# Chess Console Game Description

**-Main()** function prepares console for working with Unicode and launches GameLoop().

**-GameLoop()** creates and handle all necessary for game process data such a: *Desk* and *Players* creation,

user input retrieving.

**-User input** processing implemented as a while loop which gets input from console, process a dispatched message by calling proper functions and methods. It also checks correctness of the input, shows the game board, asks proper Player for a move.

*-Desk class* is a heart of the application. It contains current game board outlook(wstring), cells state(matrix of ChessPieces\*). It also holds vital methods such as: Start(), Move(), Print(), Save(), Load(), various checking methods which verify the game state, etc.

**-Move()** accepts coordinates off the cells and for moving Piece ‘from’ it’s cell ‘to’ the next cell and moves, checks if desired move allowed by game rules by calling Rule() method of the chosen Piece (Piece->Rule()). It also provides proper behavior of the pieces in case En Passant, Promotion, Castling and the game: Check, Checkmate, Stalemate.

**-Pieces** are present as derived classes of a ChessPiece class. Each piece knows it name(‘P’ – Pawn, ‘K’ – King, etc.) and color (‘W’ – White or ‘B’ – Black), some of them (King, Rook) remember do the made any move during current game(need for Castling validation). Also essential things is are Rules which dictates the only valid behavior of the certain Piece. Implemented through virtual functions overriding.

**Interesting trick**: Rule() method’s declarations of base classes are pure virtual but they also have implementation(body). It allows to make the rule that fits all pieces except one(kNight), reuse it and not to junk the main scope of the class at the same time.(Cool thing)