**Refactoring documentation for Project “King-Survival-7”**

Team: Plutonium

1. Redesigned the project structure:
   * Renamed the main **KingSurvival** class to **Engine**.
   * Renamed the method **Play** to **Run**.
   * Removed duplicating class **paww**, which obviously keeps the coordinates for the chess pieces.
   * Renamed coordinate class **kingg** to **Coordinates** and extracted it in a different file (Coordinates.cs).
   * Removed the global variable **flag** and the property, associated with it (**IsKingTurn**).
   * Moved all chess pieces in a new **Dictionary<char, ChessPiece>** for convenient access by symbol and removed all properties, associated with them.
   * Created a new **Dictionary<string, Coordinates>**, which keeps all possible directions (up left (**UL**), up right (**UR**), down left (**DL**) and down right (**DR**)).
   * Redesigned the moving of the chess pieces.
   * Introduced a new project **KingSurvival.Demo**, for demoing purposes.
2. Reformatted the source code:
   * Removed all useless comments (e.g. “tova e klasa Peshka, koito zadava peshak s koordinati X i Y”).
   * Split the lines containing several statements into several simple lines.
   * Formatted the curly brackets **{** and **}** according to the best practices for the C# language.
3. Renamed variables:
   * A new variable **boardRenderer**, which instantiates the **BoardRenderer** class for printing on the console.
4. Introduced constants in the main **Engine** class:
   * A constant **PawnASymbol**.
   * A constant **PawnBSymbol**.
   * A constant **PawnCSymbol**.
   * A constant **PawnDSymbol**.
   * A constant **KingSymbol**.
5. Introduced classes:
   * A class **ChessPiece**, which keeps information about a chess piece (the symbol and current coordinates).
   * A class **BoardRenderer**, which actually generates and draws the game board on the console.
   * A class **UserInput**, which gets the user input and returns it as a string, which later gets checked and if valid, executed.
6. Introduced enumerations:
   * An enum **Message**, which keeps information about all possible messages.
   * An enum **Player**, which keeps all possible players.
7. Introduced methods in the main **Engine** class:
   * A method **ExtractDirectionFromCommand**, which extracts the direction from the user input (possible directions: up left (**UL**), up right (**UR**), down left (**DL**) and down right (**DR**)).
   * A method **IsValidPawnCommand**, which checks whether a given command is a valid move pawns (pawns can’t move up).
   * A method **IsValidKingCommand**, which checks whether a given command is valid move for the king (the king can move up).
   * A method **AreValidCoordinates**, which checks whether given coordinates are in the game board.
   * A method **IsOccupied**, which checks whether a given cell is occupied by another chess piece.
   * A method **IsKingBlocked**, which checks whether the king has any valid moves left. Otherwise the game should end with a win for the pawns.
   * A method **IsKingOnFirstRow**, which checks whether the king has reached the first row, i.e. won.
   * A method **AllPawnsOnLastRow**, which checks whether all pawns have reached the last row, i.e. lost.
   * A method **MovePiece**, which moves a given chess piece to some designated coordinates.
   * A method **Print**, which calls the **Print** method of the **BoardRenderer** instance.
8. Made the rules of the game closer to those from its documentation.