1. Redesigned structure:
   1. Separated the labyrinth generation (InitializeLabyrinth()) and visualization (ShowLabyrinth()) in a separate class: LabyrinthBoard
   2. Separated the Top Scores logic (method ShowTopScores and string[] topScores) in a class TopScores
   3. Renamed the class Labyrinth to Engine as only the main logic of the moving remains there
   4. Removed some not necessary comments.
2. Work on the LabyrinthBoard class:
   1. Added method bool IsPossibleCell(int row, int col) which checks if the cell on the given row and column is free to move on it and if it is within range.
   2. Added method bool isPieceOnEdge(int row, int col) which checks if the cell where the Piece is located at the moment, is on an edge
   3. Introduced constant int LabyrinthSize = 7 for the number of row and consts in the Labyrinth
   4. Introduced fields that keep the position of the moving piece (int piecePositionRow and int piecePositionCol) plus respective properties.
   5. Introduced methods that move the piece left, right up and down if possible. This is to take away this functionality from the engine.
   6. Due to the above two changes, modified the ShowLabyrinth method, so that it can print \* where the piece is;
   7. Reconfigured the ShowLabyrinth method into a ToString() method, so that the LabyrinthBoard class be decoupled from the actual means of rendering the board.
   8. Set capital letters in constant LabyrinthSize
   9. Created new constant PIECEPOSITION for piece of position for initial row and column equal to 3.
   10. Removed the key work “this” in properties PiecePositionRow and PiecePositionCol
   11. Changed the name of ll to labyrinth.
   12. Removed some whitespaces.
   13. Added ‘this.’ Prefix to indicate a members of the class.
3. Work on the TopScores class:
   1. Removed whitespace.
   2. Added ‘this.’ Prefix to indicate a members of the class.
   3. Added a constant for the number of TopScores, int NumberOfTopScores
   4. Replaced the string array with a KeyValuePair<string, int> list, so that the scores can be sortable and easily searched.
   5. Removed the check for null values in the ShowTopScores method, as it is no longer necessary with a list
   6. Removed the long if condition in the end of ShowTopScores, replaced with check if the list count is 0.
   7. Added a new method EnterTopScore, as there was no such functionality and this made the class useless
4. Work on the Engine class:
   1. Made the engine class use the functionality of the LabyrinthBoard class:
      1. The engine doesn’t make the checks if a cell is free or not and if it possible to move there or not, instead it asks the LabyrinthBoard class
      2. The engine doesn’t keep the information where the piece is anymore, it just asks the LabyrinthBoard class where it is and to move it, if possible. Removed the fields m and n that used to keep this.
   2. Replaced the topScores functionality with the functionality from the TopScores class.
   3. Made a method “Start” which contains the logic from the Program method.
   4. Renamed the variable \_continue with correct name flagContinue.
   5. Renamed the variable ll with correct name labyrinth.
   6. Removed the rest of the occurrences of private i=0, j=0,m=3,n=3.
   7. Changed the if (input.Length > 1 || input.Length == 0) with short boolean expression input.Length == 1.
   8. Created two new methods ProcessInputDirection(input) and ProcessInputCommand(input) for process input command or direction from the user.
   9. Created new method WalkInLabirinth() – moved here all doubled logic for checking of the new cell or exit.
   10. Crated method Restart () with initialization of the flagContinue for every time when this method is called. Used when needed ‘restart’.
   11. Removed some whitespaces.
   12. Added ‘this.’ Prefix to indicate a members of the class.
   13. Made the Engine class use the functionality of the TopScores class