|  |  |  |  |
| --- | --- | --- | --- |
| Refactoring Documentation for Project “Minesweeper – 5” Team Argon Reformatted the source code:  * + Removed all unneeded empty lines.   + Removed all unnecessary using directives.   + Added a new line after each closing curly bracket, where necessary.   + Updated **DisplayTopScores()** to use **string.Format()**.   + Corrected spacing following the C# best practices.   + Corrected formatting following the C# best practices.  Renamed variables:  * + In class **Board**: **Status.AlreadyOpened** 🡪 **Status.FieldAlreadyOpened**   + In class **Board**: **Status.SuccessfullyOpened**🡪 **Status.FieldSuccessfullyOpened**   + In class **Board**: **Status.AlreadyOpened** 🡪 **Status.FieldAlreadyOpened**  Renamed classes:  * + Renamed **Game** 🡪 **GameEngine.**  Change the type of some variables:  * + In class **Board**: **fields[][]** 🡪 **fields[,]**   + In class **Board**: all occurence of variables with names like **i** and **j** 🡪 appropriate names like **row** and **column**  Create new variables to replace some peace of code:  * + In method **Board.PrintGameBoardCurrentState():** string **horizontalLine**   + In method **Board.PrintGameBoardAllFieldsRevailed():** string **horizontalLine**  Change place of some variables:  * + From class **Board: random** 🡪 to method **Board.GenerateRandomNumber(): random**  Renamed methods:  * + Method **Board.PrintGameBoard()** 🡪 **Board.ToString()**   + Method **Board.PrintAllFields()** 🡪 **Board.ToStringAllFieldsRevealed()**   + Method **Game.Menu()** 🡪 **Game.StartGame()**   + Method **HighScores.CheckHighScore** 🡪 **HighScores.IsHighScore()**  Extracted methods:  * + **Board.PrintColumnIndexes()** from the method **Board.ToString()** and **Board. Board.ToStringAllFieldsRevealed() ()**   + Extracted methods for every player command: **RestartGame(),** **DisplayTopScores(), ExitGame()** and **CheckCoordinates()** – when entering a row and a column.  Implemented methods:  * + In method **Board.CheckIfWin()** implemented method **Board.CountOpenedFields()**  Extracted interfaces:  * + Extracted interface **IInputMethod**, with a single method **GetUserInput()** which returns the user input as a string. This way the project was decoupled from the console input, but still supports it.   + Extracted interface **IRenderer**, which contains three methods – **DisplayMessage()** which display the message in the renderer, **DisplayError()**, which display the error as an error message in the renderer and **DrawBoard()**, which displays the playing board. This way the project was decoupled from the console output, but still supports it.   + Extracted interface **IGameCommandExecutor**, which contains methods that control the flow of the game and uses an instance of **IRenderer** and **IInputMethod** to control the game.  Extracted classes:  * + Extracted class **HighScores**, which contains information about the highest scores and has methods **IsHighScore()**, for checking if a score is amongst the top scores, **AddTopScore()** for adding a score to the top scores and **ProcessScore()** which checks if a score is a top score and adds it to the list.   + Extracted class **ConsoleRenderer** which implements the **IRenderer** interface and moved all related functionality in it.   + Extracted class **ConsoleInputMethod** which implements the **IInputMethod** interface and moved all related functionality in it.   + Extracted class **DefaultGameCommandExecutor** which implements the **IGameCommandExecutor** interface and moved all related functionality in it.  Updated functionality:  * + Replaced all occurences of **Console.WriteLine()** in **DefaultGameCommandExecutor** with a method call to an implementation of the **IRenderer** interface.   + Updated **IsHighScore()** to use LINQ.   + Made **StartGame()** public as it should be the only method exposed to the public.   + Moved if-else statement chain to **DefaultGameCommandExecutor** as this functionality should be there.   + Renamed string **str** 🡪 **command** for clarity   + Updated caught **Exception** to **IndexOutOfRangeException**, which is more appropriate and more specific   + Changed if-else statement chain to switch statement:  |  |  |  | | --- | --- | --- | | if (command == "restart")  {  ...  }  else if (command == "exit")  {  ...  }  else if (command == "top")  {  }  else if (command == "coordinates")  {  ...  **}** | **🡪** | switch (command)  {  case "restart":  ...  case "top":  ...    case "coordinates":  ...    case "exit":  ...  } |  Moved the entry point of the program from the **GameEngine** class to a new class **MinesweeperGame**  * + Extracted duplicating code in method called **EndGame.**   + Added a check if the player has entered a blank name and if so asks the player to enter a valid name.   + Added **ToString()** method in the **Player** class.  Added exceptions:  * + **ArgumentNullException** when trying to assign a null value to a player name.   + **ArgumentOutOfRangeException** when a player score is   + **ArgumentException** when trying to compare an instance of the **Player** class with an object that is not an instance of the **Player** class.   + **ArgumentNullException** when trying to add a null value to the top scores using the **AddTopScore()** method.   + **ArgumentException** when trying to add a negative score to the top scores via the **ProcessScore()** method.   + **ArgumentException** when trying to assign a value smaller than 0 and bigger than 8 to a **Field** value – the number of adjacent mines can be 0 at the least and 8 at most.   + **ArgumentOutOfRangeException** if the number of fields for the minefield is less than 1 or if the number of mines is bigger than the total amount of fields.  Folder reorganization:  * + All classes in the project were organized in folders by their designation.   + Created folder **Common** and moved classes and enumerations **Board, BoardStatus, Field, FieldStatus, HighScores** and **Player**   + Created folder **InputMethods** and moved classes and interfaces **ConsoleInputMethod** and **IInputMethod**   + Created folder **Renderers** and moved classes and interfaces **ConsoleRenderer** and **IRenderer**   + Created folder **CommandExecutors** and moved classes and interfaces **DefaultGameCommandExecutor** and **IGameCommandExecutor.**  1. Updated namespaces to reflect the folder organization in the format **Minesweeper.FolderName**, where **FolderName** is the name of the folder, that contains the classes.  Unit testing:  * + Created unit tests to ensure that all the methods and classes are operating correctly.  Add comments and documentation:  * + XML documentation added for all the classes and all the properties in the project.   By Team Argon:  Vasil Dininski, Petar Yankov, Maria Endarova, Alexander Polichronov |