Project 1: N Doors Puzzle

CS 3343: Analysis of Algorithms Summer 2024

Instructor: Dr. Mohammad Imran Chowdhury

Total: 100 Points

Due: 06/11/2024 11:59 PM

In this project, I invite you to solve the ``N Doors Puzzle" using the various aspects of algorithmic thinking and Python programming discussed in class.

The ``N Doors Puzzle" is a great challenge to explore various aspects of algorithmic thinking and Python programming.

- Imagine you have N doors in a row that are all initially closed.
- You make N pass by each of these doors
- The first time through, visit every door and 'toggle' the door (if the door is closed, open it; if it is open, close it).
- The second time, only visit every 2nd door (i.e., door #2, #4, #6, …) and toggle it.
- The third time, visit every 3rd door (i.e., door #3, #6, #9, …), etc., until you only visit the Nth door on the Nth pass.

Implement your algorithmic thinking using Python programming to determine the state of the doors after the last pass. That is which doors are open and which are closed after the last pass?

Looking at the "N doors puzzle" we can make the following observations:

- There are doors numbered 1 through N that start closed. You can assume a list of

Boolean variables. Initially, they are all going to be set to False, with false representing a closed door and true representing an open door.

- We will iterate over the range N times, toggling the doors open/closed (represented by true/false)
- Each iteration over the range the gap between toggles increases by a multiple, incremented by one.
- On the last iteration, only the Nth door will be toggled

Sample I/O:

Run 1:

```
Enter the number of Doors (N): 10
i: 1; j:1; step size: 1. Toggling door number 1.
i: 1; j:2; step size: 1. Toggling door number 2.
i: 1; j:3; step size: 1. Toggling door number 3.
i: 1; j:4; step size: 1. Toggling door number 4.
i: 1; j:5; step size: 1. Toggling door number 5.
i: 1; j:6; step size: 1. Toggling door number 6.
i: 1; j:7; step size: 1. Toggling door number 7.
i: 1; j:8; step size: 1. Toggling door number 8.
i: 1; j:9; step size: 1. Toggling door number 9.
i: 1; j:10; step size: 1. Toggling door number 10.
i: 2; j:2; step size: 2. Toggling door number 2.
i: 2; j:4; step size: 2. Toggling door number 4.
i: 2; j:6; step size: 2. Toggling door number 6.
i: 2; j:8; step size: 2. Toggling door number 8.
i: 2; j:10; step size: 2. Toggling door number 10.
i: 3; j:3; step size: 3. Toggling door number 3.
i: 3; j:6; step size: 3. Toggling door number 6.
i: 3; j:9; step size: 3. Toggling door number 9.
i: 4; j:4; step size: 4. Toggling door number 4.
i: 4; j:8; step size: 4. Toggling door number 8.
i: 5; j:5; step size: 5. Toggling door number 5.
i: 5; j:10; step size: 5. Toggling door number 10.
i: 6; j:6; step size: 6. Toggling door number 6.
i: 7; j:7; step size: 7. Toggling door number 7.
i: 8; j:8; step size: 8. Toggling door number 8.
i: 9; j:9; step size: 9. Toggling door number 9.
i: 10; j:10; step size: 10. Toggling door number 10.
Algorithm has finished.

Door number 1 remains open.
Door number 4 remains open.
Door number 9 remains open.
Door number 2 remains closed.
Door number 3 remains closed.
Door number 5 remains closed.
Door number 6 remains closed.
Door number 7 remains closed.
Door number 8 remains closed.
Door number 10 remains closed.
```

Run 2:

```
Enter the number of Doors (N): 20
i: 1; j:1; step size: 1. Toggling door number 1.
i: 1; j:2; step size: 1. Toggling door number 2.
i: 1; j:3; step size: 1. Toggling door number 3.
i: 1; j:4; step size: 1. Toggling door number 4.
i: 1; j:5; step size: 1. Toggling door number 5.
i: 1; j:6; step size: 1. Toggling door number 6.
i: 1; j:7; step size: 1. Toggling door number 7.
i: 1; j:8; step size: 1. Toggling door number 8.
i: 1; j:9; step size: 1. Toggling door number 9.
i: 1; j:10; step size: 1. Toggling door number 10.
i: 1; j:11; step size: 1. Toggling door number 11.
i: 1; j:12; step size: 1. Toggling door number 12.
i: 1; j:13; step size: 1. Toggling door number 13.
i: 1; j:14; step size: 1. Toggling door number 14.
i: 1; j:15; step size: 1. Toggling door number 15.
i: 1; j:16; step size: 1. Toggling door number 16.
i: 1; j:17; step size: 1. Toggling door number 17.
i: 1; j:18; step size: 1. Toggling door number 18.
i: 1; j:19; step size: 1. Toggling door number 19.
i: 1; j:20; step size: 1. Toggling door number 20.
i: 2; j:2; step size: 2. Toggling door number 2.
i: 2; j:4; step size: 2. Toggling door number 4.
i: 2; j:6; step size: 2. Toggling door number 6.
i: 2; j:8; step size: 2. Toggling door number 8.
i: 2; j:10; step size: 2. Toggling door number 10.
i: 2; j:12; step size: 2. Toggling door number 12.
i: 2; j:14; step size: 2. Toggling door number 14.
i: 2; j:16; step size: 2. Toggling door number 16.
i: 2; j:18; step size: 2. Toggling door number 18.
i: 2; j:20; step size: 2. Toggling door number 20.
i: 3; j:3; step size: 3. Toggling door number 3.
i: 3; j:6; step size: 3. Toggling door number 6.
i: 3; j:9; step size: 3. Toggling door number 9.
i: 3; j:12; step size: 3. Toggling door number 12.
i: 3; j:15; step size: 3. Toggling door number 15.
```

```
i: 3; j:18; step size: 3. Toggling door number 18.
i: 4; j:4; step size: 4. Toggling door number 4.
i: 4; j:8; step size: 4. Toggling door number 8.
i: 4; j:12; step size: 4. Toggling door number 12.
i: 4; j:16; step size: 4. Toggling door number 16.
i: 4; j:20; step size: 4. Toggling door number 20.
i: 5; j:5; step size: 5. Toggling door number 5.
i: 5; j:10; step size: 5. Toggling door number 10.
i: 5; j:15; step size: 5. Toggling door number 15.
i: 5; j:20; step size: 5. Toggling door number 20.
i: 6; j:6; step size: 6. Toggling door number 6.
i: 6; j:12; step size: 6. Toggling door number 12.
i: 6; j:18; step size: 6. Toggling door number 18.
i: 7; j:7; step size: 7. Toggling door number 7.
i: 7; j:14; step size: 7. Toggling door number 14.
i: 8; j:8; step size: 8. Toggling door number 8.
i: 8; j:16; step size: 8. Toggling door number 16.
i: 9; j:9; step size: 9. Toggling door number 9.
i: 9; j:18; step size: 9. Toggling door number 18.
i: 10; j:10; step size: 10. Toggling door number 10.
i: 10; j:20; step size: 10. Toggling door number 20.
i: 11; j:11; step size: 11. Toggling door number 11.
i: 12; j:12; step size: 12. Toggling door number 12.
i: 13; j:13; step size: 13. Toggling door number 13.
i: 14; j:14; step size: 14. Toggling door number 14.
i: 15; j:15; step size: 15. Toggling door number 15.
i: 16; j:16; step size: 16. Toggling door number 16.
i: 17; j:17; step size: 17. Toggling door number 17.
i: 18; j:18; step size: 18. Toggling door number 18.
i: 19; j:19; step size: 19. Toggling door number 19.
i: 20; j:20; step size: 20. Toggling door number 20.
Algorithm has finished.

Door number 1 remains open.
Door number 4 remains open.
Door number 9 remains open.
Door number 16 remains open.


Door number 2 remains closed.
Door number 3 remains closed.
Door number 5 remains closed.
Door number 6 remains closed.
Door number 7 remains closed.
Door number 8 remains closed.
Door number 10 remains closed.
Door number 11 remains closed.
Door number 12 remains closed.
Door number 13 remains closed.
Door number 14 remains closed.
Door number 15 remains closed.
Door number 17 remains closed.
Door number 18 remains closed.
Door number 19 remains closed.
Door number 20 remains closed.
```

Fizz Buzz

Bonus Part (25 points):

For all the open doors, i, after the last pass i.e., Nth pass, return a list where:
doors[i] == "FizzBuzz" if i is divisible by 3 and 5.
doors[i] == "Fizz" if i is divisible by 3.
doors[i] == "Buzz" if i is divisible by 5.
doors[i] == i if none of the above conditions are true.

Run 1: When N = 20,
N Doors Puzzle's Fizz Buzz Implementation: [1, 4, 'fizz', 16]
Run 2: When N = 100,
N Doors Puzzle's Fizz Buzz Implementation: [1, 4, 'fizz', 16, 'buzz', 'fizz', 49, 64, 'fizz', 'buzz']
Run 3: When N = 257,
N Doors Puzzle's Fizz Buzz Implementation: [1, 4, 'fizz', 16, 'buzz', 'fizz', 49, 64, 'fizz', 'buzz', 121, 'fizz', 169, 196, 'fizzbuzz', 256]

The submission grading rubric is as follows (points out of 100 total, max. 125 including Bonus):

Project element	**Points**
Code readability such as the usage of comments in code	10
Proper coding implementation	60
Screenshots of the program output	10
Program's sample Input/Output format matching as per project writeup	20
Bonus part	25

Submission Instructions: Create a compressed file (.zip or .tar.gz files are accepted) with all your source files such as .py files. Within this compressed .zip folder, you should provide some screenshots of your running program's output as proof. Generally speaking, to complete the N Doors puzzle problem including the bonus part, you just need one .py file. But it's better to submit everything as a compressed file. Submit the compressed file to Canvas.

Late submission policy: As described in the syllabus, any late submission will the penalized with 10% off after each 24 hours late. For example, an assignment worth 100 points turned in 2 days late will receive a 20-point penalty. Assignments turned in 5 or more days after the due date will receive a grade of 0.