# Lab: Functions Advanced

Problems for in-class lab for the [Python Advanced Course @SoftUni](https://softuni.bg/courses/python-advanced).

Submit your solutions in the SoftUni judge system at <https://judge.softuni.org/Contests/1838>.

## 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** |
| print(multiply(1, 4, 5))  print(multiply(4, 5, 6, 1, 3))  print(multiply(2, 0, 1000, 5000)) | 20  360  0 |

## 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** |
| print(get\_info(\*\*{"name": "George", "town": "Sofia", "age": 20})) | This is George from Sofia and he is 20 years old |

## 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** **with 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(  sorting\_cheeses(  Parmesan=[102, 120, 135],  Camembert=[100, 100, 105, 500, 430],  Mozzarella=[50, 125],  )  ) | Camembert  500  430  105  100  100  Parmesan  135  120  102  Mozzarella  125  50 |
| print(  sorting\_cheeses(  Parmigiano=[165, 215],  Feta=[150, 515],  Brie=[150, 125]  )  ) | Brie  150  125  Feta  515  150  Parmigiano  215  165 |

## Character Permutations

Write a program that reads a **single** **string** and **prints** **all possible combinations** of the **characters** in that string. Submit your solution in the judge system.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| abc | abc  acb  bac  bca  cba  cab |
| 12 | 12  21 |

## Chairs

Write a program that receives **names** on the first line (separated by comma and space **", "**) and a number of **chairs** on the second line (an **integer**). The chairs will always be less than the people. Find all the ways to fit those people on the chairs. **Print** each combination on a separate line.

***Note: In the example below, "Peter, George" is the same as "George, Peter", so we only print the first combination***

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Peter, George, Amy  2 | Peter, George  Peter, Amy  George, Amy |
| Mariya, Emilly, Tom, Bob  1 | Mariya  Emilly  Tom  Bob |

## 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** |
| print(operate("+", 1, 2, 3)) | 6 | 1 + 2 + 3 = 6 |
| print(operate("\*", 3, 4)) | 12 | 3 \* 4 = 12 |

## Expressions

Write a program that generates all the possible **expressions** and their **results** between a **given list of numbers** (integers) using only the **+** and **-** operators. **Print** them on the console, as shown in the example.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1, 2 | +1+2=3  +1-2=-1  -1+2=1  -1-2=-3 |
| 1, 1, 1, 1 | +1+1+1+1=4  +1+1+1-1=2  +1+1-1+1=2  +1+1-1-1=0  +1-1+1+1=2  +1-1+1-1=0  +1-1-1+1=0  +1-1-1-1=-2  -1+1+1+1=2  -1+1+1-1=0  -1+1-1+1=0  -1+1-1-1=-2  -1-1+1+1=0  -1-1+1-1=-2  -1-1-1+1=-2  -1-1-1-1=-4 |