# More Exercises: Functions

Additional exercises for the [Python Fundamentals Course @SoftUni](https://softuni.bg/trainings/3368/python-fundamentals-may-2021).   
Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1729>

***Note: All the exercises are excluded from your homework!***

## Data Types

Write a function that, depending on the first line of the input, reads an **int**, **double** or **string**.

* If the data type is int, multiply the number by 2.
* If the data type is real, multiply the number by 1.5 and format the result to the second decimal point.
* If the data type is string, surround the input with "**$**".

Print the result on the console.

input\_type = input()  
input\_data = input()  
result\_int = 0  
result\_real = 0  
result\_str = ''  
  
if input\_type == 'int':  
 input\_data = int(input\_data)  
 result\_int = input\_data \* 2  
 print(result\_int)  
elif input\_type == 'real':  
 input\_data = float(input\_data)  
 result\_real = input\_data \* 1.5  
 print(f"{result\_real:.2f}")  
elif input\_type == 'string':  
 input\_data = str(input\_data)  
 result\_str = '$' + input\_data + '$'  
 print(result\_str)

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| int  5 | 10 |
| real  2 | 3.00 |
| string  hello | $hello$ |

### Hint

Try to solve the problem using only one method with different overloads.

## Center Point

You will be given the coordinates of two points on a [Cartesian coordinate system](https://en.wikipedia.org/wiki/Cartesian_coordinate_system) - X1, Y1, X2 and Y2. Write a **function** that prints the point which 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. The resulting coordinates must be **formatted** to the **lower integer.**

import math  
  
x1 = float(input())  
y1 = float(input())  
x2 = float(input())  
y2 = float(input())  
  
diag\_1 = math.sqrt(x1\*\*2 + y1\*\*2)  
diag\_2 = math.sqrt(x2\*\*2 + y2\*\*2)  
  
if diag\_1 <= diag\_2:  
 print(f"({math.floor(x1)}, {math.floor(y1)})")  
else:  
 print(f"({math.floor(x2)}, {math.floor(y2)})")

### Examples

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

## Longer Line

You will be given the coordinates of four points. The first and the second pair of points form two different lines. Create a function which prints **the longer line** in format **"({X1}, {Y1})({X2}, {Y2})"** starting from the point which 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. The resulting coordinates must be **formatted** to the **lower integer**.

import math  
  
  
def get\_distance(a, b, c, d):  
 a = math.pow(a, 2)  
 b = math.pow(b, 2)  
 c = math.pow(c, 2)  
 d = math.pow(d, 2)  
 return math.sqrt(a + b + c + d)  
  
  
def get\_calculation(a, b):  
 a = math.pow(a, 2)  
 b = math.pow(b, 2)  
 return math.sqrt(a + b)  
  
  
x1 = float(input())  
y1 = float(input())  
x2 = float(input())  
y2 = float(input())  
x3 = float(input())  
y3 = float(input())  
x4 = float(input())  
y4 = float(input())  
  
first\_line = get\_distance(x1, y1, x2, y2)  
second\_line = get\_distance(x3, y3, x4, y4)  
  
if first\_line >= second\_line:  
 if get\_calculation(x1, y1) <= get\_calculation(x2, y2):  
 print(f"({math.floor(x1)}, {math.floor(y1)})({math.floor(x2)}, {math.floor(y2)})")  
 else:  
 print(f"({math.floor(x2)}, {math.floor(y2)})({math.floor(x1)}, {math.floor(y1)})")  
else:  
 if get\_calculation(x3, y3) <= get\_calculation(x4, y4):  
 print(f"({math.floor(x3)}, {math.floor(y3)})({math.floor(x4)}, {math.floor(y4)})")  
 else:  
 print(f"({math.floor(x4)}, {math.floor(y4)})({math.floor(x3)}, {math.floor(y3)})")

### Examples

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

## Tribonacci Sequence

In the **Tribonacci sequence**, every number is formed by the **sum of the previous 3**.

You are given a number num. Write a function which printsnumbers from the Tribonacci sequence on one line separated by a single space, starting from 1. The input comes as a parameter named num. The value num will always be positive integer.

numbers = int(input())  
number\_list = [1]  
  
for i in range(numbers - 1):  
 sum\_ot\_number\_list = sum(number\_list[-3:])  
 number\_list.append(sum\_ot\_number\_list)  
  
nums\_to\_str = [str(num) for num in number\_list]  
print(" ".join(nums\_to\_str))

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 4 | 1 1 2 4 |  | 8 | 1 1 2 4 7 13 24 44 |

## Multiplication Sign

You are given a number num1, num2 and num3. Write a program that finds if num1 \* num2 \* num3 (the product) is **negative**, **positive or zero**. Try to do this **WITHOUT** multiplying the 3 numbers.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 2  3  -1 | negative |  | 2  3  1 | positive |

def get\_result(a, b, c):  
 list = [a, b, c]  
 negatives = 0  
 for i in range(len(list)):  
 if "-" in list[i]:  
 negatives += 1  
 if negatives % 2 != 0:  
 return "negative"  
 elif "0" in list:  
 return "zero"  
 else:  
 return "positive"  
  
  
first\_num = input()  
second\_num = input()  
third\_num = input()  
  
print(get\_result(first\_num, second\_num, third\_num))