VIETNAM NATIONAL UNIVERSITY

HO CHI MINH UNIVERSITY OF SCIENCE

FACULTY INFORMATION TECHNOLOGY

PROJECT 02

WUMPUS WORLD

TEAM MEMBERS

Phạm Ngọc Thùy Trang 18127022

Võ Trần Quang Tuấn 18127248

Course: Introduction to Artificial Intelligence

Ho Chi Minh City – 2020

VIETNAM NATIONAL UNIVERSITY

HO CHI MINH UNIVERSITY OF SCIENCE

FACULTY INFORMATION TECHNOLOGY

PROJECT WUMPUS WORLD

| TOPIC |

| LECTURERS |

Mr. Le Ngoc Thanh

Ms. Ho Thi Thanh Tuyen

Ms. Nguyen Ngoc Thao

TABLE OF CONTENTS

REPORT'S TABLE OF CONTENTS

ACKNOWLEDGEMENTS

We would like to express our deepest appreciation to all those who provided me the possibility to complete this report. A special gratitude we give to our major Introduction to AI's teacher, Ms. Nguyen Ngoc Thao, whose contribution in stimulating suggestions and encouragement, helped us to coordinate my project especially in writing this report.

Furthermore we would also like to acknowledge with much appreciation the crucial role of the staff of Mr. Le Ngoc Thanh and Ms. Ho Thi Thanh Tuyen, who gave the permission to use all required equipment and the necessary material to complete the project "Searching".

We have to appreciate the guidance given by other supervisor as well as the panels especially in our project presentation that has improved our presentation skills thanks to their comment and advices.

Regards,

Team Representative,

Trang

Pham Ngoc Thuy Trang

ASSIGNMENT PLAN

- **Preparation**

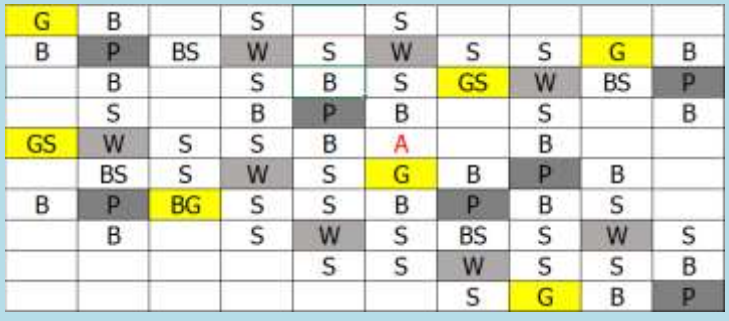
Sprites for Game

Name of sprites	
Agent	 **back**
	 **right**
	 **front**
	 **left**
Brick_Visible	
Brick_NotVisible	
Gold (Coin)	

Pit	
Wumpus	
Breeze	
Stench	
Arrow	

At least 5 maps with different structures such as position and number of Pit, Golds and Wumpus

ID	Map (Default with size: 10x10)
1	

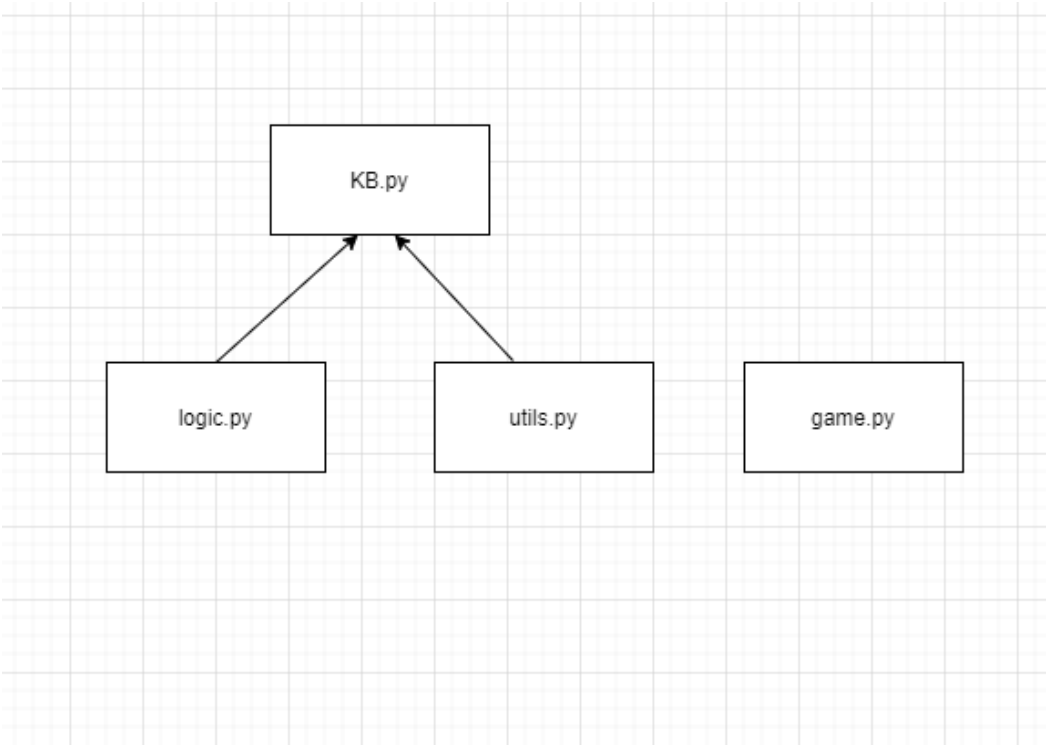
2	
3	
4	
5	

- **Ideas**

  Using propositional logic to represent the logical agent's knowledge base and using pygame to represent graphics

- **Planning UI and Workflow for game**

Workflow for game:



- **Explain some important classes and function.**

File logic.py: Important method in the classes in the logical file.py: is_valid (): check if the clause is valid or not, print_truth_table (): print the truth table, ...

Class Proposition:

It is used for propositional logic, which represents knowledge of KB passing through the cells in the game map. Example: P and Q will be represented as P & Q, or P or Q will be represented as P | Q, P deduce Q will be represented as P >> Q, P <==> Q will be represented as P << Q or P.iff (Q)

Class Constant: It is used for constant proposition

Class Variable: It is used for literals

Class Not, **Class And, Class Or, class Implies, class Iff:** It is used for connect literals together

Class ArgumentForm: It is used to deduce from a given sentence set, contains important methods such as is-valid(): check whether the clause is valid or not, the input parameter is the set of sentences in KB, concluding is a question to act

File utils.py:

def read_file(path): read data file to deal with data and test the KB classes

def find_near_cells(pos, height, width): From the current cell of the agent, this function is used to find the position of 4 cells around it

File KB.py: Knowledge demo for human and KB for human

Class KB: represent knowledge to human

KB.percept(self, state, height, width): Human receives information at the standing position, and predicts safe cells

KB .__ init __ (self, height, width, start, state): Initializing KB for human

generated thinking (curr_pos, state, height, width, visible_cells): Generating deductive statements from the experiencing state

KB.add_KB (self, state, height, width): After generating the inference statements, then add it to the KB

KB.inference(self, height, width, state): Deducing from standing position, marking and predicting safe cells

KB.show_KB (self): prints out all of KB's knowledge

KB.open_cell (self, meet_state): Opening current cell, adds perceived knowledge to KB such as cells with breeze, stench,…

KB.entail_query(self, query): Receiving the query and deduce whether the query is correct or not, to generate decisions for the agent

Class Human: Representing the agent logic in the game

human.move_toward_pos (self, pos, height, width): human moves to pos, perceives and generates action

human.move_toward_direction(self, direction = 'Right', height = 0, width = 0): the human moves in the present direction indicated, perceives where it comes from and engenders the action.

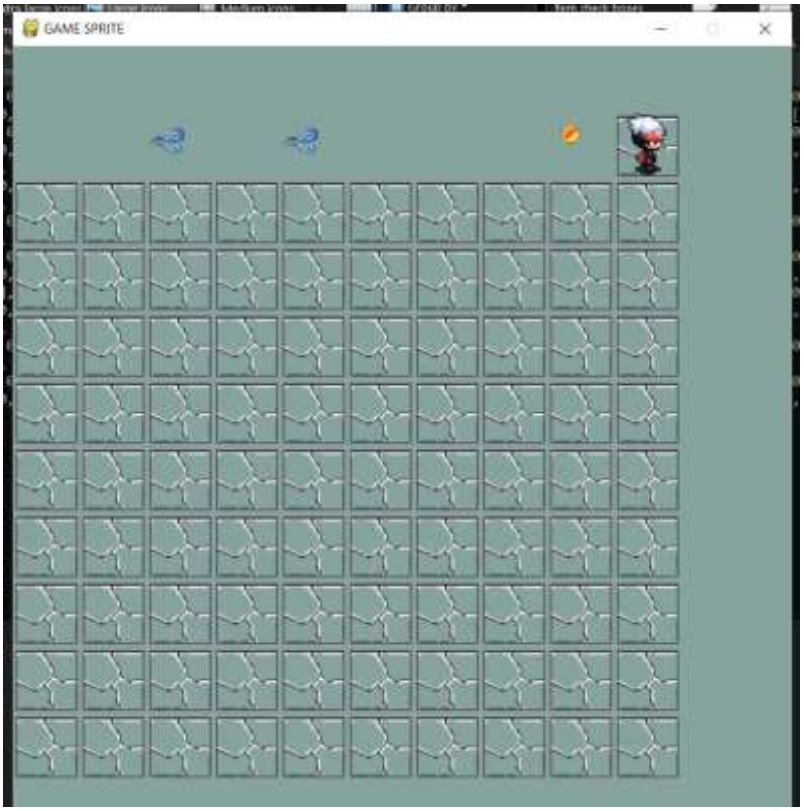
Files game.py: It is used to render the map as well as game sprites with positions corresponding to their positions when are read from the file. In addition, the file also pre-demonstrates how the agent will automatically move if the processing logic and data is successful.

Result (Console):

```
percept: [None, None, None], action: Left
¬B0 <=> (¬P1 ^ ¬P0 ^ ¬P0 ^ ¬P10)
¬S0 <=> (¬W1 ^ ¬W0 ^ ¬W0 ^ ¬W10)
¬B1 <=> (¬P2 ^ ¬P0 ^ ¬P1 ^ ¬P11)
¬S1 <=> (¬W2 ^ ¬W0 ^ ¬W1 ^ ¬W11)
percept: [None, None, None], action: Right
¬B0 <=> (¬P1 ^ ¬P0 ^ ¬P0 ^ ¬P10)
¬S0 <=> (¬W1 ^ ¬W0 ^ ¬W0 ^ ¬W10)
¬B1 <=> (¬P2 ^ ¬P0 ^ ¬P1 ^ ¬P11)
¬S1 <=> (¬W2 ^ ¬W0 ^ ¬W1 ^ ¬W11)
B2 <=> (P3 v P1 v P2 v P12)
percept: ['Breeze', None, None], action: Right
¬B0 <=> (¬P1 ^ ¬P0 ^ ¬P0 ^ ¬P10)
¬S0 <=> (¬W1 ^ ¬W0 ^ ¬W0 ^ ¬W10)
¬B1 <=> (¬P2 ^ ¬P0 ^ ¬P1 ^ ¬P11)
¬S1 <=> (¬W2 ^ ¬W0 ^ ¬W1 ^ ¬W11)
B2 <=> (P3 v P1 v P2 v P12)
¬B3 <=> (¬P4 ^ ¬P2 ^ ¬P3 ^ ¬P13)
¬S3 <=> (¬W4 ^ ¬W2 ^ ¬W3 ^ ¬W13)
percept: [None, None, None], action: Right
¬B0 <=> (¬P1 ^ ¬P0 ^ ¬P0 ^ ¬P10)
¬S0 <=> (¬W1 ^ ¬W0 ^ ¬W0 ^ ¬W10)
¬B1 <=> (¬P2 ^ ¬P0 ^ ¬P1 ^ ¬P11)
¬S1 <=> (¬W2 ^ ¬W0 ^ ¬W1 ^ ¬W11)
B2 <=> (P3 v P1 v P2 v P12)
¬B3 <=> (¬P4 ^ ¬P2 ^ ¬P3 ^ ¬P13)
¬S3 <=> (¬W4 ^ ¬W2 ^ ¬W3 ^ ¬W13)
B4 <=> (P5 v P3 v P4 v P14)
percept: ['Breeze', None, None], action: Right
¬B0 <=> (¬P1 ^ ¬P0 ^ ¬P0 ^ ¬P10)
¬S0 <=> (¬W1 ^ ¬W0 ^ ¬W0 ^ ¬W10)
¬B1 <=> (¬P2 ^ ¬P0 ^ ¬P1 ^ ¬P11)
¬S1 <=> (¬W2 ^ ¬W0 ^ ¬W1 ^ ¬W11)
B2 <=> (P3 v P1 v P2 v P12)
¬B3 <=> (¬P4 ^ ¬P2 ^ ¬P3 ^ ¬P13)
¬S3 <=> (¬W4 ^ ¬W2 ^ ¬W3 ^ ¬W13)
B4 <=> (P5 v P3 v P4 v P14)
¬B5 <=> (¬P6 ^ ¬P4 ^ ¬P5 ^ ¬P15)
¬S5 <=> (¬W6 ^ ¬W4 ^ ¬W5 ^ ¬W15)
```

UI for game (not complete)


GAME SPRITE
FINISHED !!!
SCORE: 0


GAME SPRITE

ENVIRONMENT

What we use for our project are:

- **IDEs:** Visual Code
- **Programming language:** Python
- **Outsource Platform for Python:** Anaconda
- **Libraries for Project:** PyQt5, pygame (for graphics), os
- **Version Control System**: Git (using GitHub to store projects and teamwork).
  - The primary branch is branch “master” and parallel to this branch is another branch called “dev”.
  - When the team’s source code in the “dev” branch reaches a stable point and is ready to be released, all of changes will be merged back into “master” branch.
  - Members will also have their own branch which is named after their name. These branches are used for edit/delete/update their functions….

ESTIMATE THE DEGREE OF COMPLETION LEVEL

OVERALL ESTIMATE

No.	**Specifications**	**Rating**	Issues
1	Finish problem successfully.	**3/5**	The logical agent haven't found the path to navigate the map yet.
2	Graphical demonstration of each step of the running process.	**0.5/1**	Cannot merge with the logic KB code to represent the problem
3	Generate at least 5 maps with difference structures such as position and number of Pit, Gold and Wumpus.	**2/2**	None
4	Report your algorithm, experiment with some reflection or comments.	**2/2**	None
Overall		**7.5/10**	

REFERENCES

[1]: Artificial Intelligence – A Modern Approach 3rd Edition Russel and Peter Norvig

[2]:_https://bitbucket.org/Andrew-Kay/dcaa/src/master/logic.py