Game Labyrinth Documentation:

About the game: console-based C# game, where the aim of the player is to exit a labyrinth in less possible moves. The labyrinth has 7 rows and 7 columns. Player is presented as “\*”, labyrinth accessible (empty) cells are “-”, and walls, which the player can’t cross, are “X”.

1 project containing just 1 class has been refactored into new project, containing 8 classes and 4 interfaces, placed in 6 different folders.

Refactoring:

Meaningful names of the variables, methods and classes have been given therefore the code has become easy to read, understand and maintain.

Repeatable code has been replaced.

The project maintains strong cohesion and loose coupling.

It has correct behavior in all possible scenarios (All possible input from the user).

It follows and SOLID and DRY principles.

Useless comments have been replaced.

Constants have been placed in separate class.

Class diagram has been added.

Unit tests are implemented.

Classes:

* CommandExecuter – executes player’s movements – up, down, right, and left. Restarts player’s position every time a button is pressed.
* LabyrinthConstants – this is where all constant variables are stored. They have been given proper names and are made uppercased, following OOP principles
* Messages – here constant messages, printed on the console, are stored. They also have been given proper names and are made uppercased, following OOP principles
* LabyrinthTools – inherits ILabyrinthTools interface. This is where random labyrinth is generated and printed. It is provided at least one reachable exit for the player, also is checked when the player exit the labirynth.
* Player – inherits IPlayer interface. It handles the player start position inside the labyrinth, also gets and sets player’s current position on the X and Y axis. Also provides the movement of the player to the 4 directions and checks if player’s moves are valid. Drowns the player on the console and checks if he is outside the labirynth.
* Scoreboard – inherits IScoreboard interface. Provides the number of scores showed on the scoreboard, fills and updates the scoreboard, takes the number of moves of the current player and reads his name, updates the top five players by the number of moves they made during the game. Gets the worst score from the current game only,
* LabyrinthEngine – inherits IEngine interface. It executes input commands from the console and writes the constant messages on the console, stores player’s number of movements.
* Program – main class, starts the game.