**Activity 1**

The idea there is basically to get distance between numbers the distance will tell how close the numbers are, in this case 2 numbers.

Logic is simple:

Distance is a difference in numbers from two points, example: 3 and 1, the distance would be 2, as we subtract 3-1 = 2 🡨 That is the distance.

However, as we have an array we must use some methods to create it appropriate and somehow efficient.

Array {-3, 7, 2, 4, 8, -19}

Arraysort(array)

Define (int) minDif = Math.abs(array[0]-array[1])

Define (int) valueOne = array[0]

Define (int) valueTwo = array[1]

Define/initialize (int) testVar = 0

For counterIndex = 0; counterIndex less than the length of the array; counterIndex + 1

testVar = Math.abs(array[counterIndex]-array[counterIndex+1]

if testVar less than minDif

minDif = testVar

valueOne = array[counterIndex]

valueTwo = array[counterIndex+1]

define getIndexOne = counterIndex;

define getIndexTwo = counterIndex+1;

display (The closes number are valueOne and ValueTwo and the index are getIndexOne and getIndexTwo

**Activity 2**

We can test the program by searching for possible *bugs*. With whitebox testing we have access to the code, so we can modify the code and check what could happen.

Let’s use the pseudo code:

Array of integer {-3, 7, 2, 4, 8, -19}

Arraysort(array)

Define (int) minDif = Math.abs(array[0]-array[1])

Define (int) minDif = array[0] – array[1] – array[0]

Define (int) valueOne = array[0]

Define (int) valueTwo = array[1]

Define/initialize (int) testVar = 0

For counterIndex = 0; counterIndex less than the length of the array; counterIndex + 1

testVar = Math.abs(array[counterIndex]-array[counterIndex+1]

if testVar less than minDif

minDif = testVar

valueOne = array[counterIndex]

valueTwo = array[counterIndex+1]

define getIndexOne = counterIndex;

define getIndexTwo = counterIndex+1;

display (The closes number are valueOne and ValueTwo and the index are getIndexOne and getIndexTwo

By displaying the red line instead of the black one, this will produce an error which is 🡪 with -19 and -3 which results in a negative number which is less than 0 which breaks the logic to get the closest to 0.