p4p exam @ 2023/01/27

Jani Kotakoski <jani.kotakoski@univie.ac.at>

Using Python3 and object-oriented programming, implement the game Othello (also known as Reversi).

First, you should create a directory and a virtual environment for the exam:

```
$ mkdir -p ~/p4p/exam_20230127
$ cd ~/p4p/exam_20230127
$ python3 -m venv ~/p4p/exam_20230127
$ source bin/activate
```

After this you should install numpy, and can clone the repository from the server (or download and unzip the file from moodle).

```
$ pip3 install numpy
$ git clone ssh://p4p/~/depot/othello.git
```

Now you should have a new directory, `othello`, where you find some files for the exam. The git repository (and the zip file on moodle) has the following directory structure:

```
.
+-- AUTHOR
+-- bin
+-- LICENSE
+-- README.md
+-- requirements.txt
+-- setup.py
+-- src
    +-- othgame
    |   +-- board.py
    |   +-- __init__.py
    +-- tests
        +-- __init__.py

4 directories, 8 files
```

Most of the files are empty, and you need to fill them in during the exam.

File `othello` or othello.py serves as the script running the game. It should either be in directory `bin` or `src` depending on your implementation. The program should be run via command line with:

```
$ python3 othello.py
```

or

```
$ ./othello
```

In directory `src/othgame/` needs to be three classes, `Game`, `Board`, and `Player`, each implemented in its own file (`game.py`, `board.py`, and `player`). You can of course also create additional classes, if needed.

While working on the exam, and especially at the end, remember to commit your changes and push your solution to the server.

You can always check with `git status` whether you have files with changes that should be added, committed, and pushed.

Othello

Othello is a two-player game that is played on a 8 × 8 checker board with discs that are white on one side and black on the other side. One player plays black, the other one white. In the beginning, four discs are placed in the middle as shown below.

```
O = white, @ = black

     1   2   3   4   5   6   7   8
   +---+---+---+---+---+---+---+---+
A  |   |   |   |   |   |   |   |   |
   +---+---+---+---+---+---+---+---+
B  |   |   |   |   |   |   |   |   |
   +---+---+---+---+---+---+---+---+
C  |   |   |   |   |   |   |   |   |
   +---+---+---+---+---+---+---+---+
D  |   |   |   | @ | O |   |   |   |
   +---+---+---+---+---+---+---+---+
E  |   |   |   | O | @ |   |   |   |
   +---+---+---+---+---+---+---+---+
F  |   |   |   |   |   |   |   |   |
   +---+---+---+---+---+---+---+---+
G  |   |   |   |   |   |   |   |   |
   +---+---+---+---+---+---+---+---+
H  |   |   |   |   |   |   |   |   |
   +---+---+---+---+---+---+---+---+
```

Black starts the game. The goal is to turn as many discs as possible to your own color. During the turn, the player places a new disc on the board. The only legal move is one by which you enclose a row of opponent's discs between your discs already on the board and the new disc (vertical, horizontal, or diagonal) with no empty squares in between. After the new disc has been placed, all enclosed discs are turned over (flanked) so that they become now yours. If no legal move is possible, the turn is skipped. When both players need to skip the turn, the game is over. You can find an overview of the rules on https://www.wikihow.com/Play-Othello.

Script and classes

`othello-script`

The script controls setting up the game with classes `Game` and `Player`. The all time score (number of games won by all players who have ever used the program) should be displayed after each game. After the game, the user should be asked whether another game should be played.

`Game`

Class `Game` should control the game by creating a board and implementing a method to play the game. The user needs to be able to choose how many human players are involved in the game (0,1, or 2). The computer player can play to any legal position randomly.

`Board`

Class `Board` needs to check after each new disc has been placed which discs need to be flanked, and to take care of this. It also needs to recognize when the game is over, and who won (unless there was a draw). **To help you out, this class is already implemented.** You are free to change it or to write a completely new one, but that should not be necessary. Here is the description for this class:

```
class Board(builtins.object)
 |  Board implements a class for the 8x8 board of the game Othello.
 |  Player discs are represented with integers 1 and 2, empty slots with 0.
 |  Coordinates are a combination of a capital letter and a number (e.g, A1, D3, ...)
 |
 |  Methods defined here:
 |
 |
 |  __init__(self)
 |      Constructor for the class Board
 |
 |  __repr__(self)
 |      String representation showing the current state of the board
 |
 |  legalmoves(self, player)
 |      Returns a list of all legal moves for player (integer: 1 or 2)
 |      Raises a ValueError if player is not known
 |
 |  marker(self, player)
 |      Returns the marker for player (integer: 1 or 2)
 |      Raises ValueError if player is not known
 |
 |  ndiscs(self, player)
 |      Returns the number of discs for player (integer: 1 or 2)
 |      Raises ValueError if player is not known
 |
 |  place(self, pos, player, flank=True)
 |      Places a new disc from player at position pos (string, f.ex. 'A1').
 |      Raises ValueError if player (integer: 1 or 2) is not known, position is not free or move is ill
 |      If flank is True, flanks the enclosed discs
 |      Returns the number of discs to flank
```

`Player`

Class `Player` needs to implement both a computer as well as a human player. The human player(s) must have names, so that the program can update the high score table with the number of wins for each player.

Other files

You must add a short text file `README.md` which explains how to run the program, and file `AUTHOR` with the author information (name, matriculation number and email address).

If your solution requires any additional libraries to be installed, those must be listed in file `requirements.txt`.

Grading

Tentative grading scheme (50/100 required for passing):

`>91 : 1 | 81-91 : 2 | 67-80 : 3 | 50-66 : 4 | <50 : 5`

- 40/100: Classes `Game` and `Player` implemented so that the game can be played
- 10/100: Script implemented for running from command line
- 5/100: Selecting the number of human players and the high score file from command line with `docopt`
- 15/100: Reasonable tests implemented for class `Board` methods `legalmoves` and `place`
- 10/100: Docstrings for classes and methods, other commenting when necessary
- 10/100: Decorators used correctly for getters and setters for class `Player`
- 5/100: Program does not crash with incompatible user input
- 5/100: Saving the game score to a file (and reading from the file)
- *Bonus* 5/100: Correctly provided license information in file `LICENSE`
- *Bonus* 5/100: Additional functionalities documented in file `README.md`
- *Bonus* 5/100: Correctly implemented package (including `setup.py` and script in directory `bin/`)
- Negative points can be given for clear mistakes in the logic or the implementation of the code

About the extra time

You have after the exam time extra time to correct small mistakes in the code (typos etc.) to make sure it works. Such small corrections will give full points.

You are also allowed to add new features (see grading above), but these will at maximum increase the grade of the exam by one, and only if you have passed the exam with the solution submitted during the actual time (min. 50/100).

Happy hacking!

Example run of the program, human vs. computer

Your solution can look different.

```
$ othello --humans 1
Name of player 1: Jani

Player Jani: @
Player AI-1: O

    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
B |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
C |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
D |   |   |   | @ | O |   |   |   |
  +---+---+---+---+---+---+---+---+
E |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
F |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
G |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
H |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+

Available moves: ['C5', 'D6', 'E3', 'F4']
next move for Jani (@)> C5
Player Jani placed a disc to C5

    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
B |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
C |   |   |   |   | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
D |   |   |   | @ | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
E |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
F |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
G |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
H |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+

Available moves: ['C4', 'C6', 'E6']
Player AI-1 placed a disc to C4

    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
B |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
C |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
D |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
E |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
F |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
G |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
H |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+

Available moves: ['B3', 'C3', 'D3', 'E3', 'F3']
next move for Jani (@)> D3
Player Jani placed a disc to D3

    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
B |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
C |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
D |   |   | @ | @ | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
E |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
F |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
G |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
H |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+

Available moves: ['C2', 'C6', 'E2', 'E6']
Player AI-1 placed a disc to E2

...
```

Example for computer vs. computer

Note that a game of Othello takes a bit of time, so for testing you may want to play computer against computer.

```
$ othello --humans 0

Player AI-1: @
Player AI-2: O

    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
B |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
C |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
D |   |   |   | @ | O |   |   |   |
  +---+---+---+---+---+---+---+---+
E |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
F |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
G |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
H |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+

Available moves: ['C5', 'D6', 'E3', 'F4']
Player AI-1 placed a disc to C5

    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
B |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
C |   |   |   |   | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
D |   |   |   | @ | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
E |   |   |   | O | @ |   |   |   |
  +---+---+---+---+---+---+---+---+
F |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
G |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+
H |   |   |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+

Available moves: ['C4', 'C6', 'E6']
Player AI-2 placed a disc to C4


...


    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A |   |   |   |   |   | O |   |   |
  +---+---+---+---+---+---+---+---+
B |   |   | @ | @ | O |   | @ |   |
  +---+---+---+---+---+---+---+---+
C |   | @ |   | O | @ | @ |   |   |
  +---+---+---+---+---+---+---+---+
D |   |   | @ | O | @ | @ | O |   |
  +---+---+---+---+---+---+---+---+
E |   |   | @ | @ | O |   | O |   |
  +---+---+---+---+---+---+---+---+
F |   |   | @ | O | @ |   | O |   |
  +---+---+---+---+---+---+---+---+
G |   | O | O |   |   | @ |   |   |
  +---+---+---+---+---+---+---+---+
H |   | O |   |   |   |   |   |   |
  +---+---+---+---+---+---+---+---+

Available moves: ['A4', 'A5', 'B6', 'C3', 'D8', 'E6', 'E8', 'F8', 'G4', 'G5', 'H1', 'H3']
Player AI-1 placed a disc to A4


...
```

```
    1   2   3   4   5   6   7   8
  +---+---+---+---+---+---+---+---+
A | O | O | O | O | O | O | O | O |
  +---+---+---+---+---+---+---+---+
B | O | O | @ | @ | O | O | O | O |
  +---+---+---+---+---+---+---+---+
C | O | @ | O | O | O | O | @ | O |
  +---+---+---+---+---+---+---+---+
D | O | @ | @ | O | O | O | @ |   |
  +---+---+---+---+---+---+---+---+
E | @ | @ | @ | @ | O | O | @ | O |
  +---+---+---+---+---+---+---+---+
F | @ | @ | @ | @ | @ | O | @ |   |
  +---+---+---+---+---+---+---+---+
G |   | @ | @ | @ | @ | @ | @ |   |
  +---+---+---+---+---+---+---+---+
H | @ | @ | @ | O | O | @ | @ | O |
  +---+---+---+---+---+---+---+---+

Available moves: []
Available moves: []

Player AI-2 wins (31 vs. 29 discs)!

All time high score
- AI-1: 7
- AI-2: 5

Do you want to play again [Y/n]?
```