Lab: Functions Advanced

Problems for in-class lab for the Python Advanced Course @SoftUni. Submit your solutions in the SoftUni judge system at https://judge.softuni.org/Contests/1838.

1. Multiplication Function

Write a function called multiply that can receive any quantity of numbers (integers) as different parameters and returns the result of the multiplication of all of them. Submit only your function in the Judge system.

Examples

Test Code	Output
<pre>print(multiply(1, 4, 5)) print(multiply(4, 5, 6, 1, 3)) print(multiply(2, 0, 1000, 5000))</pre>	20 360 0

2. Person Info

Write a function called get_info that receives a name, an age, and a town and returns a string in the format: "This is {name} from {town} and he is {age} years old". Use dictionary unpacking when testing your function. Submit only the function in the judge system.

Examples

Test Code	Output	
<pre>print(get_info(**{"name": "George", "town": "Sofia", "age": 20}))</pre>	This is George from Sofia and he is 20 years old	

3. Cheese Showcase

White a function called **sorting_cheeses** that receives **keywords arguments**:

- The key represents the name of the cheese
- The value is a list of quantities (integers) of the pieces of the given cheese

The function should return the cheeses and their pieces' quantities sorted by the number of pieces of a cheese kind in descending order. If two or more cheeses have the same number of pieces, sort them by their names in ascending order (alphabetically). For each kind of cheese, return their pieces quantities in descending order.

For more clarifications, see the examples below.

Input	Output
print(Camembert
sorting_cheeses(500
Parmesan=[102, 120, 135],	430
Camembert=[100, 100, 105, 500, 430],	105
Mozzarella=[50, 125],	100
)	100
	Parmesan
	135
	120
	102















	Mozzarella 125 50
<pre>print(sorting_cheeses(Parmigiano=[165, 215], Feta=[150, 515], Brie=[150, 125]))</pre>	Brie 150 125 Feta 515 150 Parmigiano 215

4. Rectangle

Create a function called **rectangle()**. It must have two parameters - **length** and **width**.

First, you need to check if the given arguments are integers:

If one/ both of them is/ are **NOT** an integer/s, **return** the string **"Enter valid values!"**

Create two inner functions:

- area() returns the area of the rectangle with the given length and width
- perimeter() returns the perimeter of the rectangle with the given length and width

In the end, the **rectangle** function should **return** a string containing the **area** and the **perimeter** of a rectangle in the following format:

```
"Rectangle area: {ract_area}
```

Rectangle perimeter: {rect_perim}"

Examples

Test Code	Output
<pre>print(rectangle(2, 10))</pre>	Rectangle area: 20 Rectangle perimeter: 24
<pre>print(rectangle('2', 10))</pre>	"Enter valid values!"

5. Recursive Power

Create a recursive function called recursive_power() which should receive a number and a power. Using **recursion, return** the result of **number** ** **power**. Submit only the function in the judge system.

Examples

Test Code	Output
<pre>print(recursive_power(2, 10))</pre>	1024
<pre>print(recursive_power(10, 100))</pre>	100000000000000000000000000000000000000











6. Operate

Write a function called operate that receives an operator ("+", "-", "*" or "/") as first argument and multiple numbers (integers) as additional arguments (*args). The function should return the result of the operator applied to all the numbers. For more clarification, see the examples below.

Submit only your function in the Judge system.

Note: Be careful when you have multiplication and division

Examples

Test Code	Output	Comment
<pre>print(operate("+", 1, 2, 3))</pre>	6	1 + 2 + 3 = 6
<pre>print(operate("*", 3, 4))</pre>	12	3 * 4 = 12















