# Sample Refactoring Documentation for Project “Balloons-Pop-4”

Team “Barium”

1. Redesigned the project structure:
   * Renamed the project to **BallonsPop**.
   * Renamed the main class **Program.cs** to **BallonsPop.cs**.
   * Separated functionality for each class: **BallonsPop.cs**, **BalloonsEngine.cs**, **ScoreEntry.cs**.
   * Added Test project: **BalloonsPop.Tests.**
   * Added unit tests and data-driven tests classes to the **BalloonsPop.Tests** project: **BalloonsPopTests.cs, ChartTests.cs, DataDrivenTests.cs, ScoreEntryTests.cs**
2. Refactored the source code:
   * Removed all unneeded empty lines in the project.
   * Added XML documentation.
   * Inserted empty lines between the methods.
   * Cleared most style cop warnings.
   * Added unit testing project.
   * Split the lines containing several statements into several simple lines, e.g.:

|  |  |  |
| --- | --- | --- |
| **if (input[i] != ' ') break;** | **🡪** | **if (input[i] != ' ')**  **{**  **break;**  **}** |

* + Formatted the curly braces **{** and **}** according to the best practices for the C# language.
  + Put **{** and **}** after all conditionals and loops (when missing).
  + Character casing: variables and fields made **camelCase**; types and methods made **PascalCase**.
  + Formatted all other elements of the source code according to the best practices introduced in the course “[High-Quality Programming Code](http://codecourse.telerik.com/)”.
  + Balloon popping is now done with 1 recursive method instead of 4(for each direction).
  + Separated responsibilities in two major parts – user i/o and balloonpop engine, that manages the inner state of the field and game stats.

1. Renamed methods:

* gen 🡪 **GeneratePlayField()**
* printMatrix 🡪 **FieldOutput()**
* checkLeft, checkRight, checkup, checkDown 🡪 **PopRowsAndCols()**
* change 🡪 **TryPopBalloons()**
* doit 🡪 **CollapseRows()**
* sortAndPrintChartFive 🡪 **GenerateChart()**
* signIfSkilled 🡪 **CheckIfWinning()**

1. Extracted the methods **ProcessMove(), GetInput(), CovertCharToInt(), PrintIntroMessage(), PrintIllegalMoveMessage(), PrintMoveMessage(), PrintInvalidCommandMessage(), PrintExitMessage()** from the method **Main()**.
2. Introduced struct **ScoreEntry** and moved all related functionality in it.
3. Introduced class **BalloonsEngine** and moved all related functionality in it( **GeneratePlayField(), CheckMoveValidity(), FieldOutput(), CollapseRows(), CheckIfWinning(), TryPopBalloons(), PopRowsAndCols(), RestartGame(), GenerateChart(), ChartPlaceIndex(), RecordHighscore()** methods).