



HERBERT

AI Project

Supervised by:

Dr. Islam ElShaarawy

Students:

Ibrahim Sa'eed Muhammad

Abanoub Wagih

Omar Mostafa

Main Idea

for us, it was really important to find the shortest path to fix the maze, regarding the obstacles and the boundary of the (25, 25) grid.

We used many components which would be explained later.

Project Link: [Here](#)

Input and output

input:

the number of the maze to be solved.

```
Please enter the level you want to solve: 3
```

output:

the string or actions, which applying on Hebert would solve the puzzle.

```
The best Solution of them is:  
lssrsslssrs  
with length of: 12
```

Tools

We Used Python 2.7, and PyCharm IDE as an environment for the project.

Used GitHub to share the Project.

To check the project please Visit this [link](#).

used libraries in the project:

- math
- copy
- heapq





Algorithms used:





- A* >> in finding the shortest path between two significant nodes.
- BFS (as in TSP) With random start point >> for getting the order of nodes visited.

Sections

Problems:

Containing the *****.txt files, which contain the problems in text.

 level1.txt
 level4.txt
 level7.txt
 level10.txt

 level2.txt
 level5.txt
 level8.txt
 level11.txt

 level3.txt
 level6.txt
 level9.txt
 level12.txt

DS.py:

Containing the non-functional or interactive classes used in the project.

Environment.py:

Containing the Environment variables and function.

script.py:

Containing the Herbert class, main, and UI functions.

Searching.py:

The A* Class and methods to perform the search in the map.

Abstracted look

First of all, we initiate the Environment,

```
def __init__(self, level):
```

which in turns call the function:

```
def read_problem(self, level):
```

given the level, then it returns the problem as a list of strings - Lines - the init function.

which converts the plain text to a map to work in later.

Then, we pass the environment variables to Herbert on initiating it and place it on the environment.

```
herbert = Herbert(initial_location, map,  
max_char, targets)  
environment.set_herbert(herbert)
```

then, Calling:

```
herbert.solve()
```

and let the magic begins!

In order to see more details, you can download the coder from the repository [Here](#), and read it.