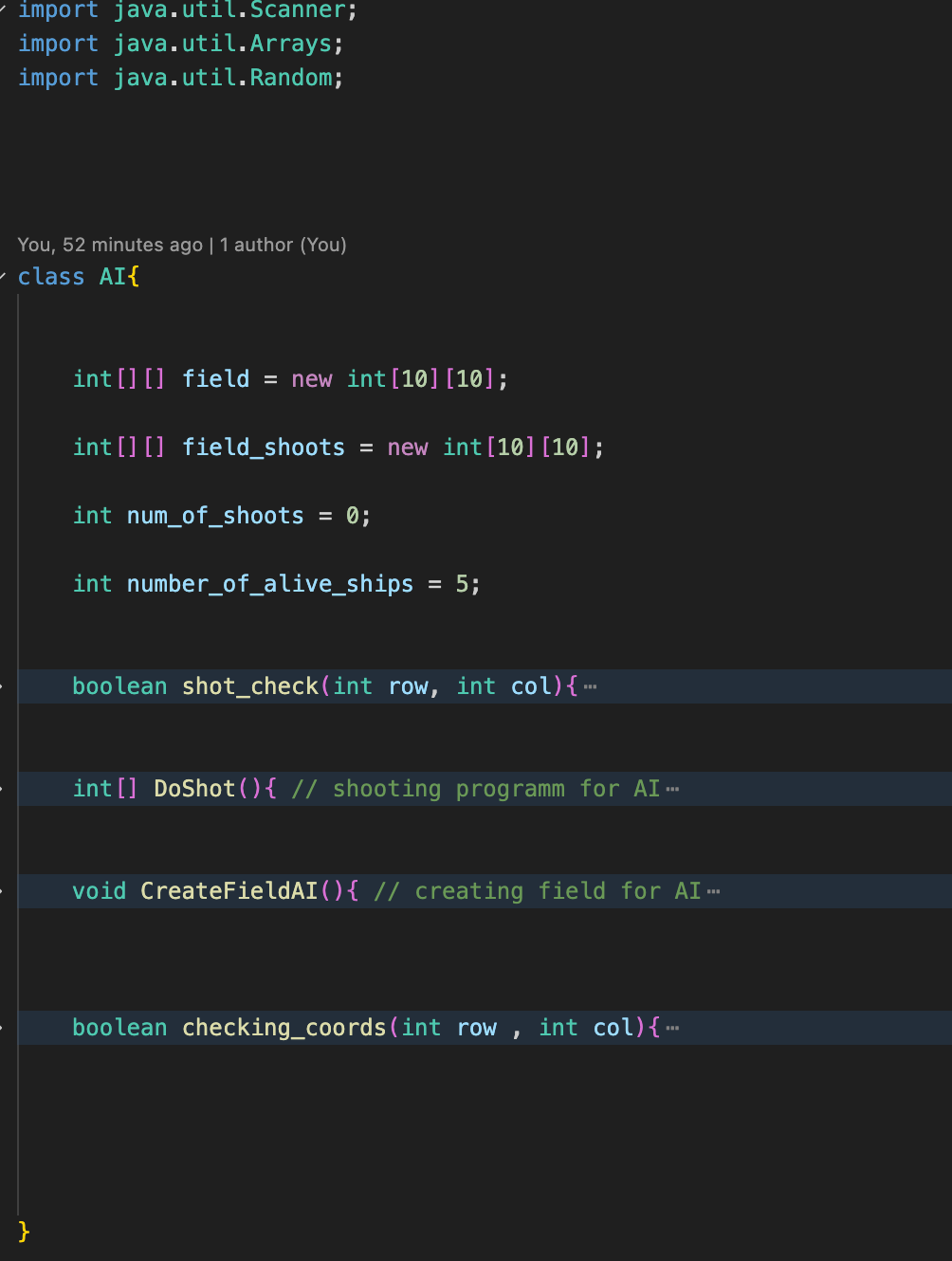
BattleShips is a very popular tactical game where the main task is to find and destroy a certain number of ships. The winner is the one who can sink all the opponent's ships first. The game continues until one of the players wins (destroys all enemy ships).

Speaking more about the process, in the game, there are 2 options – you can play against AI (ext.) and against real player. Each of the players has 5 (ext.) ships whose coordinates are checked for correctness (ext.). After setting up battle fields, each player tries to kill opponent’s ships one after another ( player 1 always starts ). After every try, there is a text which says if the shot was successful or not.

Talking about code, various mistakes that the user can commit (ext) are being checked, so there is no chance of inputing invalid data. The code itself was written as a number of classes which are connected with each other, so the OOP method was used.

**Code explanation:**



Firstly, we import all the necessary libraries.

**CLASS AI:**

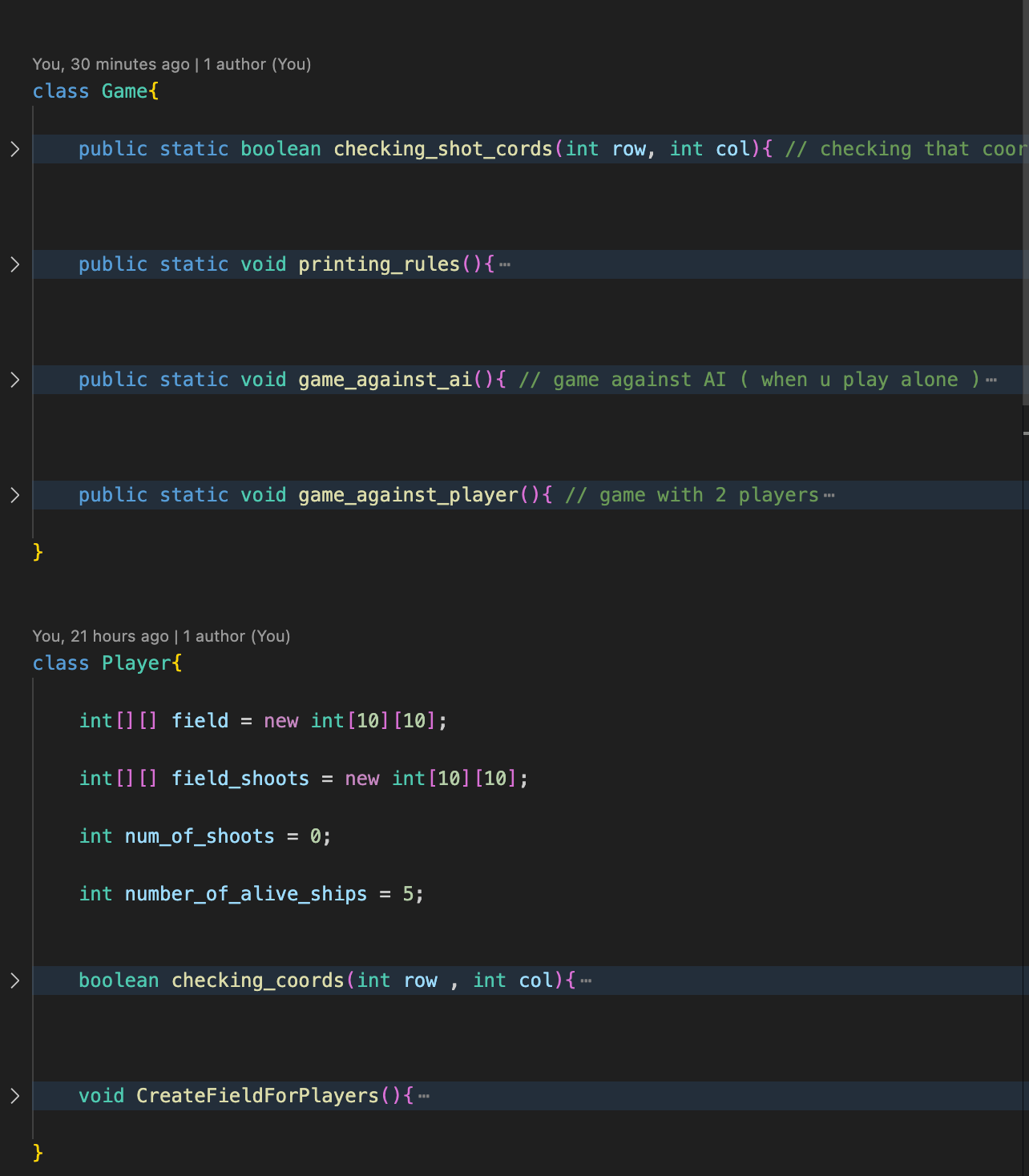
class AI – is a class which consists of all the stuff which will be used if player decides to play against AI

Values:

* field is a 2d array which represents ships
* field\_shoots is a 2d array which represents point of shots
* num\_of\_shoots counts number of shots
* number\_of\_alive\_ships is a number of ships which are still in game

Functions:

* Shot\_check checks that AI didn’t choose a point which was chosen in other tries
* DoShot generates values which will be used as a coordinates of shot
* CreateFieldAI creates field with ships ( ships’ coordinates are being chosen by random library )
* Checking\_coords just checks if coordinates of ships are valid



**CLASS GAME:**

Class game – is a class which manage all the game processes

Functions:

* Printing\_rules prints out rules of the game in the start
* Game\_against\_ai conducts the game against AI ( when player plays alone )
* Game\_against\_player conducts the game against other player
* Checking\_shot\_cors is just a function that helps checking shot coordinates of a player

**CLASS PLAYER:**

Class player - is a class which consists of different functions which will help to conduct Player’s game

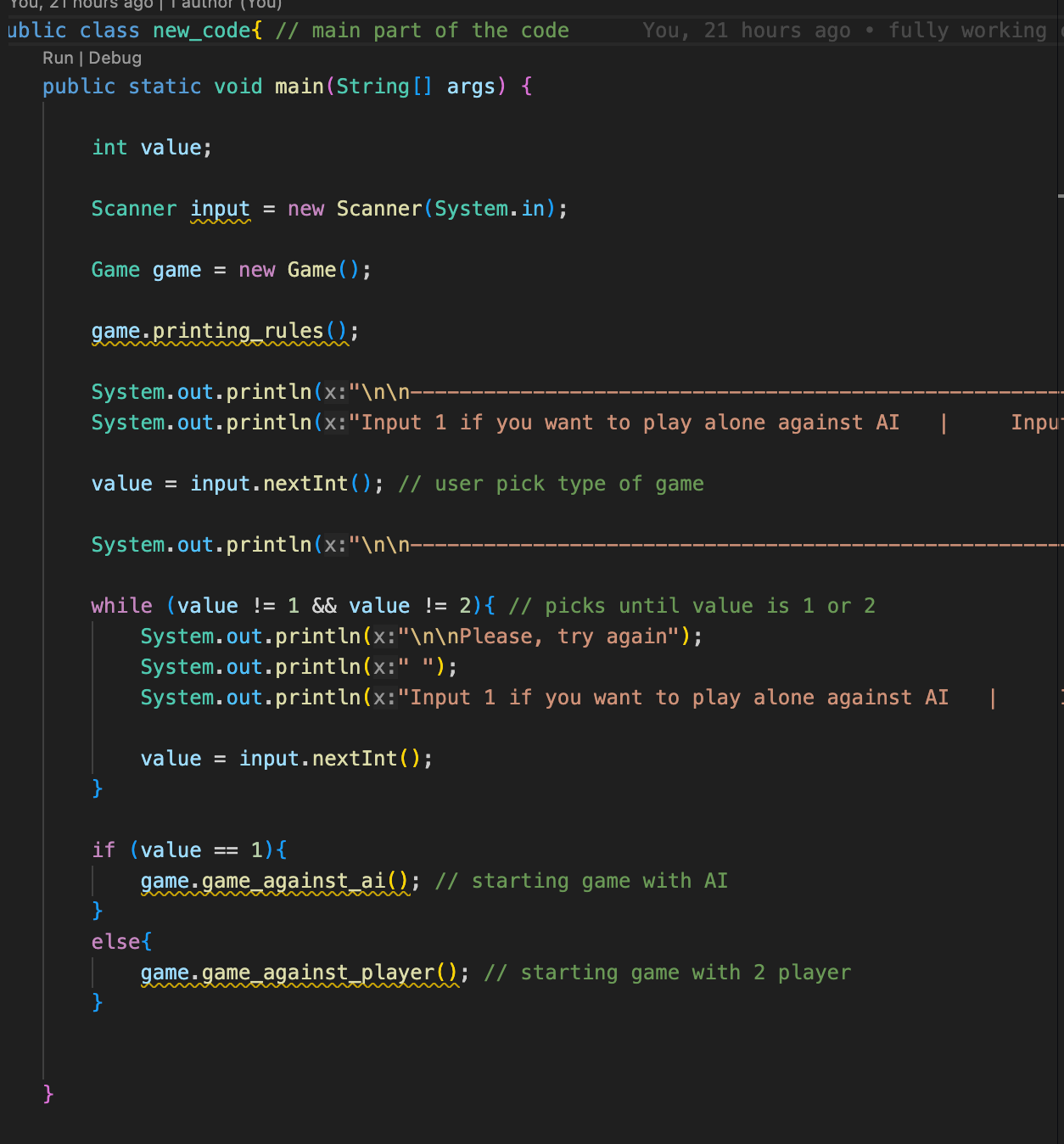
Values:

* field is a 2d array which represents ships
* field\_shoots is a 2d array which represents point of shots
* num\_of\_shoots counts number of shots
* number\_of\_alive\_ships is a number of ships which are still in game

Functions:

* CreateFieldForPlayers creates field with ships ( ships’ coordinates are being chosen by user’s input )
* Checking\_coords just checks if coordinates of ships are valid

**PUBLIC CLASS NEW\_CODE:**



It starts all the code.

Firstly, it prints game’s rules. Then it asks user to choose the mode of game - when program asks for the input, this input will be declared into the integer named value. If value equals 1, game mode is player against AI and the specific function which starts game against AI is being run . If value equals 2, then game mode is player against player and the specific function which starts game is being run.

Pseudocode:

game = new Game()

game.game\_against\_player()

CLASS Game

FUNCTION game\_against\_player()

DECLARE whos\_turn : INTEGER

DECLARE row : INTEGER

DECLARE col : INTEGER

whos\_turn = 1

player\_1 = new Player()

player\_2 = new Player

player\_1.CreateFieldForPlayers()

player\_2.CreateFieldForPlayers()

WHILE player\_1.number\_of\_alive\_ships > 0 AND player\_2.number\_of\_alive\_ships > 0

INPUT row

INPUT col

IF whos\_turn == 1 THEN

player\_1.field\_shoots[row – 1][col – 1] = 1

IF player\_2.field[row – 1][col – 1] == 1 THEN

OUTPUT “KILL”

player\_2.number\_of\_alive\_ships = player\_2.number\_of\_alive\_ships – 1

player\_2.field[row][col] = 0

ELSE

OUTPUT “MISS”

whos\_turn = 2

ENDIF

ELSE

player\_2.field\_shoots[row – 1][col – 1] = 1

IF player\_1.field[row – 1][col – 1] == 1 THEN

OUTPUT “KILL”

player\_1.number\_of\_alive\_ships = player\_2.number\_of\_alive\_ships – 1

player\_1.field[row][col] = 0

ELSE

OUTPUT “MISS”

whos\_turn = 1

ENDIF

ENDIF

ENDWHILE

IF player\_1.number\_of\_alive\_ships == 0 THEN

OUTPUT “1 player is winner”

ELSE

OUTPUT “2 player is winner:

ENDIF

ENDFUNCTION

ENDCLASS

CLASS Player()

DECLARE field : ARRAY[0:9] OF ARRAY[0:9] OF INTEGER

DECLARE field\_shoots : ARRAY[0:9] OF ARRAY[0:9] OF INTEGER

DECLARE number\_of\_alive\_ships : INTEGER

number\_of\_alive\_ships = 5

FUNCTION CreateFieldForPlayers()

DECLARE row : INTEGER

DECLARE col : INTEGER

INPUT row

INPUT col

FOR I 🡨 1 to 5 STEP 1

INPUT row

INPUT col

Field[row – 1][col – 1] = 1

ENDFUNCTION

ENDCLASS