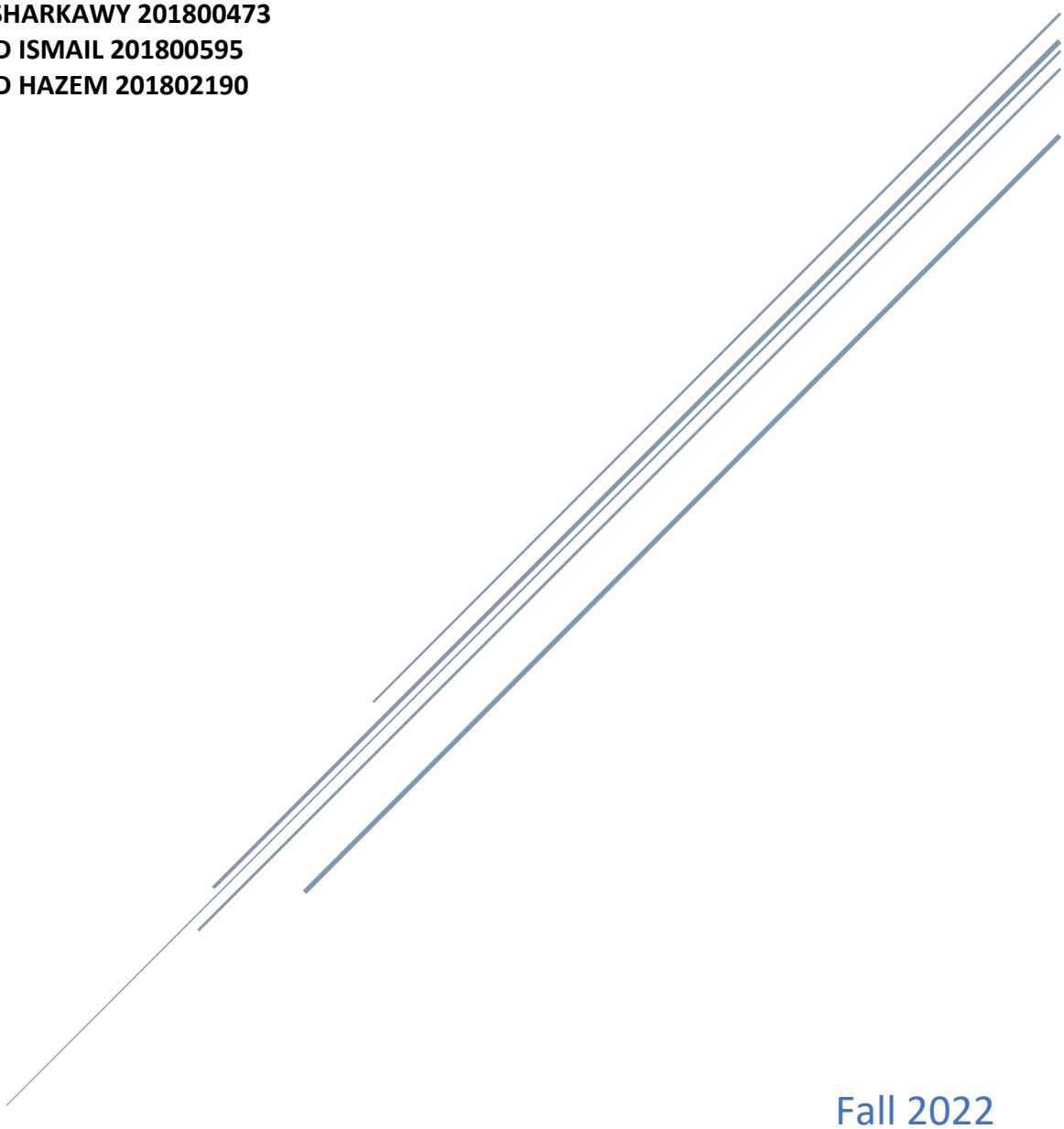
ZC CAMPUS PICK & DROP ROBOT

Project 2 – RL Q Learning

AI Team

ESLAM ELSHARKAWY 201800473
MOHAMED ISMAIL 201800595
MOHAMED HAZEM 201802190



Fall 2022

CIE 548: Artificial Intelligence

Table of Contents

Workload Distribution

Eslam El Sharkawy:

1. Problem Formulation and Modeling

Mohamed Ismail:

2. Environment Modelling
3. Apply Function

Mohamed Hazem:

4. Actions Function
5. Map GUI

Project Structure

- Single Python File that combines:
  1. Environment Class
  2. Map Visualization
  3. Q Learning Algorithm
  4. Actions/Apply Functions
  5. Test Lines (Explore - random - greedy)

Problem Formulation

In this project we need to model the problem of It needs to pick objects and deliver them from one location to another inside ZC with the minimum number of steps. The pickup locations are the NB and HB, meanwhile the drop off buildings are the AB and the One-stop-shop. Assuming that each pickup location contains 4 items to be delivered and that each delivery location can have at most 4 items.

We solved the problem using a maze approach by first, extracting multiple images for Zewail City from google maps, then we marked important landmarks and buildings and extracted x, y location as coordinates. These x, y values are scaled for fitting into our grid map.

States Representation:

- Zewail city has 6 gates and 8 buildings:

1- One-Stop Shop Building
2- Nano building
3- Helmy building
4- Service building
5- Workshops
6- Dorms
7- Administrative building
8- The Academic Building

- **Obstacles** were applied to model different real-world obstacles, to be points on the grid that the robot cannot access, therefore will stay in place if an action resulted to such points.

Initial & Goal States:

The aim is to start at Gate 1, then finally deliver all 8 items.

Actions and Successor function:

Four main Actions of [Up, Down, Left, Right], also the [Pick, Drop] actions which would be activated at the four building of interest.

Rewards:

For the main movement actions -> negative reward

And for [Pick, Drop] -> Maximum reward

Map Formulation

1. **Marking Important Buildings and path costs on Zewail City Map**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	0	0	0	0	0	0	0	0	0	0	0	G1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	G6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	G2
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	OS	0	0	0	0
9	0	0	0	0	0	0	0	He	0	0	0	0	0	0	0	0	0	Na	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	Ad	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	AC	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	G5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	G3
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	Se	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	Wo	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Do	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	G4	0	0	0	0	0	0	0	0	0	0	0	0

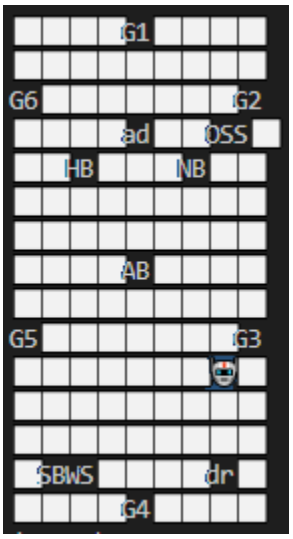
Final (coordinates) representation of the grid Map

Map Modelling Assumption:

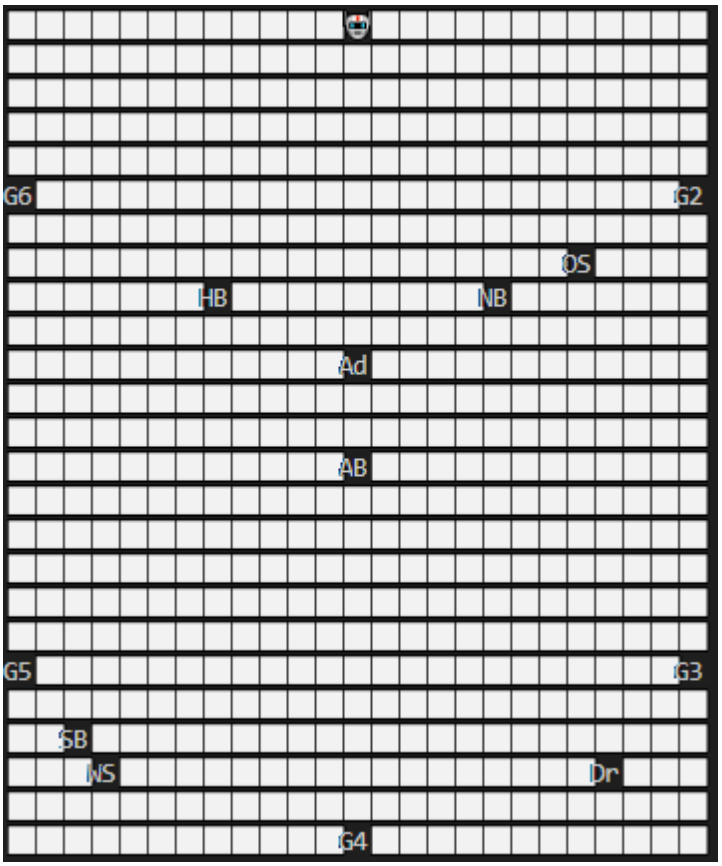
The city map was divided into a grid, with a side of 100 step.

- Symbol " " represents the robot.
- Every gate and building are represented by their initials.
    - G1 -> Gate 1, AB -> Academic Building

Map Grid:



Simple representation of the map (9*15)



Final representation of the map (25*25)

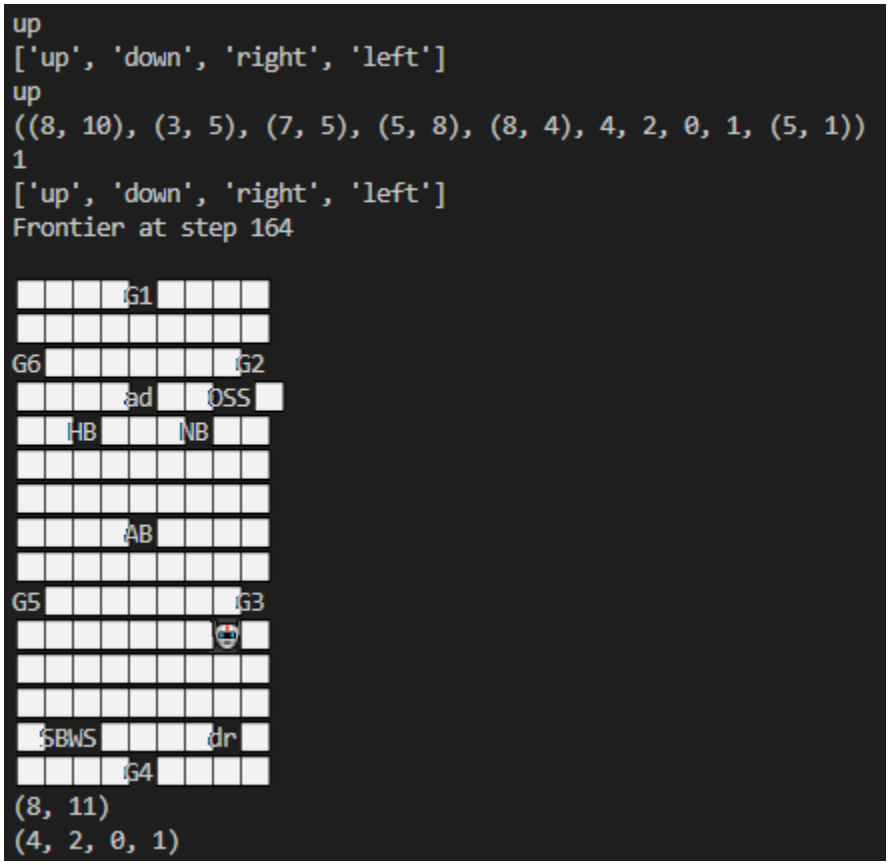
Case Study

You start the program by running the Python file.

Firstly, the program will run the exploring **Q-Learning Exploring algorithm**, then **greedy Algorithm** for testing and optimization.

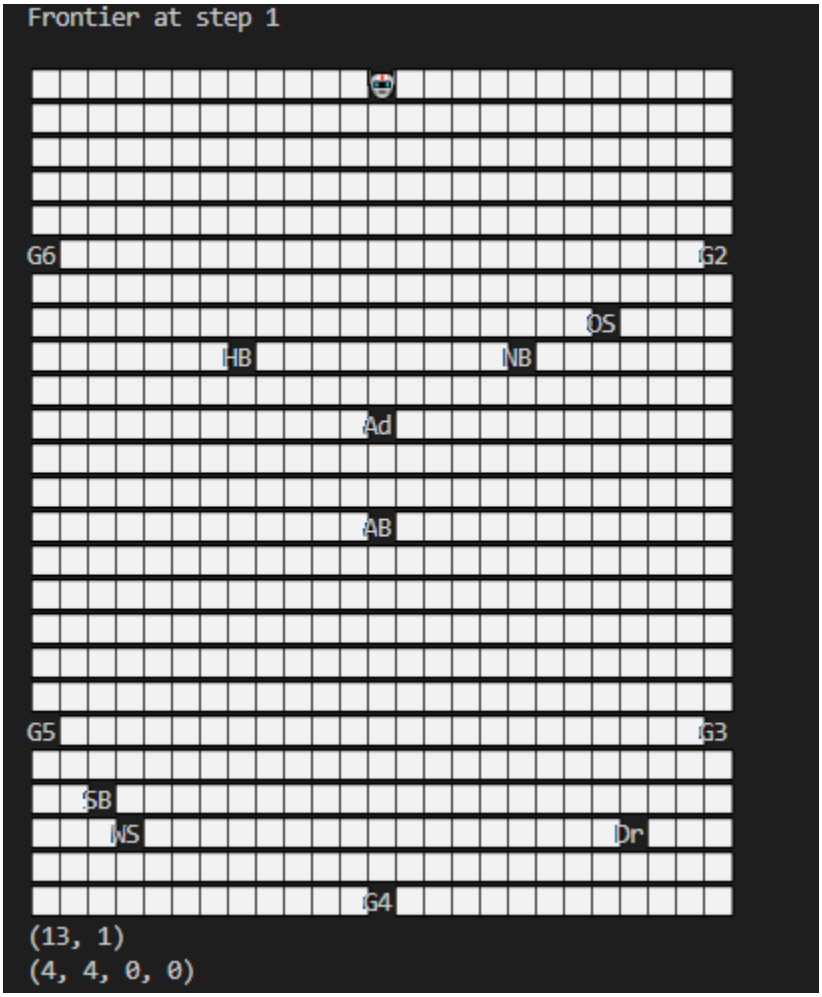
```
q, n = {}, {}
simulate(lambda: ZCRobot.new_random_instance(),duration=60, q=q, n=n, verbose=True,f=lambda q, n: 1/(n+1))
simulate(lambda:  ZCRobot.new_random_instance(), n_iterations=1, q=q, n=n, verbose=True, f=lambda q, n: q)
```

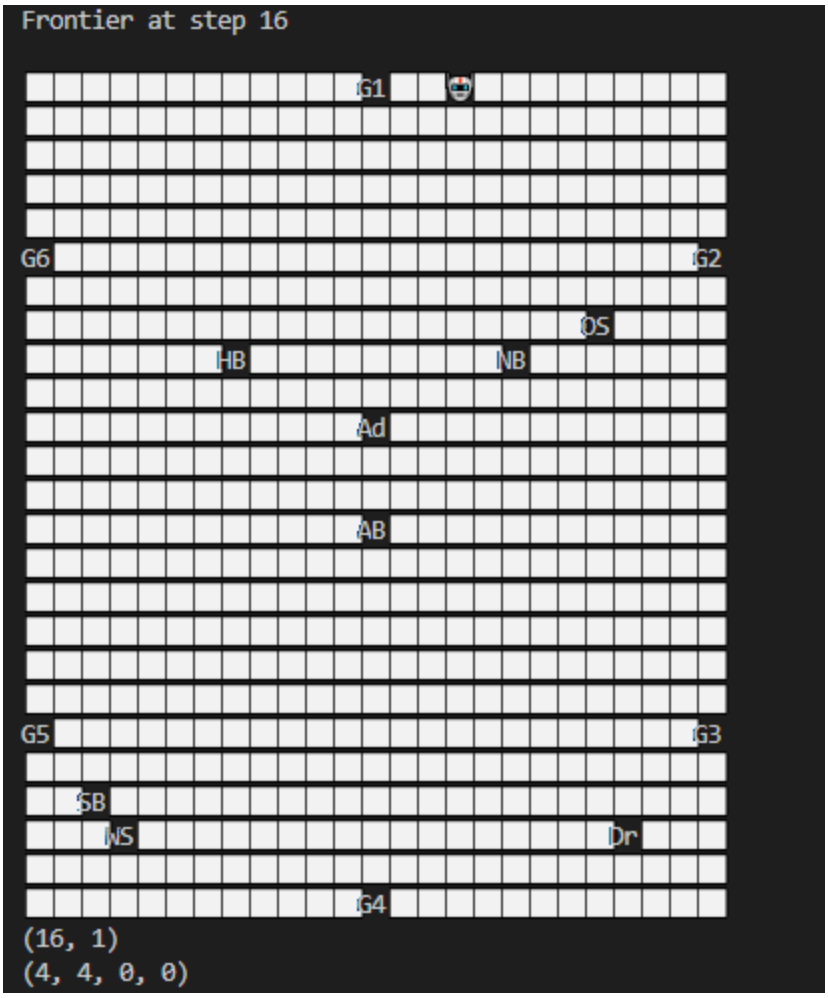
Final Output is similar to:

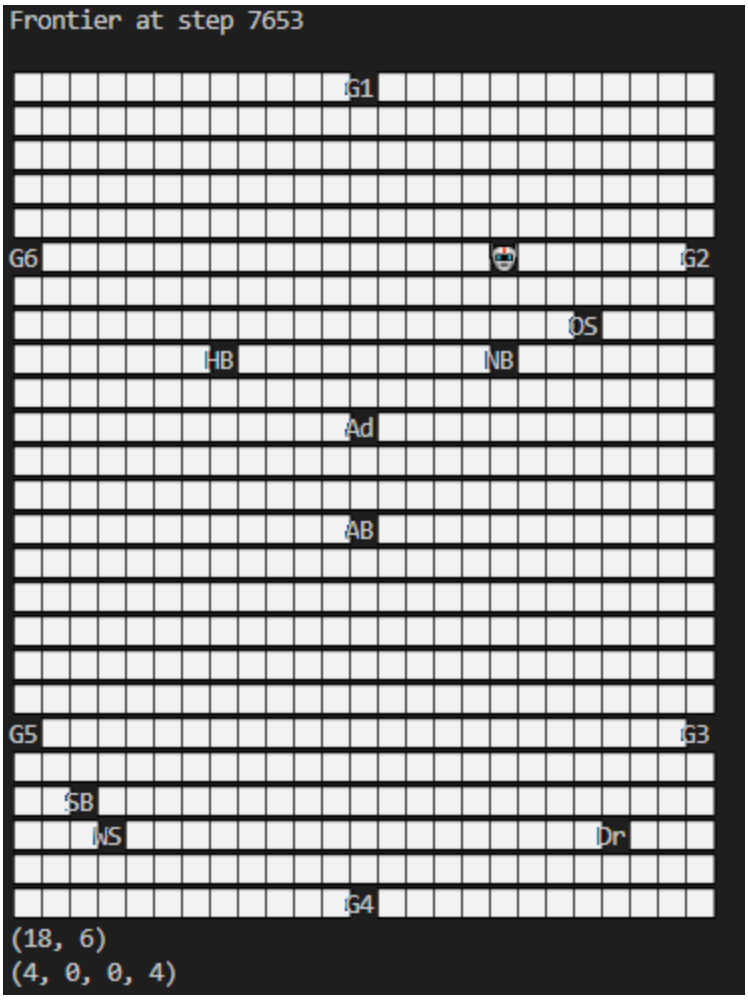


- Which have **the action** of that step, **coordinates of robot** current location and **number of items in every building** respectively (Helmy B, Nano B, Academic B, One-Stop B), in this case the robot is carrying one of the 8 items.

Results (Final Run)




Frontier at step 16
G1
G6
G2
OS
HB
NB
Ad
AB
G5
G3
SB
WS
Dr
G4
(16, 1)
(4, 4, 0, 0)


Frontier at step 7653
G1
G6
G2
OS
HB
NB
Ad
AB
G5
G3
SB
WS
Dr
G4
(18, 6)
(4, 0, 0, 4)

Q Matrix:

- For simplicity, a txt file is generated after every run with Q Matrix (States as rows, Actions as Columns)