Name: Manasi Jadhav, Shreya Jain, Neha Kolambe

Batch: B

UID: 2018140025, 2018140027, 2018140030

**Project: Tetris Game**

1. Scope well defined with objectives

‘Tetris’ game challenges the player to create complete lines by moving differently-shaped pieces, which scroll from the top to bottom of the playing field. The completed lines disappear and grant the player points, and the player can proceed to fill the vacated spaces. The game ends when the playing field is filled to the point that additional pieces can no longer descend. The longer the player can delay this inevitable outcome, the higher their score will be.

2. Detailed working of the project

The game starts with the player seeing the playing area. A random piece (random shape and color) starts falling from the top and the player can change its orientation using the arrow keys. Once the piece touches the bottom, the next piece with random shape falls in.

The goal is to complete an entire line which gets you points. This continues till the time the playing area fills. Once filled, the game ends and the final score is displayed.

The way we will represent our pieces will be using multidimensional lists. Each list will have multiple sub-lists that represent all of the possible rotations of each shape. This will make it much easier to rotate and represent our shapes visually when we eventually draw them to the screen. Grid is created using the grid data structure. Each element in the multidimensional list will be a tuple representing the color of the piece in that current position. The locked position parameter will contain a dictionary of key value pairs where each key is a position of a piece that has already fallen and each value is its color. We will loop through these locked positions and modify our blank grid to show these pieces.

When we are moving and rotating our shape we need to make sure that it is moving into a valid space. We have two parameters: grid and shape. We will check the grid to ensure that the current position we are trying to move into is not occupied. We can do this by seeing if any of the positions in the grid that the shape is attempting to move into have a color. If they have a color other than black than that means they are occupied, otherwise they are free. In order to end the game, we need to constantly be checking if the user has lost the game. We are simply going to check if any position in the given list is above the screen. If it is, we have reached the top and therefore lost the game. We make sure that the pieces fall at a certain time interval. A timer is set for the same

3. Contribution of each team member

Manasi, Shreya laid the base of the project which includes what all to involve (score, timer) and how to go about it.

Since programming in python was new to all, coding/GUI was done by three of us together.

Major GUI coding was done by Neha.

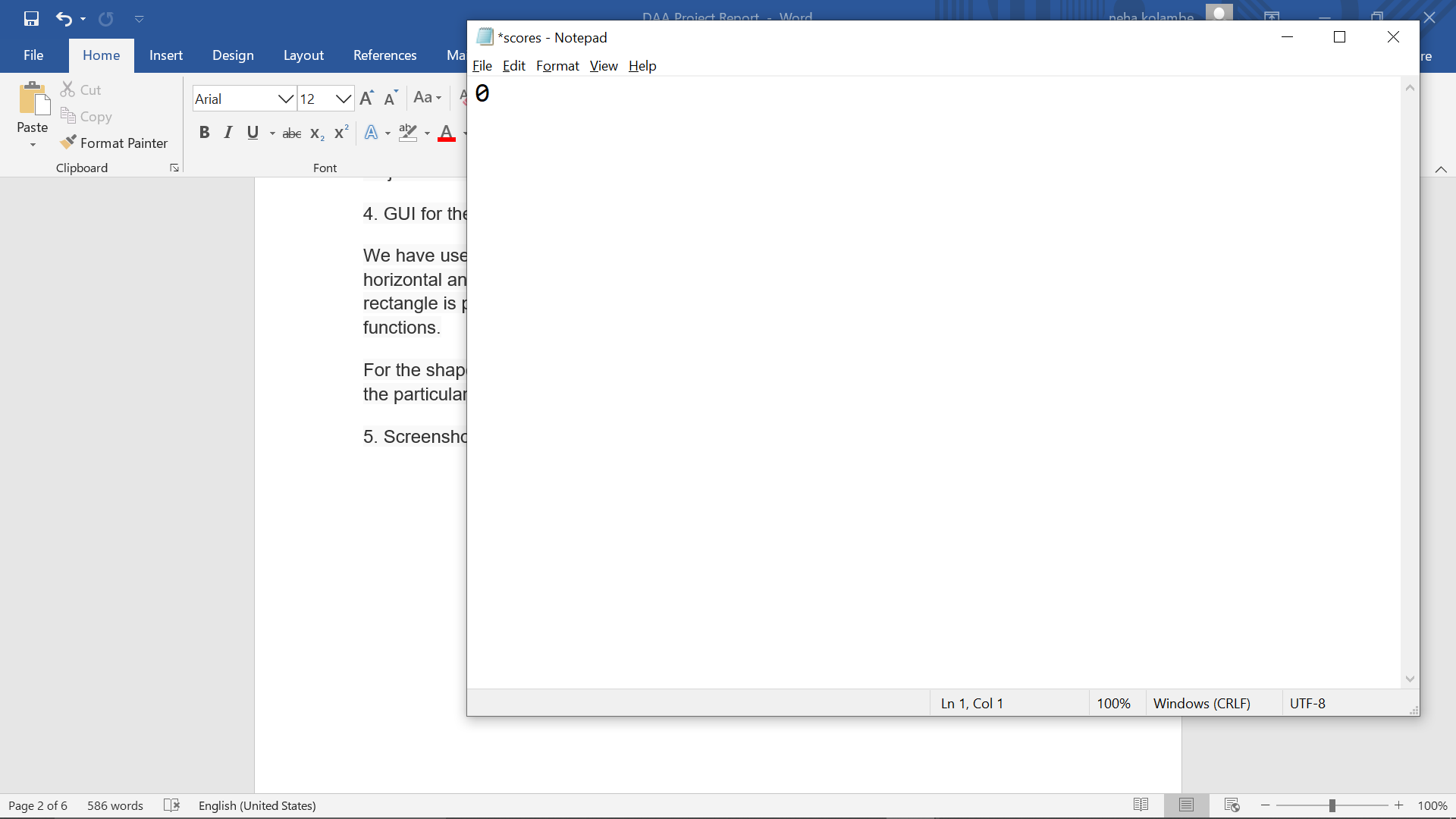
4. GUI for the project

We have used ‘pygame’ for the GUI. Using the basic functions, we created the grid (grey horizontal and vertical lines). We create a blank white surface using surfacefill. A rectangle is plotted using the blit function. Grid and borders are added using the desired functions.

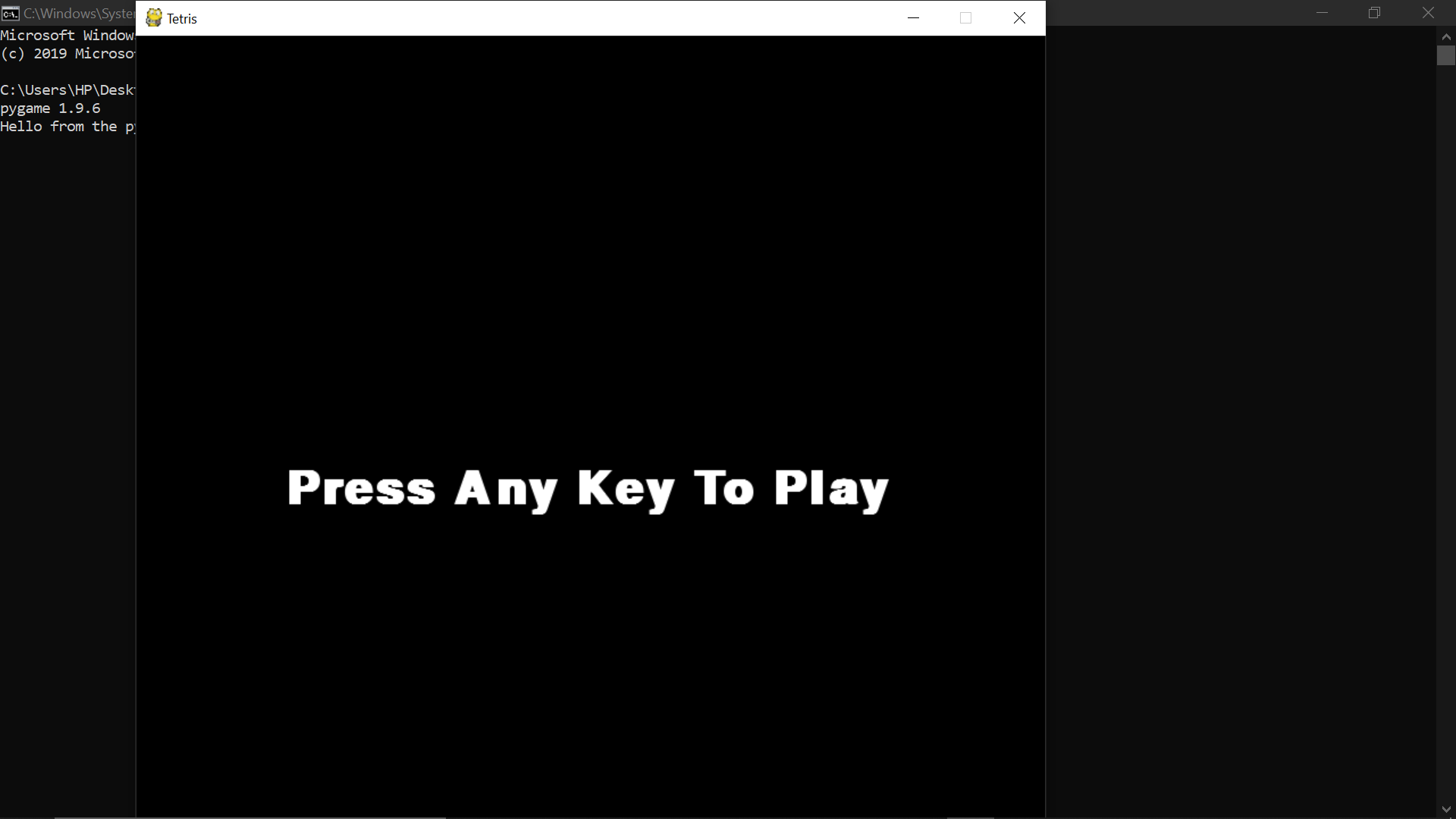
For the shapes, we made multidimensional lists, which store all possible orientations of the particular shape.

5. Screenshots

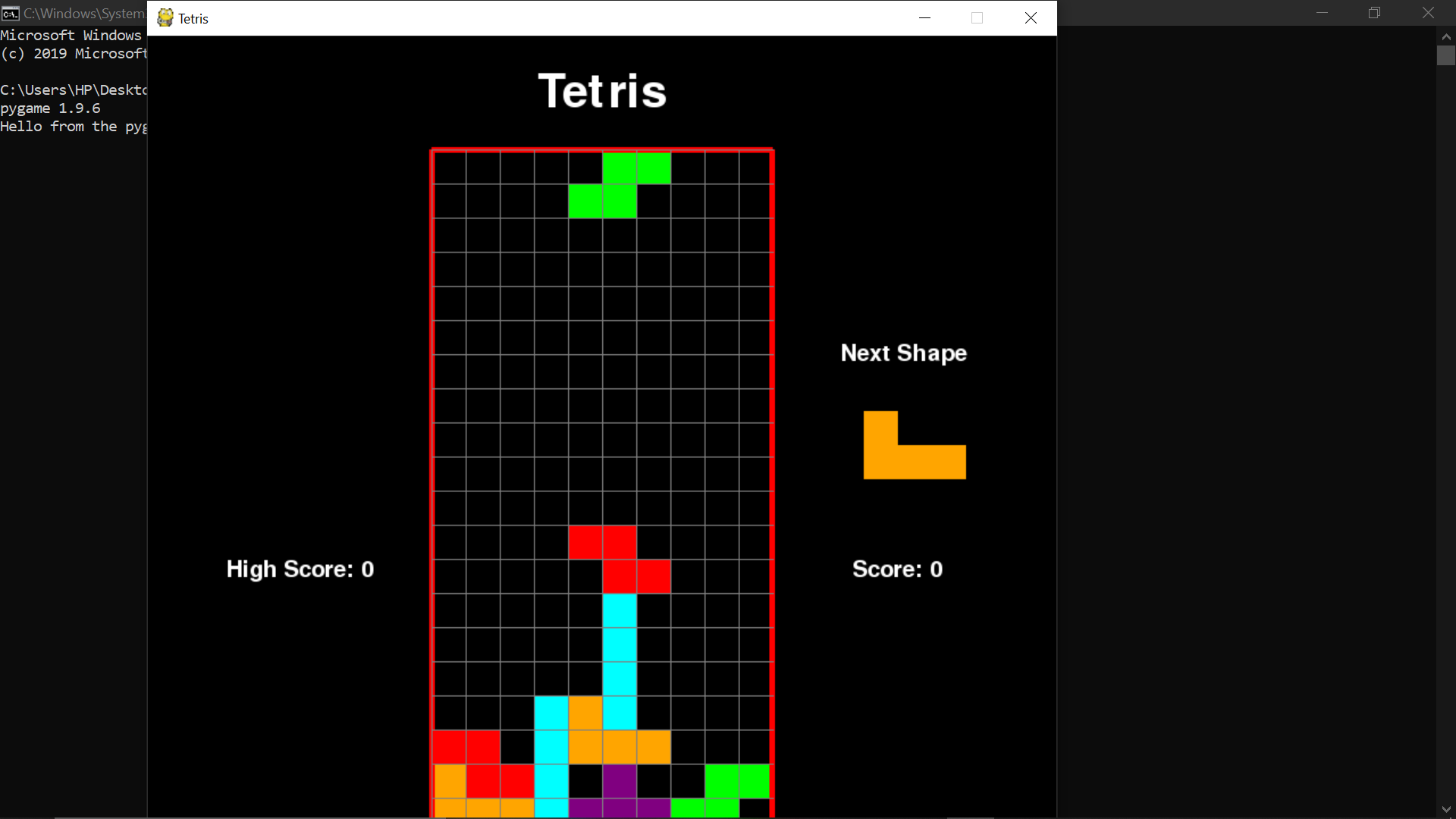
1) scores.txt file for saving Highscore



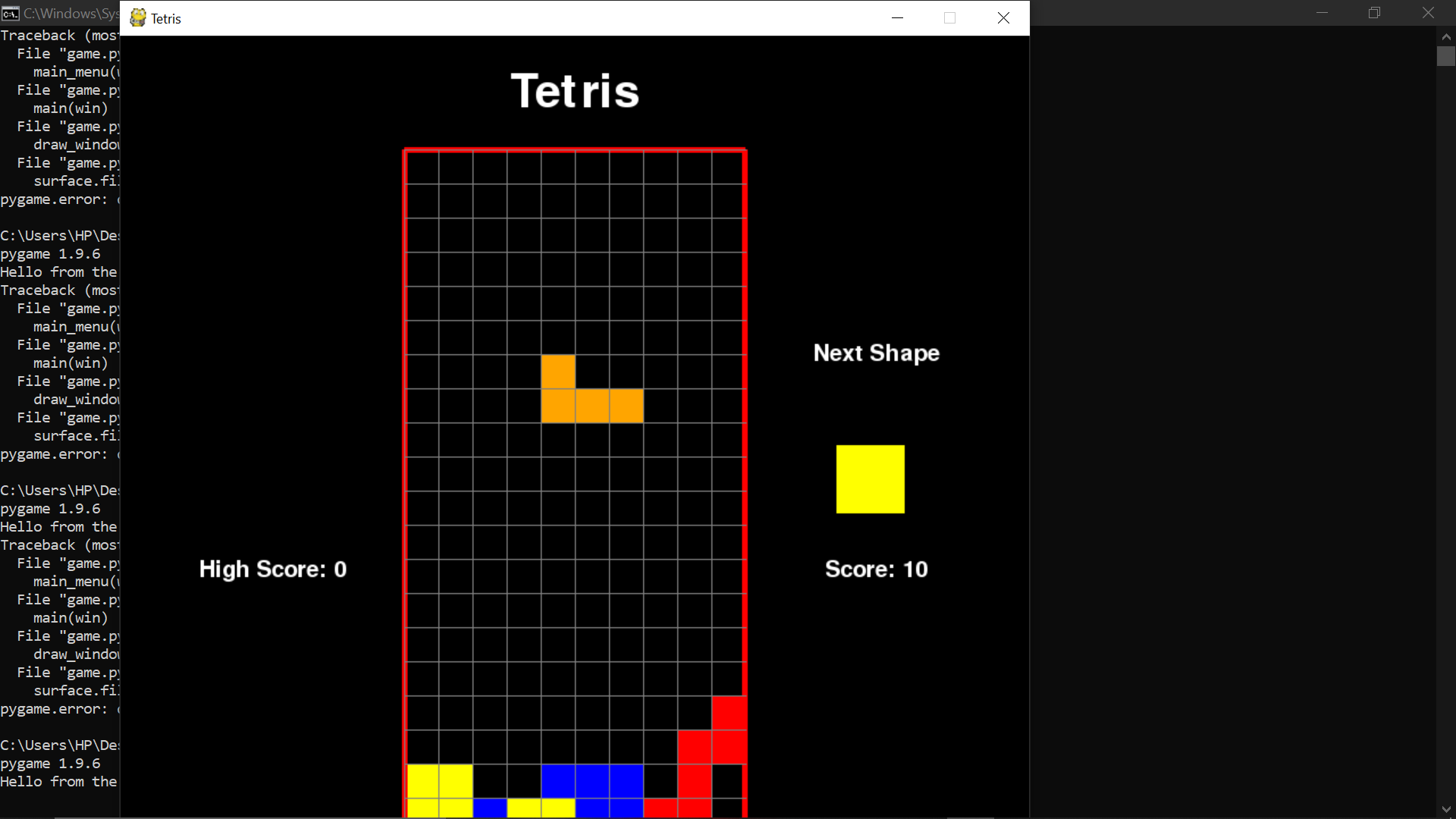
2) Here we start with the game



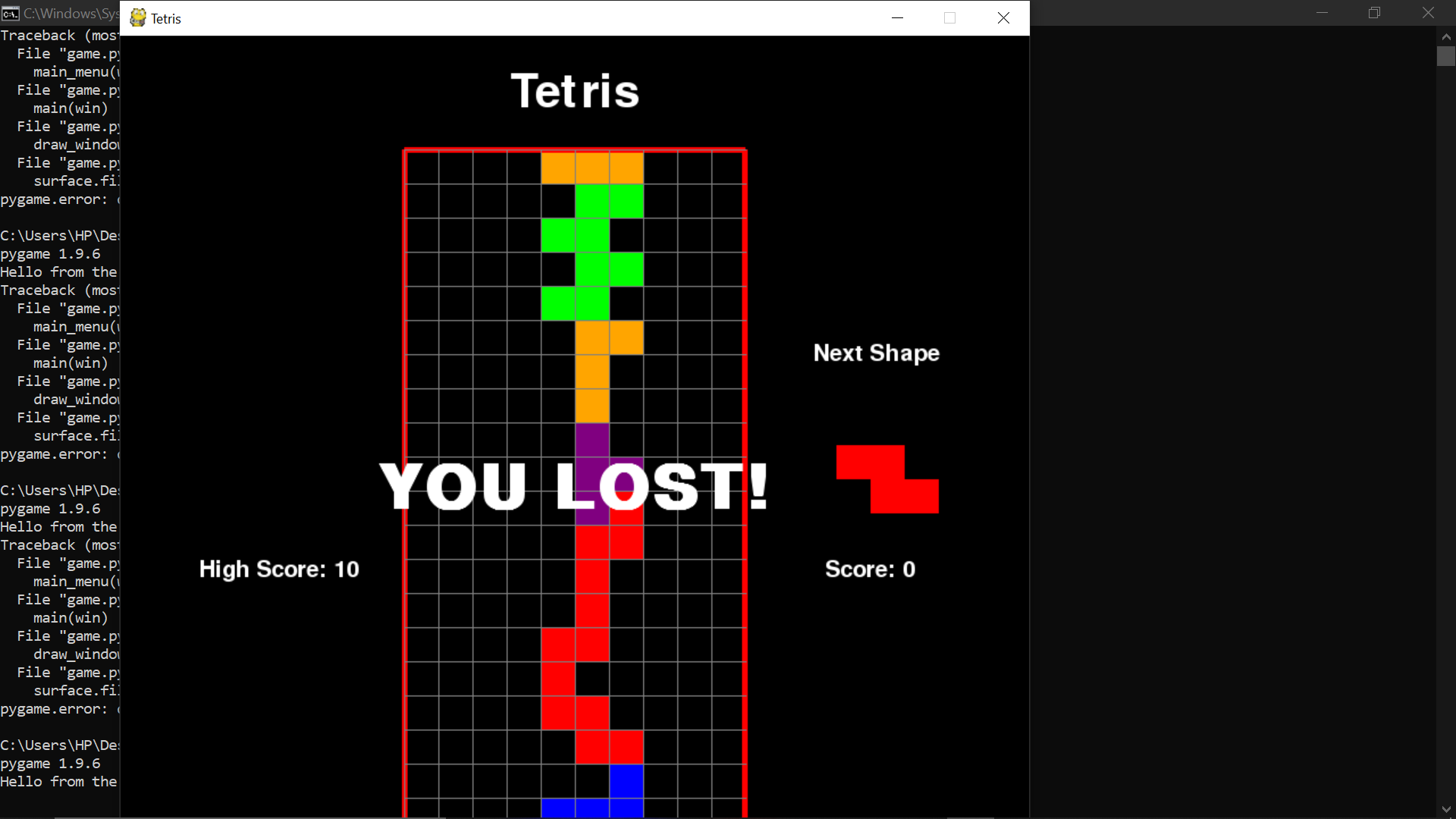
3) Initially, Score and Highscore is 0.



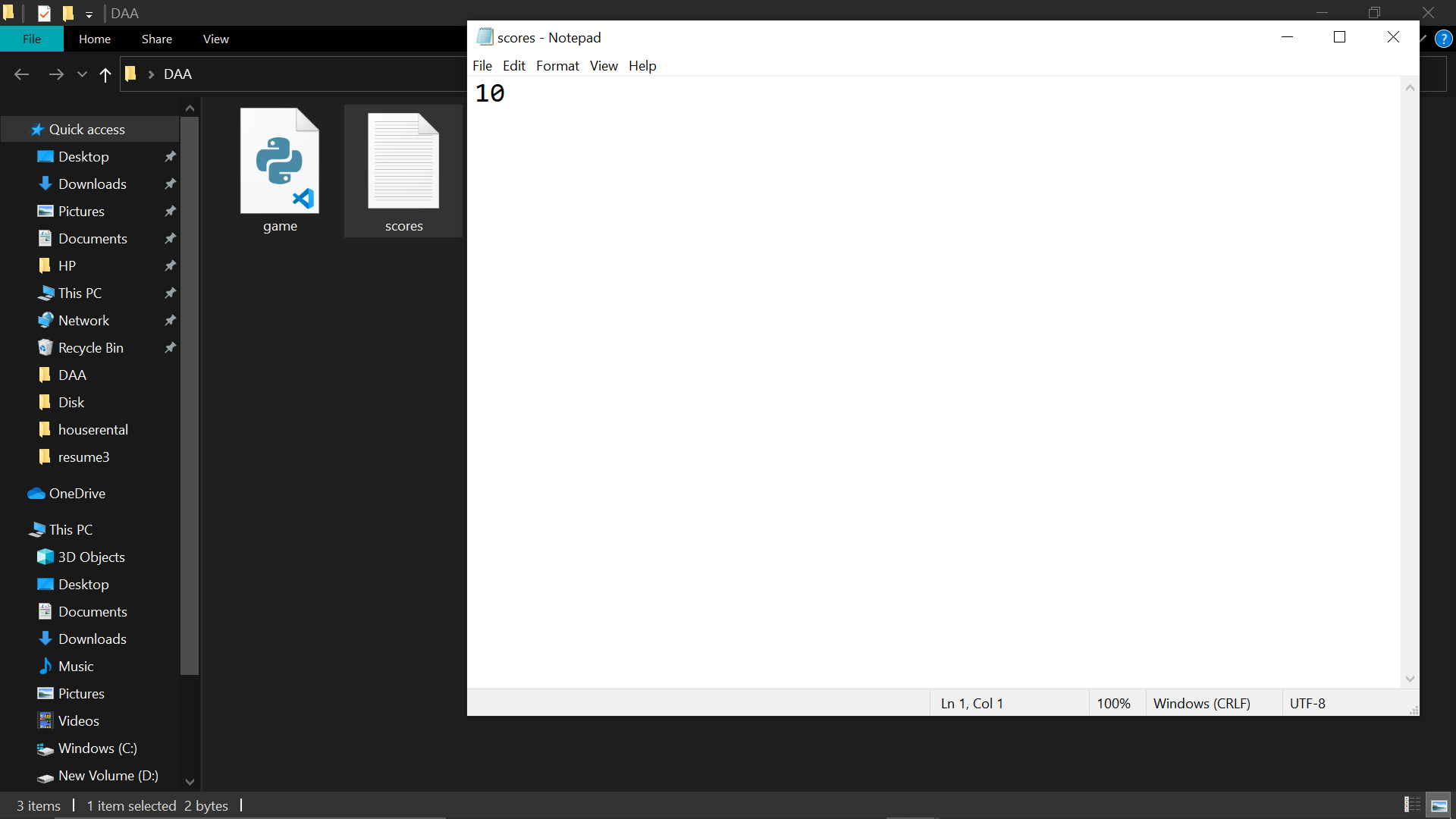
4) When the bottom line gets deleted we get 10 points.



5) When the grid gets fully filled, we get lost in the game.



6) And the Highscore gets updates in scores.txt file.



7) When we start again with the game, Highscore gets displayed.



6. References

<https://www.geeksforgeeks.org/introduction-to-pygame/>

<https://pythonprogramming.net/pygame-python-3-part-1-intro/>

<https://stackoverflow.com/>