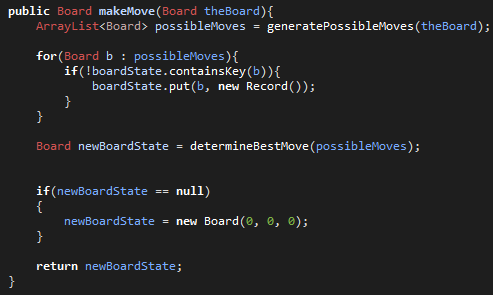
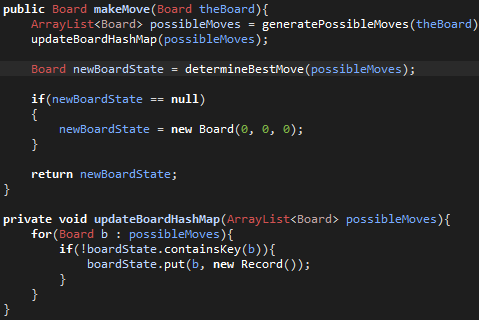
**Nim Refactor ch 4 & 5**

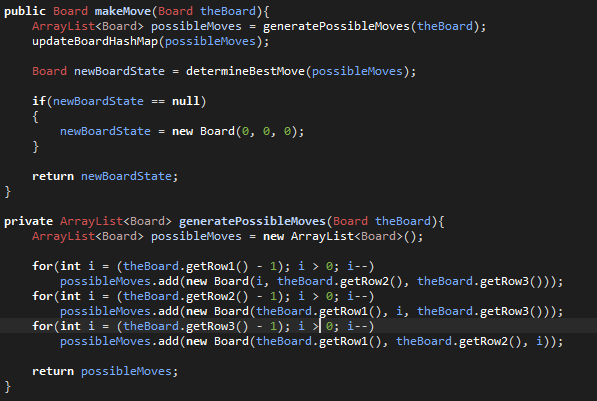
Jonathan Jensen, Zane Ferrell, Jake Wedemeyer

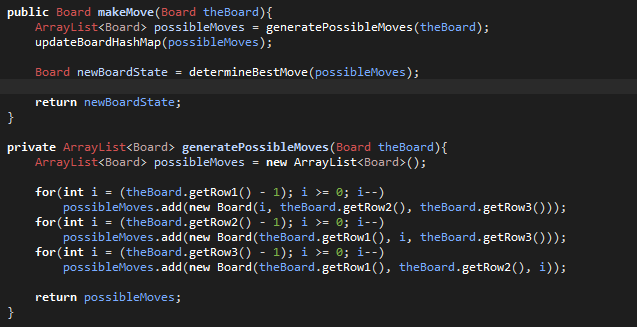
Computer Class:





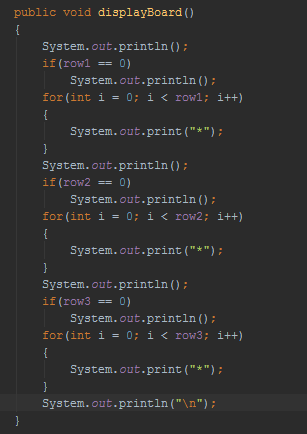
The Extract Method Rule: Here we were doing an extra task within our makeMove method and causes some Coincidental Cohesion, so we have moved it our of our makeMove method and put it in its own.

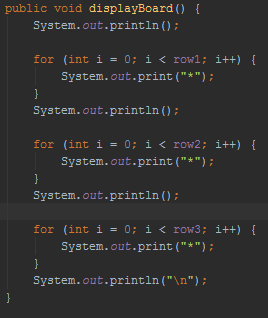




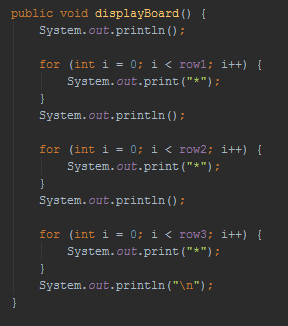
Replace Algorithms: Here we had an extra check made in the makeMove method largely due to an oversight in our algorithm. So, we fixed our generatePossibleMoves method to also account for the 0 stars in a row condition and removed unnecessary code.

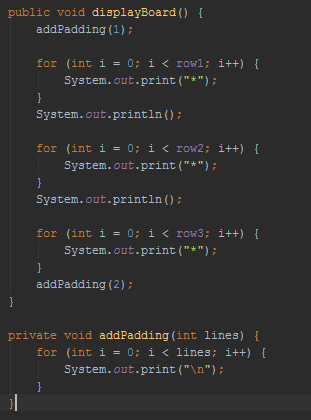
Board Class:





Replace Algorithms rule: Here there was code checking for an empty row which was not needed because the for loop already performs that check.





Cohesion: Using a more descriptive means to tell people who look at the code what is happening

**Before:**

**public** **static** **void** HvH()

{

**boolean** isRunning = **true**, turn = **true**;

Board board = **new** Board();

**while**(isRunning)

{

**if**(board.isOver())

{

System.*out*.println("Game over! Player " + (turn?"1 ":"2 ") + "wins!");

isRunning = **false**;

}

**else**

{

**int** row, x;

board.displayBoard();

System.*out*.println("Player " + (turn?"1's ":"2's ") + "turn! Please choose a row(1, 2, or 3):");

**while**(**true**)

{

**try**

{

row = *input*.nextInt();

**if**(row > 3 || row < 1)

**throw** **new** Exception();

**if**(board.getRow(row) == 0)

{

System.*out*.println("That row is already empty! Please try another.");

**throw** **new** Exception();

}

**break**;

}

**catch**(Exception e)

{

System.*err*.println("Invalid Input");

System.*out*.println("You goofed. Please choose an integer 1, 2, or 3:");

}

}

System.*out*.println("You have chosen row " + row + ". Now please choose a number to remove between 1 and " + board.getRow(row));

**while**(**true**)

{

**try**

{

x = *input*.nextInt();

**if**(x > 0 && x <= board.getRow(row))

{

board.makeMove(row, x);

System.*out*.println(x + " removed from row " + row + ".");

turn = turn?**false**:**true**;

}

**else**

{

**throw** **new** Exception();

}

**break**;

}

**catch**(Exception e)

{

System.*err*.println("Invalid Input");

System.*out*.println("You goofed. Please choose an integer between 1 and " + board.getRow(row));

}

}

}

}

*runMenu*();

}

**After:**

**public** **static** **void** HvH()

{

Player p1 = **new** Player("1"), p2 = **new** Player("2");

**boolean** isRunning = **true**, turn = **true**;

Board board = **new** Board();

**while**(isRunning)

{

**if**(board.isOver())

{

System.*out*.println("Game over! Player " + (turn?"1 ":"2 ") + "wins!");

isRunning = **false**;

}

**else** **if**(turn)

{

board = p1.takeTurn(board);

turn = turn?**false**:**true**;

}

**else**

{

board = p2.takeTurn(board);

turn = turn?**false**:**true**;

}

}

*runMenu*();

}

**public** **class** Player {

**private** String name;

**private** **static** Scanner *input*;

**public** Player(String name)

{

**this**.name = name;

}

**public** Board takeTurn(Board board)

{

**int** row, x;

board.displayBoard();

System.*out*.println("Player " + name + "'s turn! Please choose a row(1, 2, or 3):");

**while**(**true**)

{

**try**

{

row = *input*.nextInt();

**if**(row > 3 || row < 1)

**throw** **new** Exception();

**if**(board.getRow(row) == 0)

{

System.*out*.println("That row is already empty! Please try another.");

**throw** **new** Exception();

}

**break**;

}

**catch**(Exception e)

{

System.*err*.println("Invalid Input");

System.*out*.println("You goofed. Please choose an integer 1, 2, or 3:");

}

}

System.*out*.println("You have chosen row " + row + ". Now please choose a number to remove between 1 and " + board.getRow(row));

**while**(**true**)

{

**try**

{

x = *input*.nextInt();

**if**(x > 0 && x <= board.getRow(row))

{

board.makeMove(row, x);

System.*out*.println(x + " removed from row " + row + ".");

}

**else**

{

**throw** **new** Exception();

}

**break**;

}

**catch**(Exception e)

{

System.*err*.println("Invalid Input");

System.*out*.println("You goofed. Please choose an integer between 1 and " + board.getRow(row));

}

}

**return** board;

}

}

**Rule: Extract Subclass**

Although I did not show the entire class to save space, this is demonstrating the changes from not having a player class and creating and doing everything it does locally (in multiple methods) and now having a Player class and coding appropriately. This is not only used in the method HvH() but also the method HvC().