Task 5 and 6: Pseudocode

Coordinate.printCoordinate() method

1. Initialise String coordinate
2. If showShip is true (players board)
3. If isHit is true and hasShip is true (coordinate is hit and has ship)
4. Set coordinate to HIT (constant for ‘x’)
5. Else if isHit is true and hasShip is false (coordinate is hit but doesn’t have ship)
6. Set coordinate to MISS (constant for ‘-‘)
7. Else if isHit is false and hasShip is true (coordinate isn’t hit but has a ship)
8. Set coordinate to ship.getInitial converted to string
9. Else if isHit is false and hasShip is false (coordinate isn’t hit and doesn’t have ship)
10. Set coordinate to EMPTY (constant for ‘ ‘)
11. Else (opponents board)
12. If isHit is true and hasShip is true
13. Set coordinate to HIT
14. Else if isHit is true and hasShip is true
15. Set coordinate to MISS
16. Else if isHit is false and hasShip is true
17. Set coordinate to UNKNOWN (constant for ‘ ‘)
18. Else if isHit is false and hasShip is false
19. Set coordinate to UNKNOWN
20. Return coordinate

ArtificialPlayer.takeGuess() method

1. Initialise Random rand from Random class import
2. Initialise int randRow
3. Initialise int randColumn
4. Initialise Boolean coordinateNotHit to false
5. While coordinateNotHit is false
6. Let randRow = random number from rand
7. Let randColumn = random number from rand
8. If coordinate on opponents board at randRow and randColumn is not hit
9. Set coordinateNotHit to true
10. If coordinate on opponents board at randRow and randColumn has a ship
11. Display Player 2 has scored a hit!
12. Increment hitsOnOpponent by 1
13. Else
14. Display Player 2 missed!
15. Increment missesOnOpponent by 1

Task 7: Test data table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data set name | Input | | Output | |
| **Variable** | **Value** | **Variable** | **value** |
| Data Set 1 | Ship 1 | Name = “Aircraft Carrier”  Row = 2  Column = 2  Orientation = HORIZONTAL | Message | Fleet placed successfully |
| Ship 2 | Name = “Destroyer”  Row = 6  Column = 6  Orientation = HORIZONTAL |
| Data set 2 | Ship 1 | Name = “Battleship”  Row = 3  Column = 8  Orientation = HORIZONTAL | Message | Fleet could not be placed: [Battleship is two spaces too big] |
| Data set 3 | Ship 1 | Name = “Battleship”  Row = 3  Column = 8  Orientation = VERITCAL | Message | Fleet placed successfully |
| Data set 4 | Ship 1 | Name = “Cruiser”  Row = 4  Column = 5  Orientation = HORIZONTAL | Message | Fleet placed successfully |
| Ship 2 | Name = “Submarine”  Row = 5  Column = 4  Orientation = VERTICAL |
| Data set 5 | Ship 1 | Name = “Cruiser”  Row = 4  Column = 5  Orientation = VERTICAL | Message | Fleet placed successfully |
| Ship 2 | Name = “Submarine”  Row = 5  Column = 4  Orientation = HORIZONTAL |

Length: Aircraft Carrier = 5, Battleship = 4, Cruiser = 3, Submarine = 3, Destroyer = 2

Task 8: Desk check table. Player.placeFleet() method.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Line number | Variables | | | | | |
| orientation | fleet[] | ocean[][] | placeCheck | row | column |
| 1 |  | fleet[0] = Ship(“Aircraft Carrier”, 5) |  |  |  |  |
| 2 |  |  |  |  | = 4 |  |
| 3 |  |  |  |  |  | = 2 |
| 4 | = HORIZONTAL |  |  |  |  |  |
| 5 |  |  |  |  |  |  |
| 6 |  |  |  | = ocean[4][2].hasShip() |  |  |
| 6 |  |  |  | = ocean[4][3].hasShip() |  |  |
| 6 |  |  |  | = ocean[4][4].hasShip() |  |  |
| 6 |  |  |  | = ocean[4][5].hasShip() |  |  |
| 6 |  |  |  | = ocean[4][6].hasShip() |  |  |
| 7 |  |  |  |  |  |  |
| 8 |  |  | ocean[4][2].setShip(fleet[0]) |  |  |  |
| 8 |  |  | ocean[4][3].setShip(fleet[0]) |  |  |  |
| 8 |  |  | ocean[4][4].setShip(fleet[0]) |  |  |  |
| 8 |  |  | ocean[4][5].setShip(fleet[0]) |  |  |  |
| 8 |  |  | ocean[4][6].setShip(fleet[0]) |  |  |  |