Documentation – Match 3 game

Table of Contents:

1. Program’s structure

„assets” directory – the graphic files needed to create the GUI are stored here.

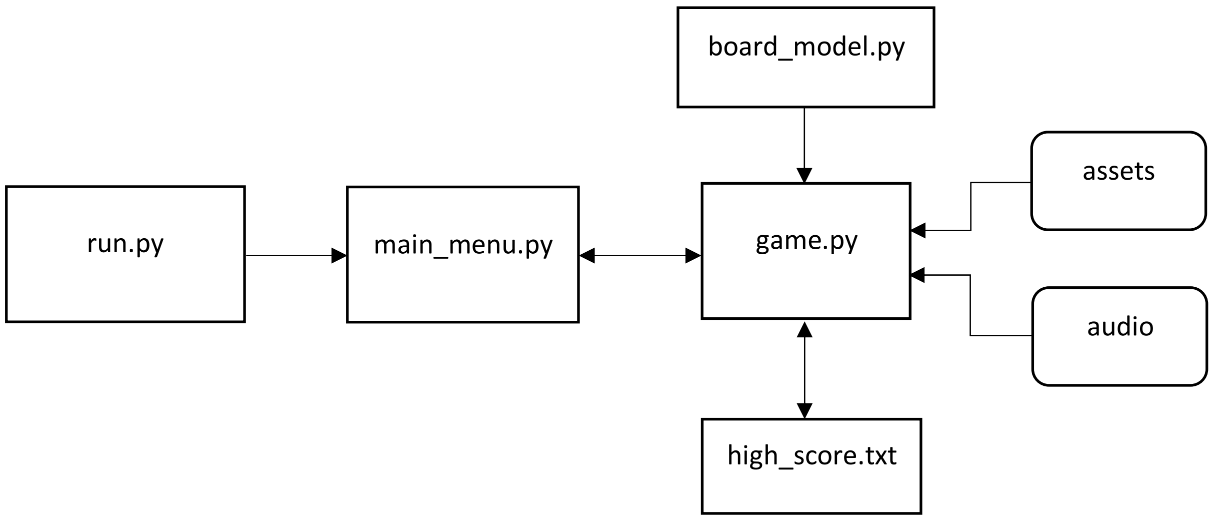
„audio” directory – storage of audio files.

high_score.txt – stores the best score.

main_menu.py – is responsible for the visualization of the main menu as well as for the entire logic of the menu operation.

game.py – responsible for the visualization of the game and its logic.

run.py – initializes the pygame window and launches the main menu. Is used to start the game.



2. main_menu.py

2.1. MainMenu class

Defines all of variables neccessary to create pygame window (graphics, width, height). Takes as an argument the „window” object that was created in the „run.py” module.

3. board_model.py

3.1. Cell class

Represents an object of cel in the board. It has 3 arguments: color, x, y. __eq__ comparator defines the comparision of cells on the board according to their colors.

3.2. Board class

Takes a single argument „size” which defines the size of board (8x8).

generate_grid(level) -> list – creates a random board consisting of various gems (the amount depends on the level number:
lvl 1 - 4 types
lvl 2 – 5 types
lvl 3 – 6 types
lvl 4+ - 7 types)

The board must be no loss [check_lost()], at least one match-making move must be available, and no matches have already been created on it [find_cells_to_delete()].

4. game.py

4.1. Game class

Takes arguments: window object, window width and height, board size. Sets the necessary arguments (including graphics, audio and fonts).

draw_window(grid) – displays all elements on the screen and updates the state of the board (graphically).

draw_selection(coords) – displays the frame of the selected gem in the screen (given coordinates).

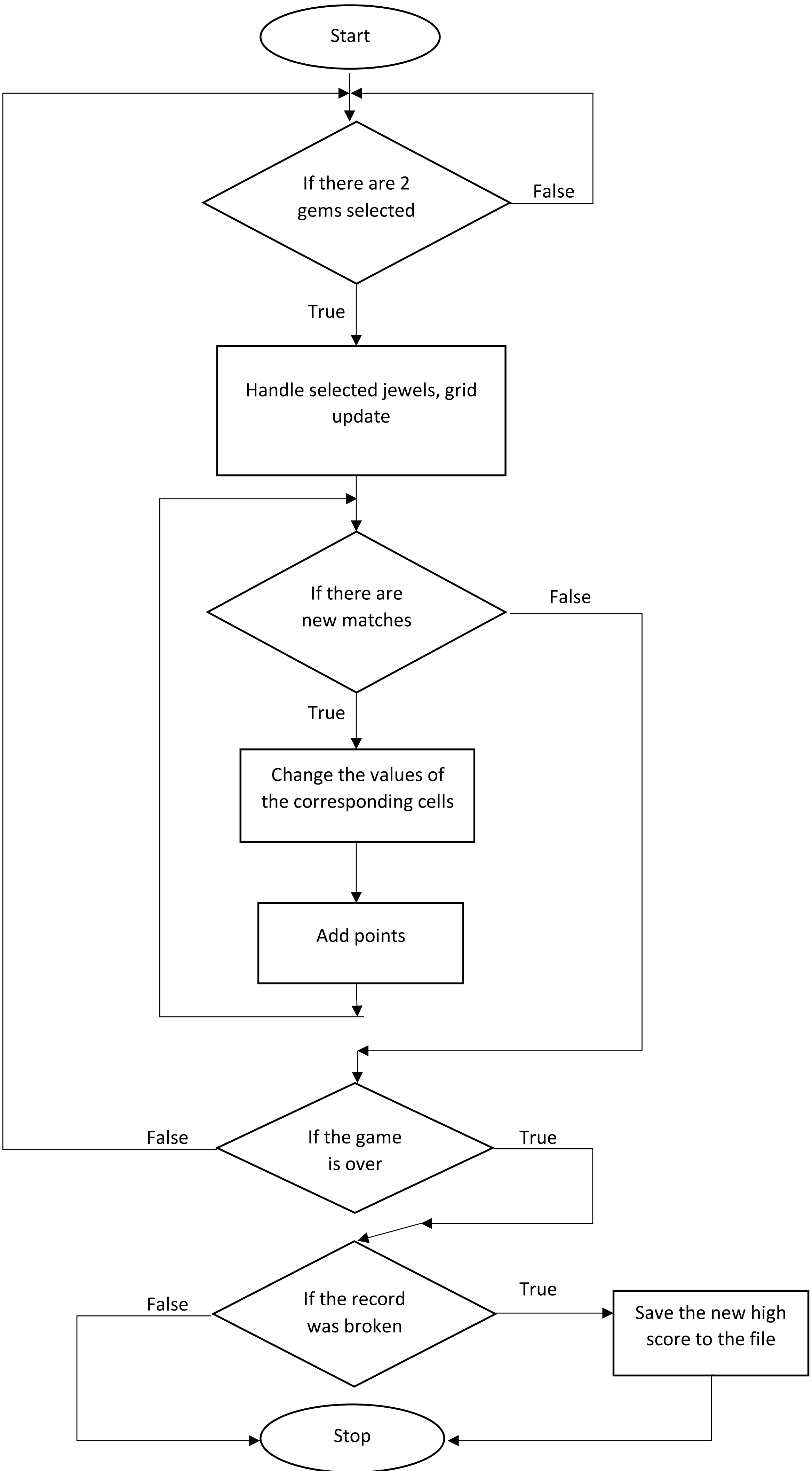
new_high_score() – displays information about the broken record on the screen and generates a sound.

next_level() – displays information about the next level and generates a sound.

game_over() – displays a failure message on the screen and pauses the music.

run() – is responsible for the entire logic of the game and calls particular functions each time the player makes a change on the board.

4.2. Game logic:


Start
If there are 2 gems selected
False
True
Handle selected jewels, grid update
If there are new matches
False
True
Change the values of the corresponding cells
Add points
If the game is over
False
True
If the record was broken
False
True
Save the new high score to the file
Stop