# Exercises: Methods, Debugging and Troubleshooting Code

Problems for exercises and homework for the [“Programming Fundamentals” course @ SoftUni](https://softuni.bg/courses/programming-fundamentals).

You can check your solutions here: <https://judge.softuni.bg/Contests/305/Methods-and-Debugging-Excercises>.

## Hello, Name!

Write a **method** that receives a name as **parameter** and prints on the console. “Hello, <name>!”

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Peter | Hello, Peter! |

## Max Method

Create a method GetMax(int a, int b), that returns the **largest** of two numbers. Write a program that reads **three numbers** from the console and **prints** the **biggest** of them. Use the GetMax(…) method you just created.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 1  2  3 | 3 |  | -100  -101  -102 | -100 |

## English Name оf the Last Digit

Write a **method** that returns the **English name** of the last digit of a given number. Write a program that reads an integer and prints the returned value from this method.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 1024 | four |  | 512 | two |

## Numbers in Reversed Order

Write a method that **prints the digits** of a given decimal number in a **reversed order**.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 256 | 652 |  | 1.12 | 21.1 |

## Fibonacci Numbers

Define a method **Fib(n)** that calculates the nth [Fibonacci number](https://en.wikipedia.org/wiki/Fibonacci_number). Examples:

|  |  |
| --- | --- |
| **n** | **Fib(n)** |
| 0 | 1 |
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 5 |
| 5 | 8 |
| 6 | 13 |
| 11 | 144 |
| 25 | 121393 |

1. **Prime Checker**

Write a Boolean method **IsPrime(n)** that check whether a given integer number **n** is [prime](https://en.wikipedia.org/wiki/Prime_number). Examples:

|  |  |
| --- | --- |
| **n** | **IsPrime(n)** |
| 0 | false |
| 1 | false |
| 2 | true |
| 3 | true |
| 4 | false |
| 5 | true |
| 323 | false |
| 337 | true |
| 6737626471 | true |
| 117342557809 | false |

1. **\* Primes in Given Range**

Write a method that calculates **all prime numbers in given range** and returns them as list of integers:

|  |
| --- |
| static List<int> FindPrimesInRange(startNum, endNum)  {  …  } |

Write a method to **print a list of integers**. Write a program that enters two integer numbers (each at a separate line) and prints all primes in their range, separated by a comma.

### Examples

|  |  |
| --- | --- |
| **Start and End Number** | **Output** |
| 0  10 | 2, 3, 5, 7 |
| 5  11 | 5, 7, 11 |
| 100  200 | 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199 |
| 250  950 | 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947 |
| 100  50 | *(empty list)* |

## Center Point

You are given the coordinates of two points on a [Cartesian coordinate system](https://en.wikipedia.org/wiki/Cartesian_coordinate_system) - X1, Y1, X2 and Y2. **Create a method** that prints the point that is closest to the center of the coordinate system (0, 0) in the format (X, Y). If the points are on a same distance from the center, print only the first one.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2  4  -1  2 | (-1, 2) |

## Longer Line

You are given the coordinates of four points in the 2D plane. The first and the second pair of points form two different lines. Print the longer line in format "(X1, Y1)(X2, Y2)" starting with the point that is closer to the center of the coordinate system (0, 0) (You can reuse the method that you wrote for the previous problem). If the lines are of equal length, print only the first one.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2  4  -1  2  -5  -5  4  -3 | (4, -3)(-5, -5) |

## Cube Properties

Write a program that can calculate the length of the face diagonals, space diagonals, volume and surface area of a **cube** (<http://www.mathopenref.com/cube.html>) by a given side. On the first line you will get the side of the cube. On the second line is given the parameter (**face**, **space**, **volume** or **area**).

Output should be rounded to the second digit after the decimal point:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5  face | 7.07 |
| 5  volume | 125.00 |

## Geometry Calculator

Write a program that can **calculate the area** of **four different geometry figures** - triangle, square, rectangle and circle.

**On the first line** you will get the **figure type**. Next you will get parameters for the chosen figure, **each on a different line**:

* Triangle - side and height
* Square - side
* Rectangle - width and height
* Circle - radius

The output should be rounded to the second digit after the decimal point:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| triangle  3  6 | 9.00 |
| rectangle  4  5 | 20.00 |

## Master Numbers

A master number is an integer that holds the following properties:

* Is **symmetric** (palindrome), e.g. 5, 77, 282, 14341, 9553559.
* Its **sum of digits is divisible by 7**, e.g. 77, 313, 464, 5225, 37173.
* Holds at least **one even digit**, e.g. 232, 707, 6886, 87578.

Write a program to **print all master numbers** in the range [1…**n**].

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 600 | 232  383  464  545 |  | 5000 | 232  383  464  545  626  696  707  858  1661  2552  3443  4334 |

### Hints

1. Write 3 utility methods:

* IsPalindrome(int num)
* SumOfDigits(int num)
* ContainsEvenDigit(int num)

1. Loop through all numbers in range [1…n] and check every number with the helper methods.

## \* Factorial

Write a program that calculates and prints the n! for any n in the range [1…1000].

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 5 | 120 |  | 100 | 93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864000000000000000000000000 |

### Hints

Use the class BigIntegerfrom the built-in .NET library System.Numerics.dll.

1. First add reference to System.Numerics.dll.





1. Import the namespace “System.Numerics”:



1. Use the type BigInteger instead of long or decimal to keep the factorial value:



## Factorial Trailing Zeroes

Create a program that counts the trailing zeroes of the factorial of a given number.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 5 | 1 | 5! = 12**0** -> One trailing zero |
| 100 | 24 | 100! = 93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864**000000000000000000000000** -> 24 trailing zeroes |

### Hints

1. You may use your solution from the previous problem. Add additional method that counts and returns the number of zeroes a number has.