# chapter 11: QUESTIONS

**IMPORTANT: The following questions should be submitted individually to the relevant Practical11 questions within repl.it. The outline class below can be used in Netbeans to work on your solutions. Each question is worth 25% of this exercise so aim to get at least question 1 and question 2 completed.**

**Question 1**

**This question assesses your understanding of the selection statement and value methods**

Complete the divisor method so that it checks if the parameter (op) matches one of the division operators ('/' or '%') and returns the result of applying this operator to parameters a and b. For any other operator the method should return a result of 0.

public static int divisor(int a, char op, int b) { … }

Some examples of calling the method:  
divisor(6, '/', 2) will return 3 as 6 / 2 equals 3

divisor(6, '%', 2) will return 0 as 6 % 2 equals 0

divisor(6, '\*', 2) will return 0 as '\*' is not a divisor

**Submit to repl.it Practical11-q1**

**Question 2**

**This question assesses your understanding of arrays, array indexes and accessing array elements**

Complete the firstLastSame method so that it checks if the first and last elements in the array parameter (data) are the same (equal). If so then the method should return true otherwise return false

public static boolean firstLastSame(int[] data) { … }

Some examples of calling the method:

int[] a = { 1, 2, 3 }; int[] b = { 1, 2, 3, 1 };

firstLastSame(a) would return false as first element (1) and last element (3) are not equal

firstLastSame(b) would return true as first element (1) and last element (1) are equal

**Submit to repl.it Practical11-q2**

**Question 3**

**This question assesses your understanding of counter-controlled loops**

Complete the method countBelow so that it counts the number of elements in the parameter array (data) are less than the parameter (val) and returns this count as the method result

public static int countBelow(int val, int[] data) { … }

Some examples:

int[] a = {10, 2, 13, 4};

countBelow(4, a) will return 1 as there is only one number (2) in this array less than 4

countBelow(2, a) will return 0 as there are no numbers in this array less than 2

**Submit to repl.it Practical11-q3**

**Question 4**

**This is a challenging question that tests your ability to design an algorithm and not everyone will work out the solution – and that’s ok ☺**

Complete the method inOrder so that it verifies that each element in the array is less than or equal to the next element i.e. the elements are in sorted order

public static boolean inOrder(int[] data) { … }

Some examples of calling the method:

int[] a = {1, 2, 3 }; int[] b = { 1, 2, 3, 1 };

inOrder(a) would return true

inOrder(b) would return false as the last element (1) is out of order

The basic algorithm is:

1. loop through each element in the array (apart from the last element)\*\*

2. in the loop, compare current element data[i] with the next element data[i+1} and return false if they are out of order.

2. - note they are out of order if (data[i] > data[i+1]) - and by returning false the loop is terminated early.

4. if we reach the end of the loop and have not returned early then the elements must be in order so return true

\*\* Why do we not loop to last element in array. Well remember that when comparing an element with the next element (data[i] with data[i+1]), you must ensure that your index (i) only goes to the second last element in the array (if you go to the last element then data[i+1] will cause an out of bounds error).

You can run the question4() method to check if it prints the results as expected.

**Submit to repl.it Practical11-q4**

public class Practical11 {

// ---------- COMPLETE THESE OUTLINE METHODS -------------

// Question 1

public static int divisor(int a, char op, int b) {

// complete this method

}

// Question 2

public static boolean firstLastSame(int[] data) {

// complete this method

}

// Question 3

public static int countBelow(int val, int[] data) {

// complete this method

}

// Question 4

public static boolean inOrder(int[] data) {

// complete this method

}

// main method

public static void main(String[] args) {

// CALL THE TEST METHOD FOR EACH QUESTION TO VERIFY ITS OPERATION

question1();

//question2();

//question3();

//question4();

}

// ---------- DON’T MODIFY CODE BELOW THIS LINE ----------

// THESE ACTION METHODS PERFORM TESTS on EACH QUESTION ABOVE

public static void question1() {

System.out.println(" \nDivisor Tests" );

System.out.println(" =============" );

System.out.println( "4 / 2 == " + divisor(4, '/', 2) );

System.out.println( "3 % 2 == " + divisor(3, '%', 2) );

System.out.println( "3 $ 2 == " + divisor(3, '$', 2) );

}

public static void question2() {

int[] a = {1, 2, 3, 1};

int[] b = {1, 2, 3, 4};

System.out.println( "\nFirstLast Same Tests" );

System.out.println( "====================" );

System.out.println( arrayToString(a) + " == " + firstLastSame(a) );

System.out.println( arrayToString(b) + " == " + firstLastSame(b) );

}

public static void question3() {

int[] a = {1, 2, 3, 1};

int[] b = {1, 2, 3, 4};

System.out.println( "\nCount Below 4" );

System.out.println( "=============" );

System.out.println( arrayToString(a) + " == " + countBelow(4, a) );

System.out.println( arrayToString(b) + " == " + countBelow(4, b) );

}

public static void question4() {

int[] a = {1, 2, 3, 1};

int[] b = {1, 2, 3, 4};

System.out.println( "\nInOrder Tests" );

System.out.println( "=============" );

System.out.println( arrayToString(a) + " == " + inOrder(a) );

System.out.println( arrayToString(b) + " == " + inOrder(b) );

}

// utility value method that returns an array as a string for printing

public static String arrayToString(int[] data) {

String r = "[ ";

for (int i=0; i<data.length; i = i + 1) {

r = r + data[i] + " ";

}

return r + "]";

}

}