# Lab: Modules

The following problem descriptions **do not require** **submissions** to the Judge System.

## Calculate Logarithm

Write a program that **prints** the calculated logarithm of **any** given number

### Input

* On the **first** line, you will **receive** the **number** (an integer)
* On the **second** line, you will **receive** a number, which is the **logarithm base**. It can be **either** a number or the word "natural"

The **output** should be **formatted** to the **2nd** decimal digit

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 10  natural | 2.30 |
| 10  10 | 1.00 |

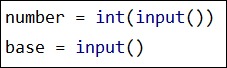
### Hints

Use the **math** module. You can read more about it [**here**](https://www.tutorialsteacher.com/python/math-module)

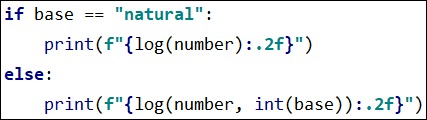
1. **Import** the module:



1. **Read** the variables:



1. Implement the logic:



## ASCII Art

Write a program that **encrypts** **given words** by using the characters: **"-|\_/\()"** to structure the word. Use the **pyfiglet** module. You can read more about it [**here**](https://www.geeksforgeeks.org/python-ascii-art-using-pyfiglet-module/)

### Directions

1. First you need to **install** the module that we will be using. To install it go to **Setting** --> **Project** **<your\_project\_name>** --> **Project Interpreter** --> + --> **search for pyfiglet** --> **install package**.
2. Import the module
3. Implement the logic. We will be using the **figlet\_format** method

### Examples

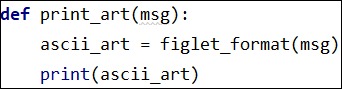
|  |  |
| --- | --- |
| **Input** | **Output** |
| Hello World! |  |
| Python 3.8 |  |

### Hints

1. First, we need to **import** the module:



1. Then we implement the logic:



1. Lastly, we **print** the message.

## Triangle

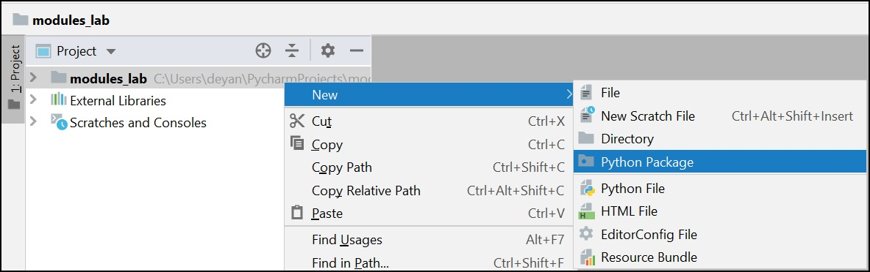
Create a **module** for printing a triangle. You will **receive** an integer **number** which is the **size** of the triangle.

### Examples

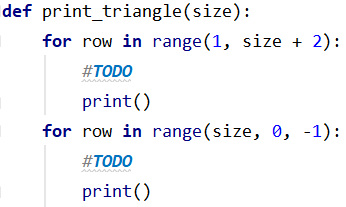
|  |  |
| --- | --- |
| **Input** | **Output** |
| 3 | 1  1 2  1 2 3  1 2  1 |
| 4 | 1  1 2  1 2 3  1 2 3 4  1 2 3  1 2  1 |

### Hints

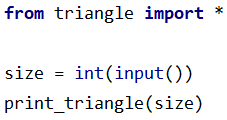
1. We'll start with **creating a package** called a triangle



1. Then we implement the logic. You can use 2 nested loops, **one** starting from 1 and **another** starting from our limit, each printing a line per cycle



1. And finally, **import** the module:



## Mathematical Operations

Create a **module** that does basic calculations. You will **receive** 2 numbers and a sign between them all in **one string**

### Input

You will **receive** a single **string** in the following **format**

**"{number1} {sign} {number2}"**

* **number1** - a float number in the range (0.0, 1000.0)
* **sign** -a char that can be  
  '/' - divide the first number by the second  
  '\*' - multiply the 2 numbers   
  '-' - subtract the first number from the second  
  '+' - add the 2 numbers  
  '^' - raise the first number to the second
* **number2** -an integer number in the range (0, 1000)

### Output

**Print** only the **result** of the operation

The result should be **formatted** to the **second** decimal point

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2.5 \* 2 | 5.00 |
| 6.66 ^ 2 | 44.35 |
| 36.66 / 6 | 6.11 |

## Fibonacci Sequence

Create a **module** that can **create** a **Fibonacci sequence** up to a number (**count of numbers in the sequence**) and **print** them, separating them with a **single space**. The module should also be able to **locate a specific number** in the sequence. You can **read** more about the Fibonacci sequence [**here**](https://en.wikipedia.org/wiki/Fibonacci_number).

You will be receiving **commands** until the "**Stop**" command. The commands are:

* "**Create Sequence {count}"**. Create a series of numbers up to a specific **count** and **print** them in the **following format**:  
   **"{n1} {n2} … {n}"**
* "**Locate {number}"**  
  Check if the sequence **contains** the number. If it **finds** the number, it should **print**:

**"The number - {number} is at index {index}"**

And if it **doesn't find** it:

"**The number {number} is not in the sequence**"

### Input

* You will be receiving **commands** until the "**Stop**" command. All inputs will be **valid**.

### Output

* **Print** the **output** of every **command** in the **format** **described** **above**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Create Sequence 10  Locate 13  Create Sequence 3  Locate 10  Stop | 0 1 1 2 3 5 8 13 21 34  The number - 13 is at index 7  0 1 1  The number 10 is not in the sequence |