**Explanation of my solution**

**Content**

[**Basic Operations** 2](#_Toc105855599)

[1 FindMaxValue 2](#_Toc105855600)

[2 FindMinValue 3](#_Toc105855601)

[3 Swap 4](#_Toc105855602)

[4 ShifLeftByOne 5](#_Toc105855603)

[5 CreateRandomArray 6](#_Toc105855604)

[6 CreateRandomMatrix 7](#_Toc105855605)

[7 CoppyArray 8](#_Toc105855606)

[8 FindInSortedArray 9](#_Toc105855607)

[**Recursion** 10](#_Toc105855608)

[**Object Oriented Programming** 11](#_Toc105855609)

## **Basic Operations**

For all the following methods, I’ve also created some UnitTests in Array.Tests.cs.

# 1 FindMaxValue

****

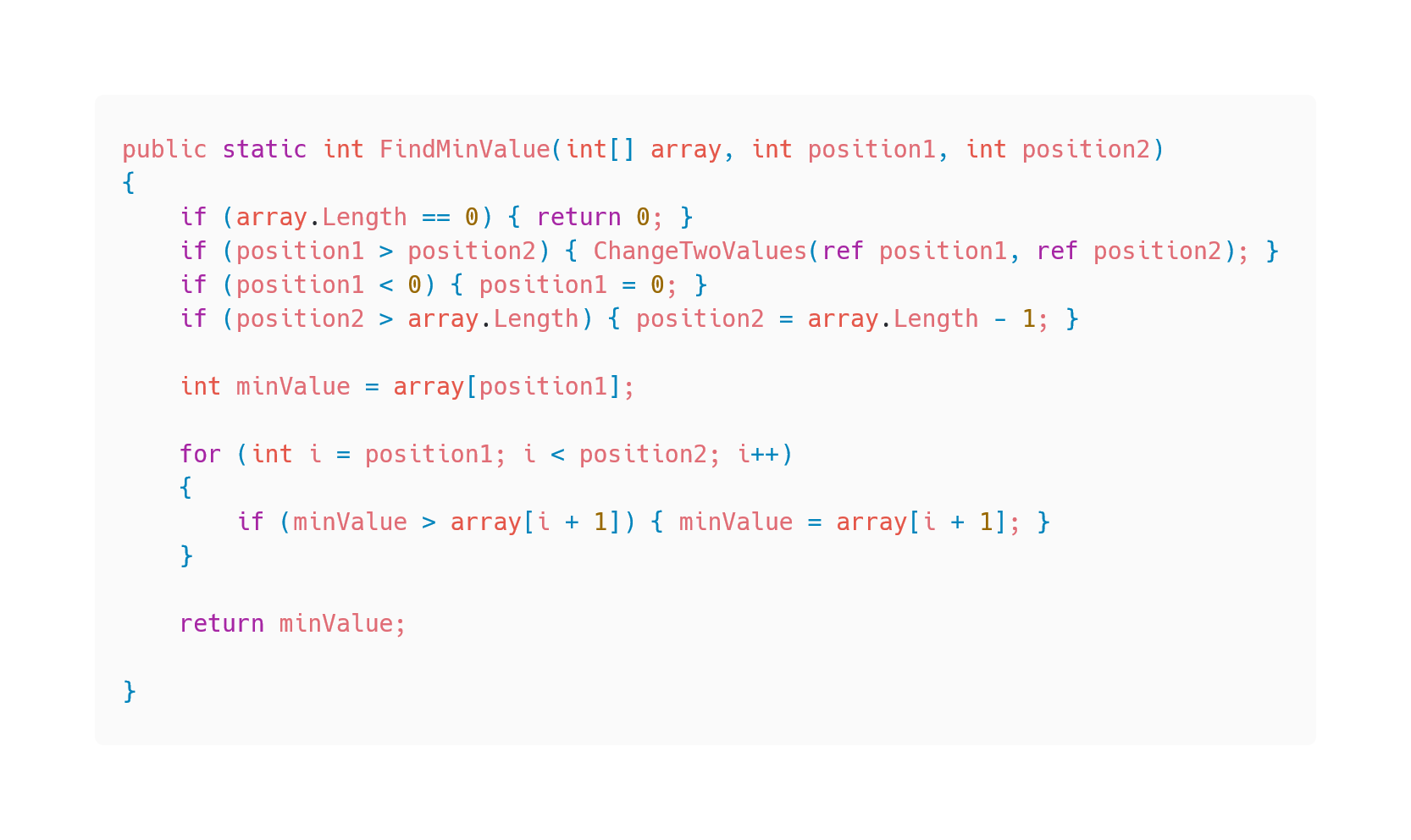
Explanation:

This method takes 1 array of integers and 2 integer variables. The first variable is the start position and the second one is the end position. The goal of this method is to search the highest number in this array between these 2 positions. First, the method checks if the array is empty or not. If it’s empty my method returns 0. After that it checks with another method if the first position is higher as the second one, if a user is entering the numbers this way I expect that he just messed up the order and I swap the values. I also took care of the scenario if the user is entering numbers who could go out of range of the array, if he enters a number which is < 0, the start position will be 0 (start of the array) and if he enters a number which is higher as the array length, the number will be as high as much numbers are in the array.

I added an integer variable – maxValue and I initialized it with the first number of the array. After that I wrote a for loop who is looping through the array and checks if maxValue is higher as the current number of the array, if it’s higher the loop goes to the next number, if maxValue is lower, then the current number is the new maxValue.

After the for loop is finished, the method returns the variable maxValue.

# 2 FindMinValue



Explanation:

For this method I used almost the same code as for the FindMaxValue method.

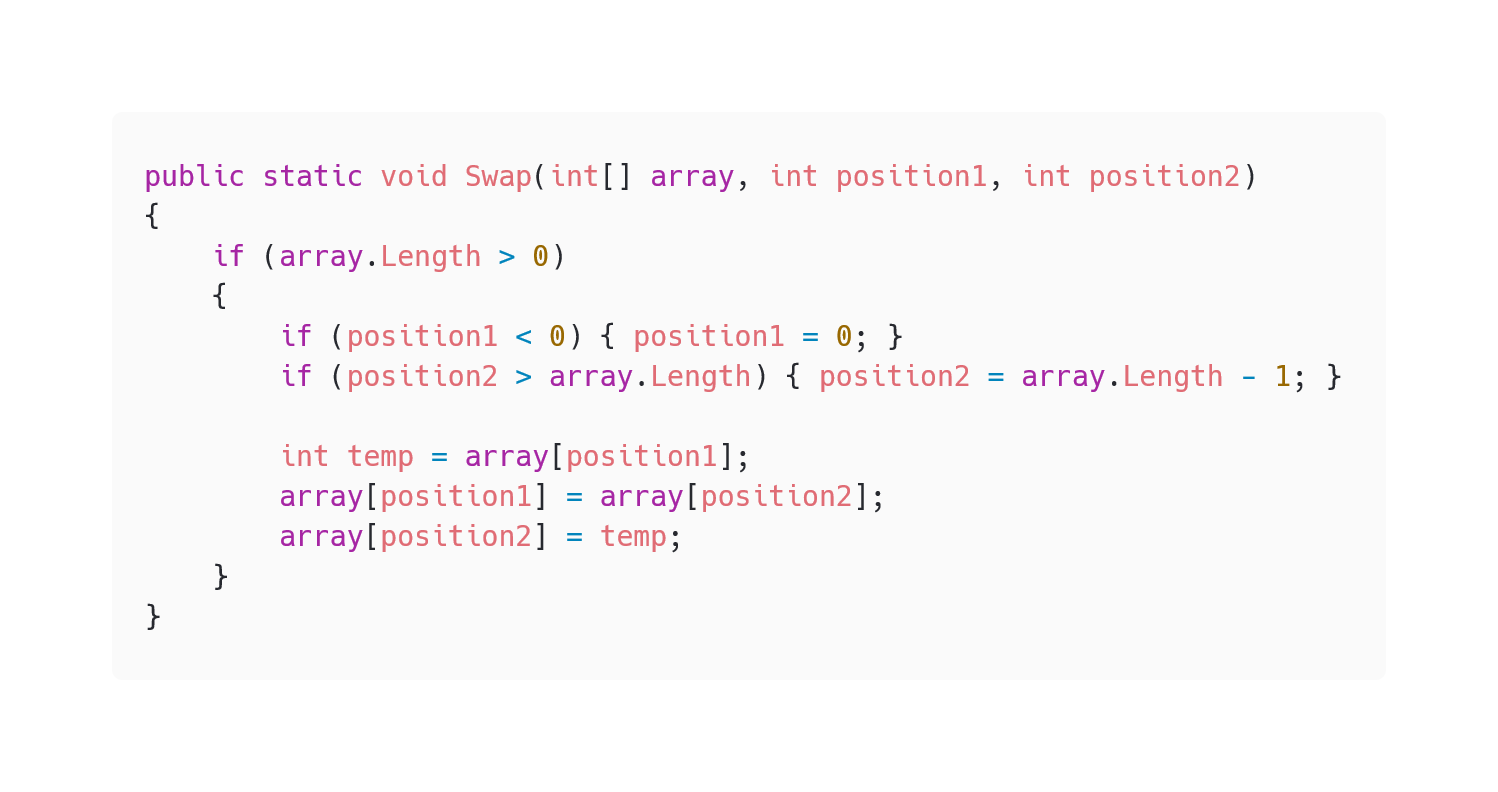
The only two things I changed are:

* The name of the return variable (maxValue 🡪 minValue)
* The operator in the if statement of the for loop (< 🡪 >)



Here we can see the little method which swaps the variables, if the first one is higher as the second one.

# 3 Swap



Explanation:

This method takes 1 array of integers and 2 integer variables. The 2 variables are positions in the array. The goal of the method is to change the array values of these 2 positions. First the method checks if the positions are in the range of the array, if not the lower position will be 0 and the higher position will be the array length. After that, the method switches the values of the positions with the help of a temporary integer variable to store one value.

# 4 ShifLeftByOne

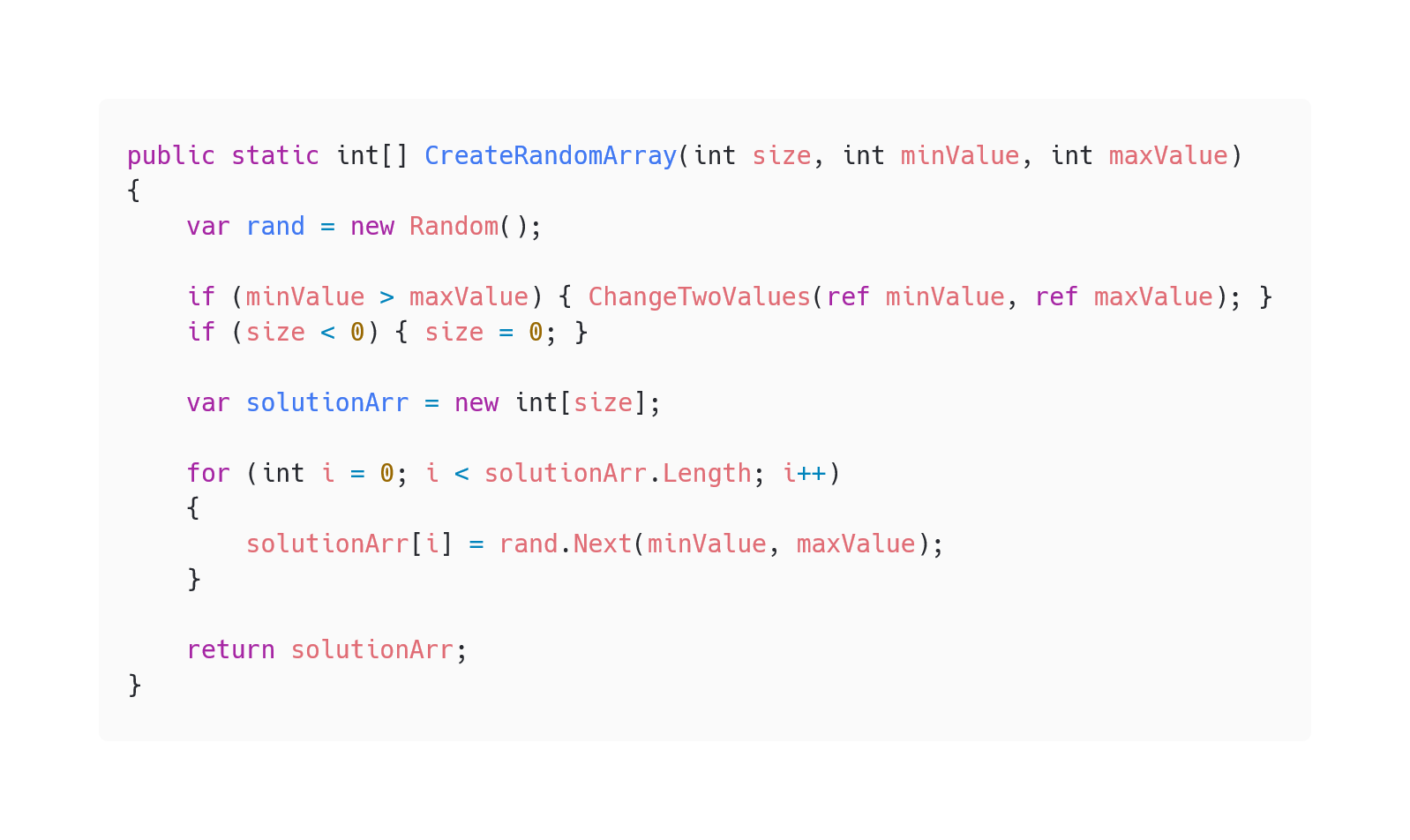


Explanation:

This method takes 1 array of integers and 2 integer variables. The 2 variables are positions in the array. The goal of the method is to shift all elements of the array between the positions to the left by one. The element of position2 should never be changed or shifted. First the method checks again if the first position is lower as the second one, if not they get changed. Again, if the positions are out of bounds, the lower one will be 0 and the higher one will be the array length.

I added an if statement with a for loop which is looping through the elements between the positions. The current element gets the same value as the following one, then the loop goes on until the loop reached the position2 element.

# 5 CreateRandomArray



Explanation:

This method takes 3 integer variables. The first one declares a size and other 2 variables are borders.

The goal of the method is to create a one-dimensional array, filled with random elements between 2 values. First, I made an object from the random class in C#, then I checked if the minValue is lower as the maxValue, if not they get changed. I also implemented if someone makes the size < 0 then the size will always be 0.

After that I created an array with the length of the size variable. To fill the array, I decided to make a for loop which is replacing the current null with a random generated number between the min and max value, then the loop goes on until the solutionArr is filled up.

When the for loop is finished, the method returns the solutionArr.

# 6 CreateRandomMatrix



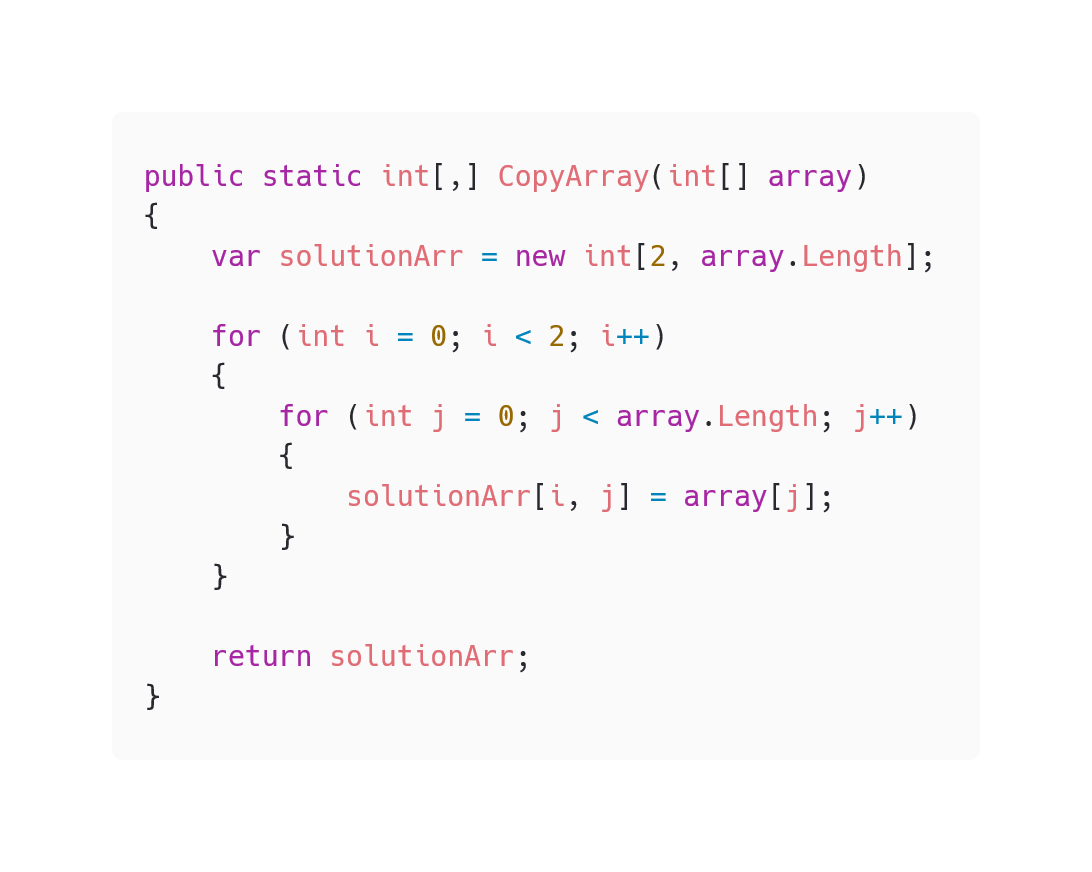
Explanation:

This method takes 4 integer variables. The goal of this method is to create a two-dimensional array.

The elements are again random elements within a min and max Value. Basically, the method checks first the same things like at the CreateRandomArray method, and then we will in the array with a double for loop.

After the array is filled, the method returns it.

# 7 CoppyArray



Explanation:

This method takes an array. The goal of the method is to copy this array in a two-dimensional array.

I solved this the same way like I solved the CreateRandomMatrix method, with a double for loop to fill in all the elements.

After the array is filled, the method returns it.

# 8 FindInSortedArray



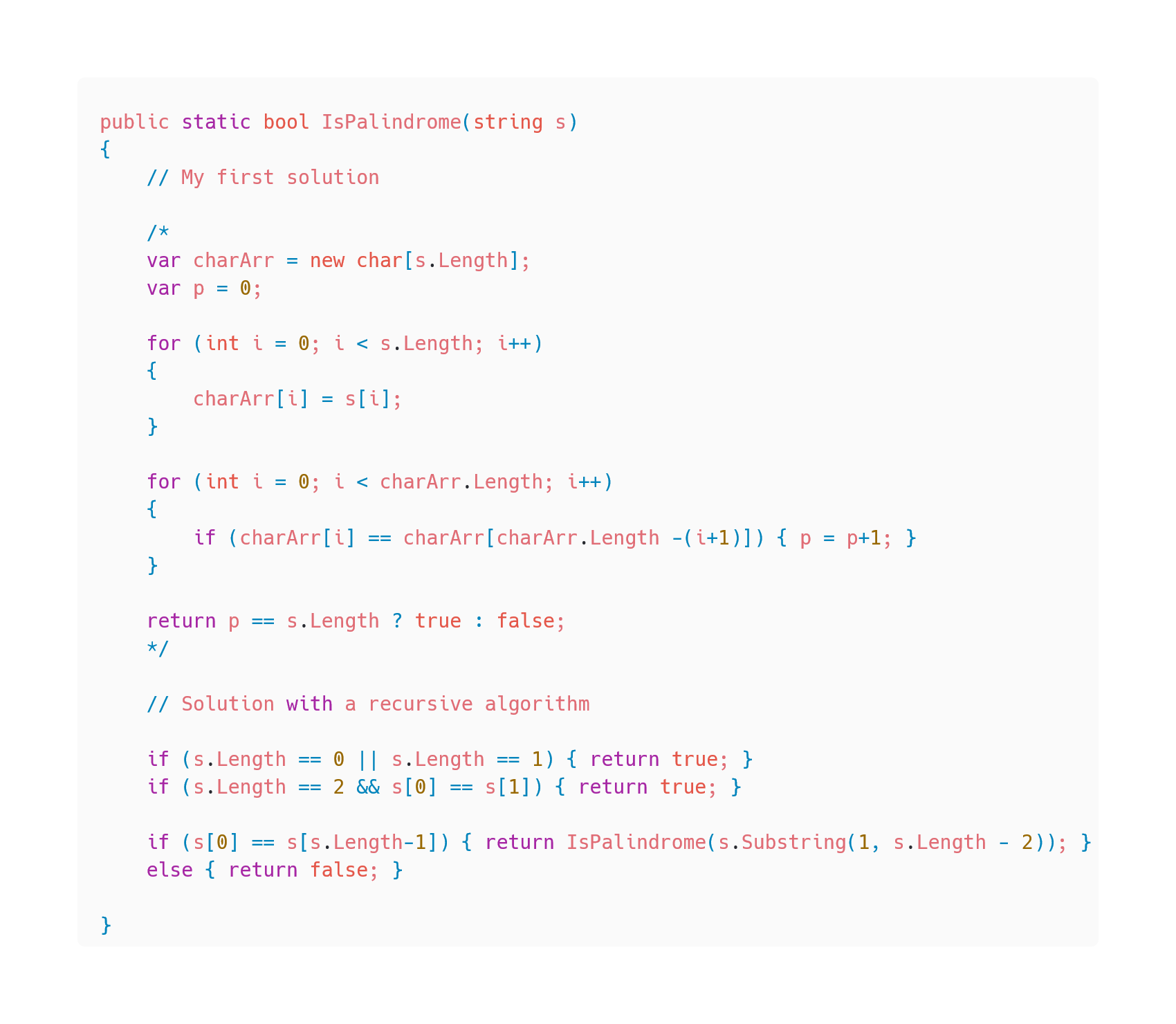
Explanation:

This method takes one array and one integer variable. The goal of this method is, to search the number in this array and return the position of the array where this number is located. If the number doesn’t exist in this array, the method should return -1. I implemented here 2 ways, a linear search solution and a binary search solution.

For the linear search solution, I wrote a simple for loop, looping through the whole array until the number is found, if the loop finds the number, it returns the position of the number, else it returns -1.

For the binary search solution, I implemented 3 variables, one which is the highest point, one which is the lowest point and one variable which is the middle of these 2 points. After that we find a while loop, the loop is not ending until the number is found or the lowest or highest point is equal to the middle point, which means the number could not be found. The loop checks if the number is higher or lower as the middle point, for example if it’s higher, the new lowest point is the current middle point and the new middle point will be determined. The loop keeps going until he finds the number or not.

## **Recursion**



Explanation:

This method takes one string variable. The goal of this method is, to check if the string is a palindrome.

I implemented here 2 ways, my first solution what came in my mind and then a solution with a recursive algorithm like asked in the Task.

In my solution I created a char array and filled it with the letters of the string and after that I looped through the array and checked if the first and last letter is equal, if it’s equal check the second and the second last letter and so on. Every time the loop finds equal letters it counts up a integer variable, if the integer variable has the same value and the string has letters, the string is a palindrome.

For the recursive algorithm the method checks first if there is zero, one or two letters. It returns always true except the 2 letters are not equal. If the string has more as 2 letters, the third if statement will check the first and last letter if they’re equal, if they are the method repeats itself with a new string. The new string is the same as the original one but without the first and last letter. If the original string was “radar” the new method just checks “ada”. This method is repeating itself until the word has zero to two letters or finds an unequal combination. If something is unequal the method returns false, if not, the method returns true which means the string is a palindrome.

## **Object Oriented Programming**