Programming with Python (COMP4008-PRG) 2nd exercise

Thorsten Altenkirch and Isaac Triguero

October 21, 2021

The guessing game

You are asked to implement a Python program that plays the *number guessing game*. The user thinks of a number within a given interval (e.g. [1, 100], including both extremes of the interval), but does not input anything to the program; Your program should ask questions to the user to know whether the number is less than, equal or greater than another number.

The goal is to guess the answer with as few steps as possible. Here is how a run of the program **must** look like:

```
Think of a number between 1 and 100!
Is your number greater (>), equal (=), or less (<) than 50?
Please answer <,=, or >! >
Is your number greater (>), equal (=), or less (<) than 75?
Please answer <,=, or >! <
Is your number greater (>), equal (=), or less (<) than 62?
Please answer <,=, or >! <
Is your number greater (>), equal (=), or less (<) than 56?
Please answer <,=, or >! >
Is your number greater (>), equal (=), or less (<) than 59?
Please answer <,=, or >! >
Is your number greater (>), equal (=), or less (<) than 60?
Please answer <,=, or >! =
I have guessed it!
I needed 6 steps!
```

To get full marks, please take the following points into consideration:

- Your program should handle incorrect inputs by printing an error message and asking the user to answer again.
- The program should require a minimal number of steps.
- The program should spot if the information the user provided is inconsistent (i.e. the user is lying). The program should finish once it detects an inconsistency/lie.

- The program should behave exactly as in the example above, i.e. it should play the game only once and use the same interface.
- The code should be as simple as possible and easily understandable, avoiding unnecessary code repetition.
- Comments should be used where appropriate.
- It should be straightforward to modify the code to use a different interval, e.g. -10 and +10.

You should use a Python IDE to complete this coursework (e.g. Spyder), (don't use jupyter notebook). Once you have completed your program, name your code as "ex02.py" and submit it on Moodle. Then, to get marked, you need to show your code to one of the demonstrators and answer any questions about it before the *demonstration deadline.*

Important notes:

- Use only the operations that have been introduced in the lectures.
- Optionally, if you finish and submit before the deadline, you can demo your code to one of the demonstrators in the lab and they will give you feedback
- Doing or not doing the demo does not affect your mark.
- We will not give you the marks immediately during the demo.
- Make sure you note down the name of the demonstrator if you do the demo. *You cannot resubmit your solution after demoing it to us.*
- You may be (randomly) selected to demo your solution the week after the deadline. Lab demonstrators will contact you to ensure you attend the lab session. *Being unable to explain your solution may affect your mark.*
- **Submission deadline: 9th November 2021**.