Workshop

Computational Thinking and Programming 21/22

Silvio Peroni

Digital Humanities Advanced Research Centre (/DH.arc)
Department of Classical Philology and Italian Studies, University of Bologna, Bologna, Italy
silvio.peroni@unibo.it – @essepuntato – https://www.unibo.it/sitoweb/silvio.peroni/en

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
DEPARTMENT OF CLASSICAL PHILOLOGY
AND ITALIAN STUDIES

Escape the Tower Labyrinth

Picture “In Your Shadow 2016” by Hannes Flo, https://www.flickr.com/photos/hannesflo/24671722057/, CC-BY

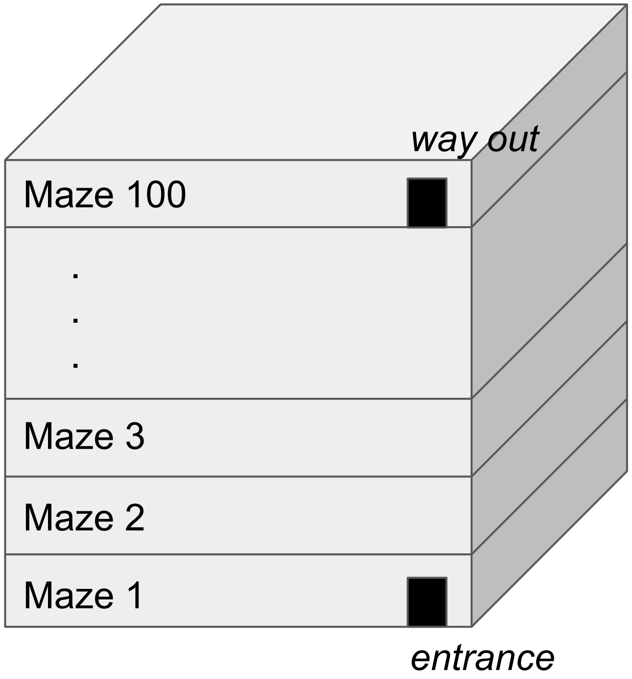
Plot

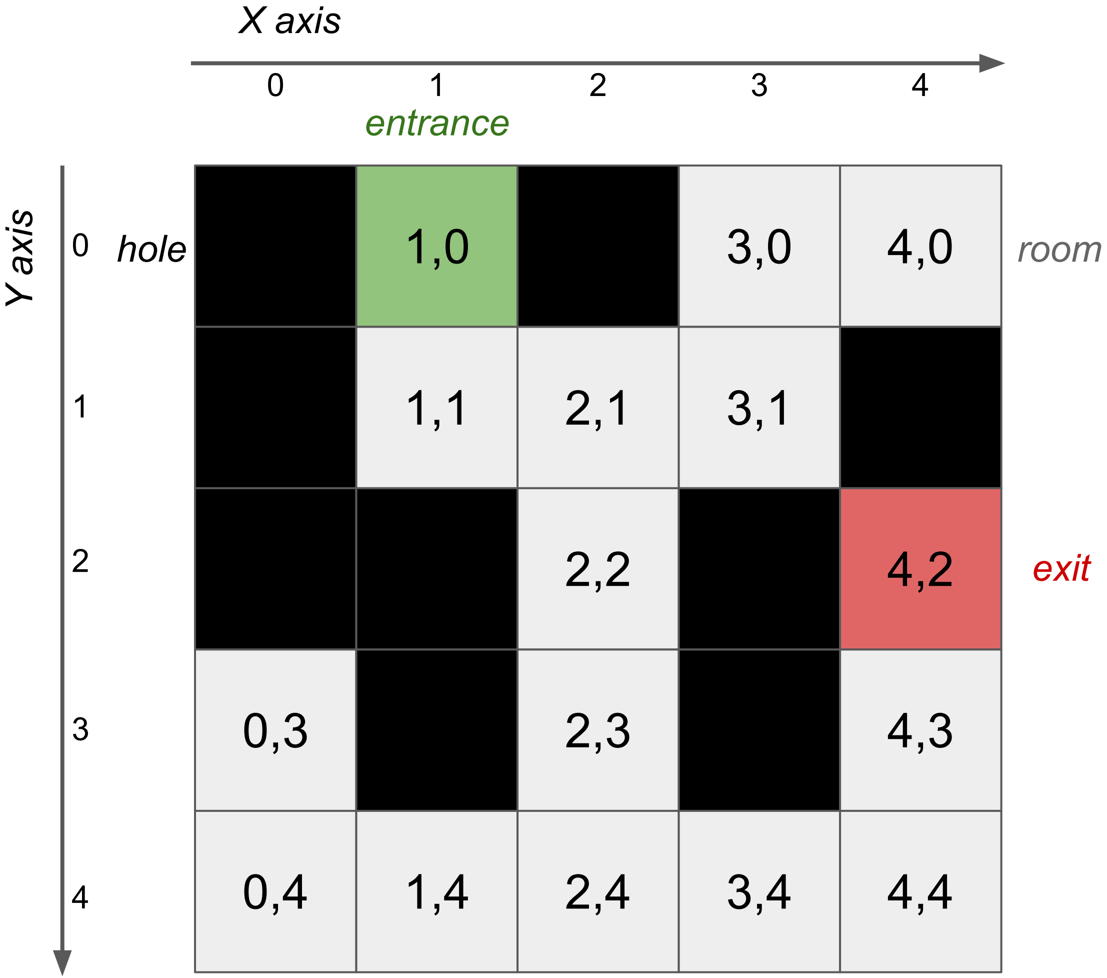
You, great explorers always looking for treasures around the globe, found a map of a mysterious place, containing the most ancient collections of the Library of Babel, that is reachable passing through a back door at the World End’s Inn

After talking with (and fighting against) several of the most marvellous creatures from the whole Universe, you arrived to the Inn, entered the door, and found just one item flourishing in the room you reached: the Book of Indefinite Pages, containing all the possible books ever written in just one portable volume

But then, Myntrakor, also known as Who Must Not Be Thought and ruler of that place, decided to complicate a bit your way out and pushed you in the darkest meta-dimension known to human beings, the Tower Labyrinth, made of 100 squared mazes of different dimensions placed one upon the other

Structure of a maze





	0	1	2	3	4
0	■	1,0	■	3,0	4,0
1	■	1,1	2,1	3,1	■
2	■	■	2,2	■	4,2
3	0,3	■	2,3	■	4,3
4	0,4	1,4	2,4	3,4	4,4

Rules

1. Each maze is organised within a square; it has one entrance in one side of the square and from one to three exits positioned in the other sides (at most one exit per side).
2. Each explorer can move horizontally or vertically, passing from one room to another.
3. All moves are organised in turns; every turn, each explorer uses its lamp to see all the adjacent rooms and choose to move to one of them; if the adjacent room reached does not contain mud, the explorer has the chance to move to another room in the same turn.
4. An explorer dies if (a) decides not to move, (b) moves to a place which is not an adjacent room (e.g. a hole, a non-adjacent room, or beyond the borders of the maze), (c) does not reach the exit within the number of turns set by Myntrakor for that maze
5. The maximum number of turns to find the exit of the maze is dependent on the the square defining the maze (4 turns in a 2x2 square, 9 in a 3x3, 14 in a 4x4, etc.).

Function to implement

```
def do_move(valid_moves, notebook)
```

It takes in input:

- **`valid_moves`**, a list of dictionaries each representing an adjacent room to move to
- **`notebook`**, a dictionary (empty in the first turn) available to an explorer where to keep notes about a particular maze

It returns a tuple of two items:

- the first item contains a tuple defining the X/Y coordinates of the next adjacent room to move to – e.g. **`(1,1)`**
- the second item contains the notebook that can be modified with additional information as a consequence of the choice of the move in the first item

Example of list of valid moves

```
[
    {   'x': 4,                 X coordinate of the adjacent room
        'y': 2,                 Y coordinate of the adjacent room
        'type': 'normal',       type of the adjacent room (either 'normal' or 'mud')
        'players': set(),       names (strings) of other players in the adjacent room
        'exit': 'no'            whether the adjacent room is an exit ('yes') or not ('no')
    },
    { … }
]
```

How it works

The file responsible for running the turns is **`labyrinth.py`**

The function **`do_move`** is included in the file **`group.py`** and will be called by the function **`play`** in **`labyrinth.py`** either once or twice per turn depending on the kind of the first adjacent room an explorer move to during a turn (either ‘normal’ or ‘mud’)

The file **`test_labyrinth.json`** contains an exemplar labyrinth to use

To run (and test) the implementation of **`do_move`** you can run

```
python labyrinth.py
```

Points to the groups

1 point will be assigned to all groups that developed a function that allowed their explorer to avoid non-permitted moves (i.e. either not to move or moving to a non-adjacent room)

1 point will be assigned to all groups that developed a function that allowed their explorer to find the exit of at least 30 mazes

1 point will be assigned to all groups that developed a function that allowed their explorer to find the exit of all the 100 mazes

1 point will be assigned to the group that won the greatest number of mazes (i.e. it reached an exit before the others)

Material and submission

All the files are available at
https://github.com/comp-think/2021-2022/tree/main/docs/workshop

At the end, each group must send an email to silvio.peroni@unibo.it where are specified:

- The name of the group
- The members of the group
- The file containing the implementation of **`do_move`** (rename it from **`.py`** to **`.txt`** before attaching it to the email)

End

Workshop

Computational Thinking and Programming 21/22

Silvio Peroni

Digital Humanities Advanced Research Centre (/DH.arc)
Department of Classical Philology and Italian Studies, University of Bologna, Bologna, Italy
silvio.peroni@unibo.it – @essepuntato – https://www.unibo.it/sitoweb/silvio.peroni/en

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
DEPARTMENT OF CLASSICAL PHILOLOGY
AND ITALIAN STUDIES