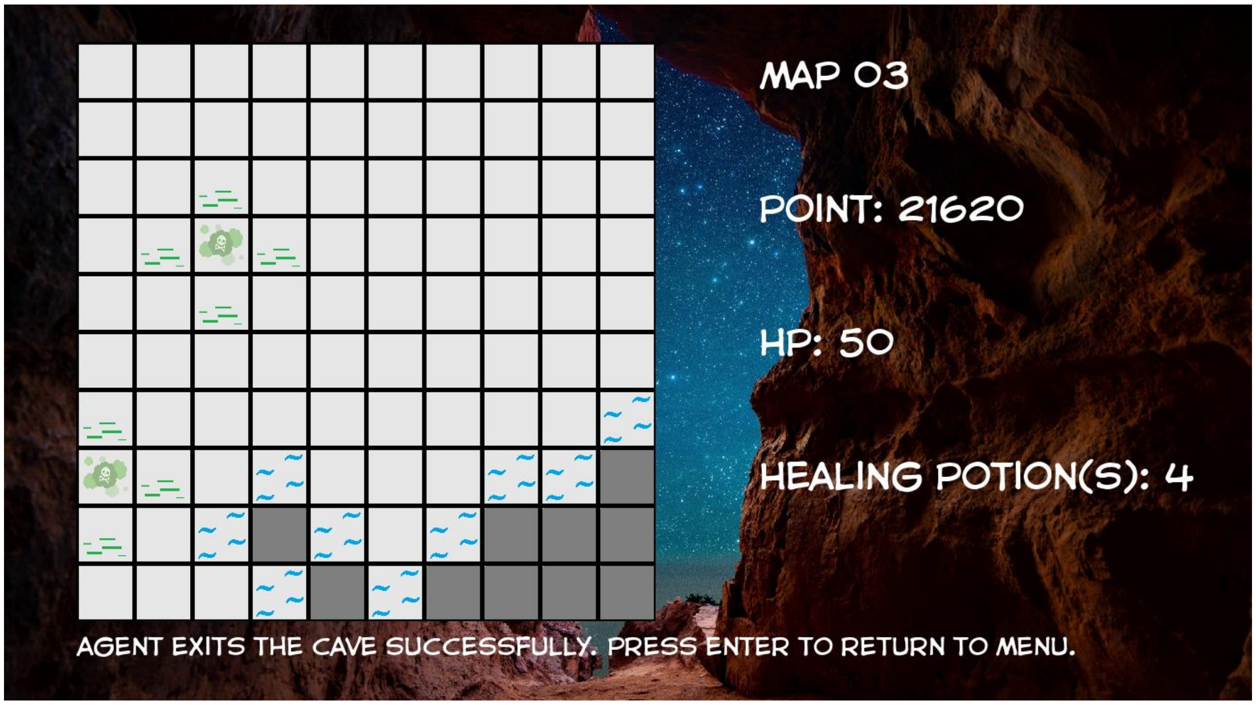
Description:
This map contains 4 Wumpus, 1 Gold chest and some other objects. The Agent only killed 2 Wumpuses instead of killing all of them because it could reduce point. The Agent also grabbed 1 potion but did not use it.

Map 02:
Input Text File:
```
10
-.-.-.-.P_G.-.-.-.-.-
-.-.H_P.-.-.-.P_G.-.-.-
-.-.-.-.-.-.W,W,W.-.-.-
-.-.P_G.-.-.-.-.-.-.-
P.-.-.-.G.-.-.-.-.-
-.-.-.-.-.-.W,W.-.-.-
-.-.G.-.-.-.-.-.-.-
-.-.-.-.G.-.-.-.-.-
-.-.W,W,W.-.-.P.P.-.-.-
P.-.-.-.P_G.-.-.-.-.-
```
UI View:


MAP 02
POINT: 21430
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

Description:
This map has 3 multi-Wumpus in the same position: 3 Wumpuses in (3, 7), 2 Wumpuses in (6, 7) and 3 Wumpuses in (9, 3). The Agent had to shoot many times to 1 position on the map to check whether the Wumpus(es) died. The Agent grabbed all 3 Gold chests on this map, so the final point was high. The Agent avoided moving to the fourth poisonous gas although he had gotten a potion.

Map 03:
Input Text File:
10
-.-.-.G.P.-.-.-.-.G
-.-.-.P.G.-.-.P.-.-
P_G,-,-.-.-.W.-.-.-.-.-.P
-.-.-.-.-.W.-.-.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.H_P.-
-.-.G,P_G.-.-.-.-.-.W.H_P
-.-.-.-.-.-.-.-.H_P.-
-.-.-.-.H_P.-.-.-.-.-
W.-.-.-.-.-.-.-.-.-
UI View:


MAP 03

POINT: 21620

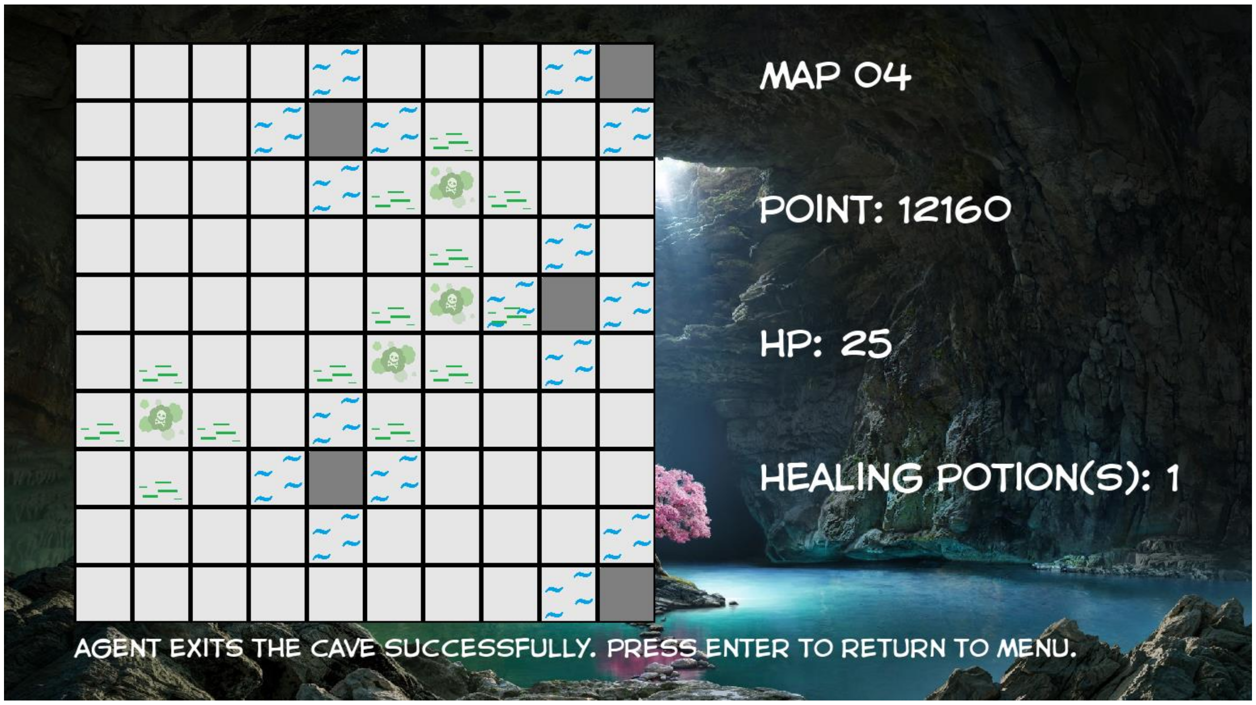
HP: 50

HEALING POTION(S): 4

AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

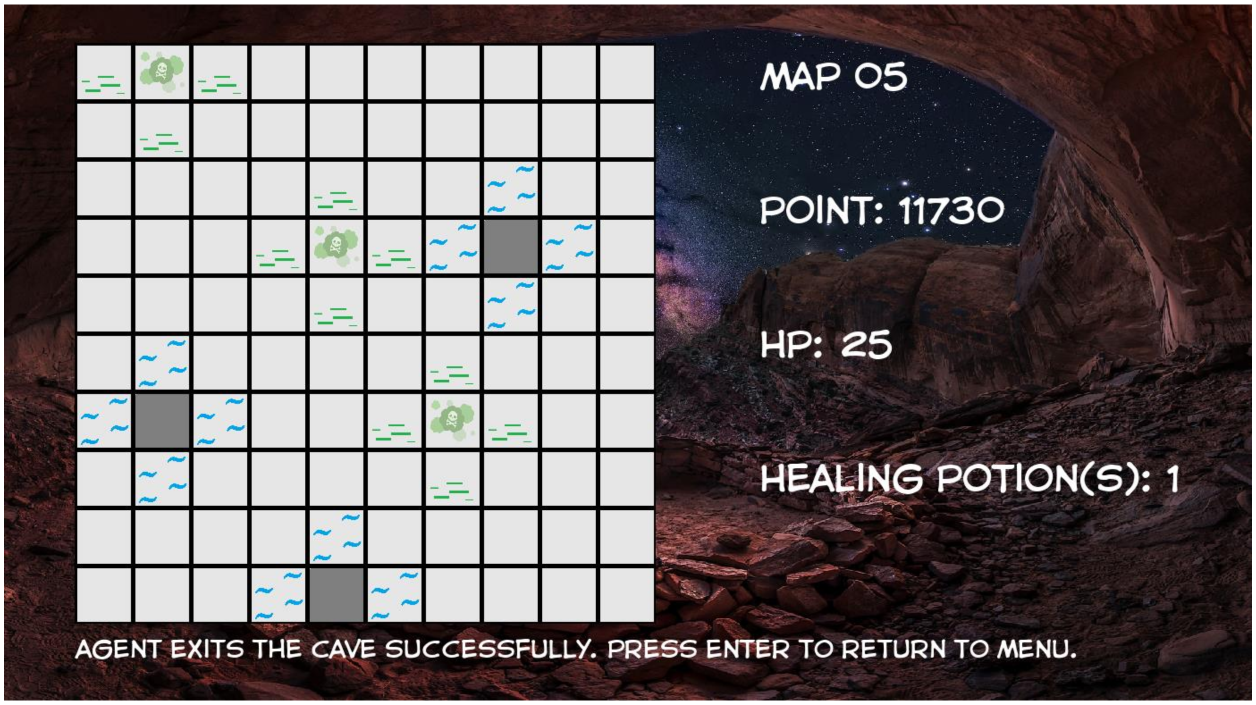
Description:
There are 5 pits in this map, they mainly locate at lower right corner of the map. The Agent only grabbed 3 Gold chests and left one because he was scared to fall in a pit.

Map 04:
Input Text File:
10
-.-.-.-.-.-.-.-.-.P
-.-.-.H_P.-.-.-.-.-
-.-.-.-.P.-.-.-.-.-
-.P_G.-.-.-.-.-.-.-
-.-.-.W.G.P_G.-.-.-.-
-.-.-.-.-.-.P_G.-.P.-
-.-.W.-.-.-.-.-.-.-
-.-.-.-.-.-.P_G.-.-.-
-.-.-.-.P.-.-.-.-.-
W.-.-.-.-.-.-.-.-.P
UI View:


MAP 04
POINT: 12160
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

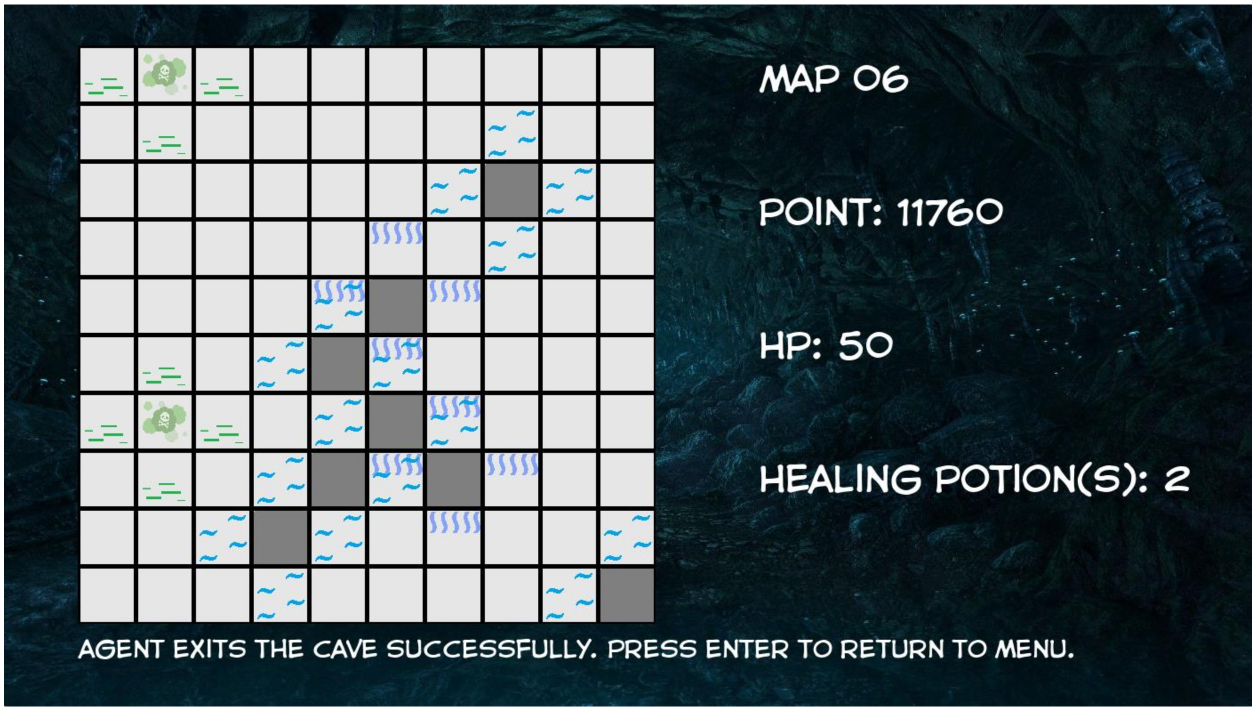
Description:
In map 4, there are many poisonous gas positions, pits and Wumpuses. The Agent had to avoid many unfavorable objects but still grabbed 1 Gold chest and exited the cave successfully.

Map 05:
Input Text File:
10
-.-.-.-.P.-.-.-.-.-
-.-.-.-.-.-.G.-.-.-
-.-.W.-.-.-.-.-.W.-
-.P.-.-.-.-.P_G.-.-.-
-.-.-.-.-.-.-.-.-.-
-.W.-.-.-.-.-.-.-.-
-.-.-.-.P_G.-.-.P.-.-
-.-.-.-.-.-.W.-.-.-
-.-.-.-.-.-.-.-.-.-
-.P_G.-.-.-.-.-.H_P.-.-
UI View:


MAP 05
POINT: 11730
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

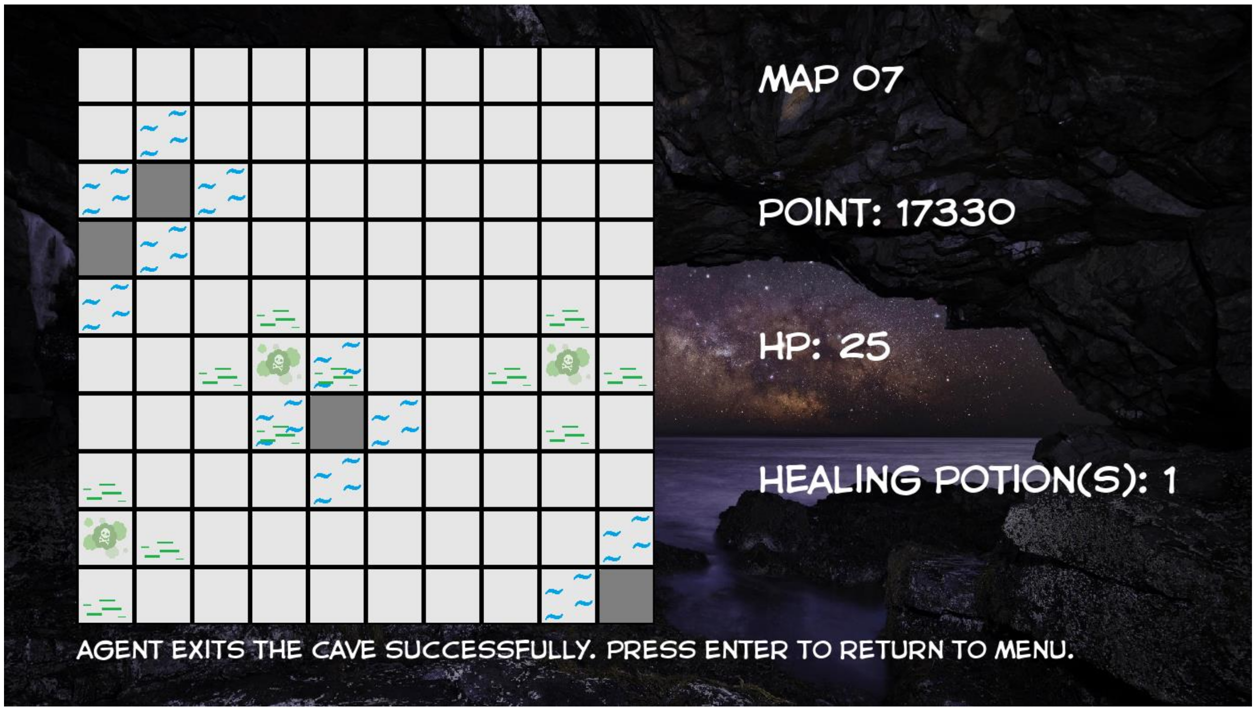
Description:
In map 5, there are 4 Wumpuses in 4 different positions. The Agent moved into the cave to grab items first. Then based on his knowledge, the Agent killed the Wumpuses and checked whether any items existed in those positions, but he found nothing. Finally, the Agent exited the cave.

Map 06:
Input Text File:
10
```
-.-.-.-.-.-.-.-.-.P
-.-.-.P.-.-.-.-.-.-
-.-.W.-.-.-.W.-.-.-
-.P_G.-.-.-.P.-.-.-.-
-.-.-.-.P.-.-.H_P.-.-
-.-.-.-.-.W.-.-.-.-
-.-.W.-.-.-.-.-.-.-
-.W.-.-.-.-.-.P.-.-
-.-.-.-.-.H_P.-.-.-.-
-.P_G.-.-.-.-.-.-.-.G
```
UI View:


MAP 06
POINT: 11760
HP: 50
HEALING POTION(S): 2
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

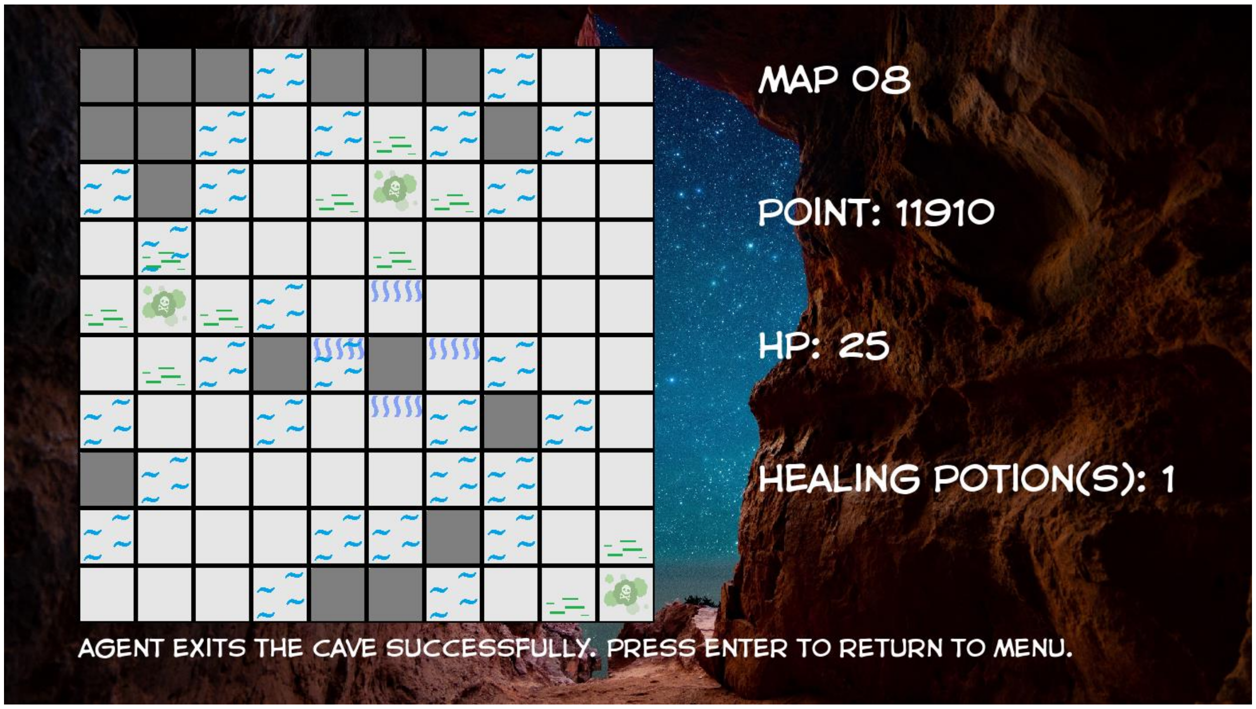
Description:
This map has the number of Wumpus which equals to that of Pit. Also, this map has 2 poisonous gases and 2 healing potions. Moreover, most of Wumpus and Pit is at the center of the map, so the agent must move around and can get poisoned, and he will face difficulties searching for the objects in the heart of the map.
This map contains a small number of Poisonous Gases, so the 2 Healing Potions are not used effectively.

Map 07:
Input Text File:
```
10
-.-.-.-.-.-.-.-.-.P
P_G.-.-.-.-.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.-.P.-.-.W.-.-
-.-.-.P_G.-.-.-.-.P_G.-
-.-.-.-.G.-.-.-.-.-
P.-.-.-.-.-.-.W.-.-
-.P.-.-.-.G.-.-.-.-
-.-.-.-.-.-.-.H_P.-.-
-.-.-.-.-.-.-.-.-.-
```
UI View:


MAP 07
POINT: 17330
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

Description:
The most significant difference of this map compared to the previous map is that it contains 2 Golds. And the dangerous objects are a little bit less than the previous one. This map only contains 2 Wumpus and 4 Pits. However, it has more Poisonous Gas than the previous map with 3 Poisonous Gases.
In this map, the agent can face a big difficulty due to the limitation of Healing Potion. The 7th map contains only one Healing Potion, but it has 3 Poisonous Gases. However, the agent can gain more points because there are 2 Golds for him in the map.

Map 08:
Input Text File:
10
-.-.-.-.P.-.-.-.-.P_G
-.-.-.-.-.-.P.-.-.-
P.-.-.-.-.W.-.-.-.-
-.-.W.-.W.-.-.P.-.-
-.-.-.P.-.W.-.-.-.-
-.P_G.-.-.G.-.-.-.-.-
-.-.-.-.-.-.-.-.-.-
-.P.-.-.-.P_G.-.-.-.-
-.-.-.-.-.-.-.P.-.-
-.-.P.-.P.-.-.-.H_P.-
UI View:

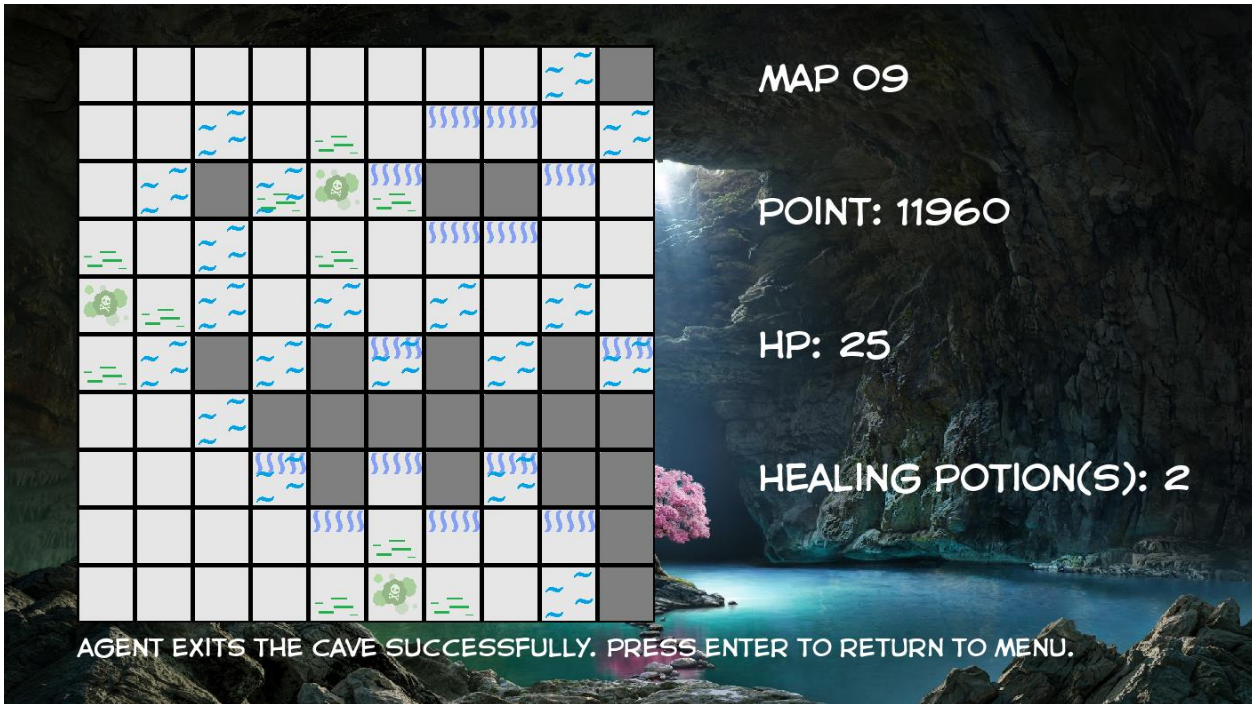

MAP 08
POINT: 11910
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

Description:
The 8th map contains many Pits with 9 of them, and it includes only 4 Wumpus near the heart of the map. The number of Poisonous Gases and Healing Potions are 3 and 1 respectively which are the same as the previous map. However, this map only contains one Gold, so the agent will be hard to gain the high score.
He also faces the large number of Poisonous Gases and avoids Pits with only one healing Potion.

Map 09:
Input Text File:

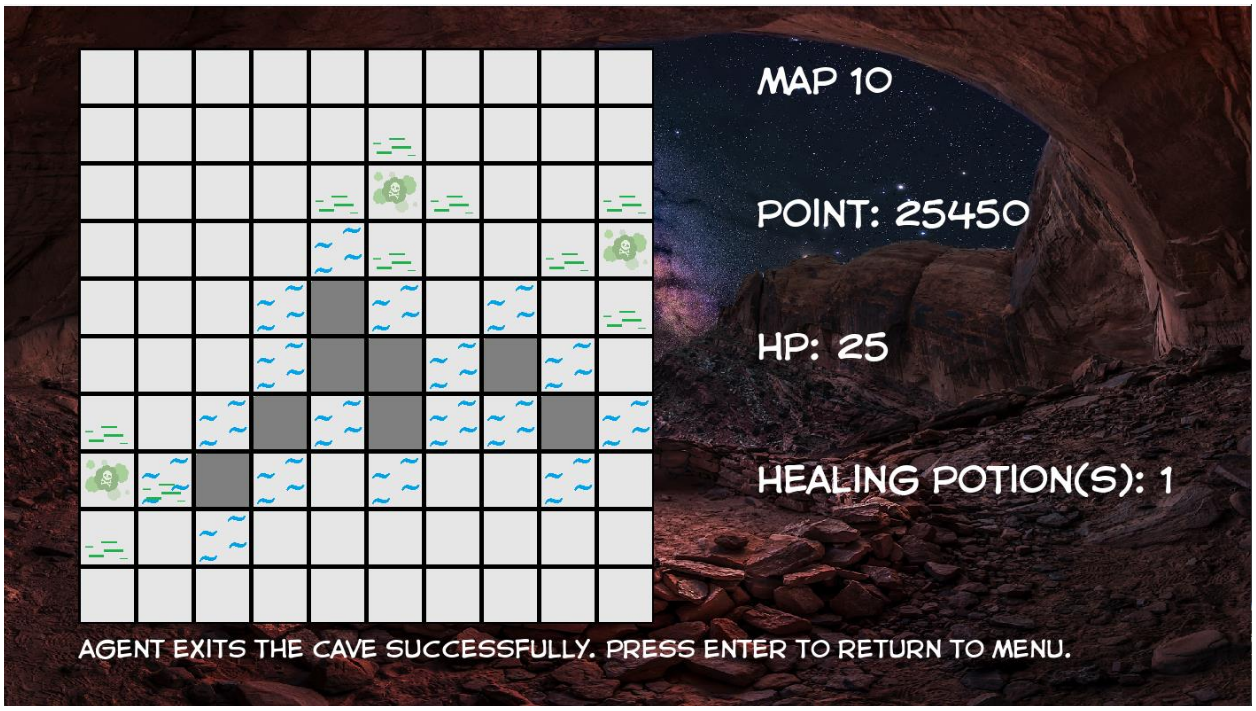
```
10
-.-.W.-.-.P_G.-.-.-.P
-.-.-.-.-.-.H_P.-.-.-
-.-.W.-.W.-.W.-.W.-
-.W.-.P.-.W.-.P.-.W
-.-.P.-.P.-.P.-.P.-
P_G.-.-.-.-.-.-.-.-.-
-.H_P.-.-.-.-.-.-.-.-
-.-.P.-.P_G.-.W.W.-.-
-.-.-.-.-.-.-.-.-.-
-.-.-.G.-.-.-.-.-.P
```

UI View:


MAP 09
POINT: 11960
HP: 25
HEALING POTION(S): 2
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

Description:
Next, on the 9th map, it appears lots of obstacles with 10 Wumpus and 9 Pits. Also, we can see that the 3rd line and the 4th line in the UI map has a fence of Wumpus and the 5th line contains lots of Pits. In this map, it has 3 Poisonous Gases, but this time, it contains 2 Healing Potions so that the agent may not run out of HP.
With lots of Pits and Wumpus appearing, it is hard for the agent to collect high scores because he must trade a certain number of points to shoot a Wumpus. And it also takes a lot of time to finish the map because of avoiding the Pits surrounding the map.

Map 10:
Input Text File:
10
-.-.-.G.-.-.-.-.-.-
-.-.-.-.-.-.W,W,W.-.-.-
P_G.-.P.-.W.-.-.-.-.-
-.-.-.-.-.P.-.-.P.-
-.H_P.W.-.P.-.-.P.-.-
-.-.G.-.P.-.-.-.G.-
-.-.W.-.-.-.-.-.-.P_G
-.-.-.-.-.P_G.-.-.-.-
-.-.W,W,W,W,W.-.-.-.-.-.-.-
-.W,W,W.-.-.-.-.-.-.G.-
UI View:


MAP 10
POINT: 25450
HP: 25
HEALING POTION(S): 1
AGENT EXITS THE CAVE SUCCESSFULLY. PRESS ENTER TO RETURN TO MENU.

Description:
Here, in the final map, we can see from the input file text that there are lots of lines that contain multiple Wumpus in a single room. The view of map UI does not show that clearly because the agent has finished this map.
With the large number of Wumpus in this map, the agent may find it hard to collect high scores. However, in this map, we have 4 Golds which will compensate for the lost points from shooting Wumpus.
Besides the large number of Wumpus, this map also has a great number of Pits with 6 of them; this can take the agent lots of time to avoid them and travel around the map.
In addition, in this final map, the agent must overcome this difficult test with 3 Poisonous Gases and only one Healing Potion.

3. Experimental Evaluation

Đối với các map mà có nhiều golds, wumpus, pit với các vị trí khác nhau (1 ô chỉ có một đối tượng) thì agent sẽ tốn nhiều thời gian và điểm số để tìm kiếm tối ưu các kho vàng.
Đối với các map mà có nhiều đối tượng cùng một ô như Wumpus thì agent sẽ phải thực hiện hành động nhiều lần tại một ô, tức là bắn mũi tên nhiều lần.
Đối với các map bị chặn nhiều bởi pit thì tần suất agent bị thua rất cao.
Đối với các map có nhiều golds thì agent hành động rất nhanh, tuy nhiên agent sẽ có xu hướng tìm các golds nào thuận tiện nhất thông qua A*.
Đối với map có nhiều poisonous gas thì agent bị mất máu nhiều thì số lần dùng healing potions tăng lên.

REFERENCES

During the process of researching and implementing **Project #2: Logical Agent – Wumpus World**, our team used and referenced some of the following open and electronic documents:

Programming

[1] *https://github.com/nrupeshsurya/Wumpus-world (last updated: 12/08/2024)*

[2]*https://www.transtutors.com/questions/in-python-wumpus-world-problem-the-agent-starts-with-an-initial-cash-amount-of-5000--9099956.htm (last updated: 12/08/2024)*

Report

[3] *https://www.geeksforgeeks.org/proposition-logic/ (last updated: 11/08/2024)*

[4] Slides of Introduction to Artificial Intelligence 2024 – Nguyễn Ngọc Thảo, Nguyễn Hải Minh *(last updated: 10/08/2024)*

In addition, the demo video about recording the running process of our program for each test case and our source code should be mentioned in this part:

Demo Video

[9] Youtube Video: link here

Source Code

[10] Github Link: link here