Exercises: Data Types and Variables

Problems for exercise and homework for the "JS Fundamentals" Course @ SoftUni.

Submit your solutions in the SoftUni judge system at: https://judge.softuni.org/Contests/1229

1. Sum Digits

Write a **function**, which will be given a single **number**. Your task is to find the **sum** of its digits.

Examples

Input	Output
245678	32
97561	28
543	12

2. Chars to String

Write a **function**, which receives **3 parameters**. Each parameter is a single character. Combine all the characters into **one** string and print it on the console.

Examples

Input	Output
'a', 'b', 'c'	abc
'%', '2', 'o'	%2o
'1', '5', 'p'	15p

3. Town Info

You will be given **3 parameters**. The first parameter will be the name of the **town** (string), the second – the **population** (number), and the third the **area** (number). Print the result in the following format:

"Town {town name} has population of {population} and area {area} square km."

Examples

Input	Output
'Sofia', 1286383, 492	Town Sofia has population of 1286383 and area 492 square km.

'Plovdiv', 1481353, 512	Town Plovdiv has population of 1481353 and area 512 square km.

4. Convert Meters to Kilometres

You will be given a **number** that will be the distance in **meters**. Write a program that converts **`meters`** to **`kilometers`** formatted to the **second decimal** point.

Examples

Input	Output
1852	1.85
798	0.80

5. Pounds to Dollars

Write a **function** that converts British **pounds** to **dollars** formatted to the **3rd decimal point**.

- 1 British Pound = 1.31 Dollars

Examples

Input	Output
80	104.800
39	51.090

6. Reversed Chars

Write a program that takes **3 parameters** (characters) and prints them in **reversed order** with a space between them.

Examples

Input	Output
'A', 'B', 'C'	C B A
'1', 'L', '&'	& L 1

7. Lower or Upper

Write a **function** that prints whether a given character is **upper-case** or **lower-case**.

Examples

Input	Output
'L'	upper-case

`'f'`	`lower-case`

8. *Calculator

Write a **function** that receives 3 parameters: a **`number`**, an **`operator`** (string), and **`another number`**.

The **operator** can be: **`'+', '-', '/', '*'.`** Print the result of the calculation on the console formatted to the **second decimal** point.

Examples

Input	Output

`5,`
`'+',`
`10`
	`15.00`

`25.5,`
`'-',`
`3`
	`22.50`

9. *Gladiator Expenses

As a gladiator, Peter has to repair his broken equipment when he loses a fight. His equipment consists of a helmet, sword, shield, and armor. You will receive Peter`s **lost fights count**.

- Every **second** lost game, his helmet is broken.
- Every **third** lost game, his sword is broken.
- When both **his sword and helmet are broken** in the same lost fight, his **shield also breaks**.
- **Every second time**, when his shield brakes, his **armor** also needs to be repaired.

You will receive the price of each item in his equipment. Calculate his expenses for the year for renewing his equipment.

Input / Constraints

You will receive 5 parameters to your function:

- The first parameter - **lost fights count** - is an integer in the range **[0, 1000]**.
- The second parameter - **helmet price** - is the floating-point number in the range **[0, 1000]**.
- The third parameter - **sword price** - is the floating-point number in the range **[0, 1000]**.
- The fourth parameter - **shield price** - is the floating-point number in the range **[0, 1000]**.
- The fifth parameter - **armor price** - is the floating-point number in the range **[0, 1000]**.

Output

- As output you must print Peter`s total expenses for new equipment rounded to the second decimal point: **`"Gladiator expenses: {expenses} aureus"`**
- Allowed working **time** / **memory**: **100ms** / **16MB**.

Examples

Input	Output	Comment

`7,`
`2,`
`3,`
`4,`
`5`
	`Gladiator expenses: 16.00 aureus`	`Trashed helmet -> 3 times`
		`Trashed sword -> 2 times`
		`Trashed shield -> 1 time`
		`Total: 6 + 6 + 4 = 16.00 aureus;`

`23,`
`12.50,`
`21.50,`
`40,`
`200`
	`Gladiator expenses: 608.00 aureus`	

10. *Spice Must Flow

Spice is Love, Spice is Life. And most importantly, Spice must flow. It must be extracted from the scorching sands of Arrakis, under the constant threat of giant sandworms. To make the work as efficient as possible, the Duke has tasked you with the creation of management software.

Write a program that calculates the **total amount** of spice that can be extracted from a source. The source has a **starting yield**, which indicates how much spice can be mined on the **first day**. After it has been mined for a day, the **yield drops** by 10, meaning on the second day it'll produce 10 less spice than on the first, on the third day 10 less than on the second, and so on (see examples). A source is considered profitable only while its yield is **at least** 100 – when less than 100 spices are expected in a day, abandon the source.

The mining crew **consumes** 26 spices **every day** at the end of their shift and **an additional** 26 after the mine has been exhausted. Note that the workers **cannot** consume more spice than there is in storage.

When the operation is complete, print on the console on two separate lines how many **days** the mine has operated and the **total** amount of spice extracted.

Input

You will receive a number, representing the **starting yield** of the source.

Output

Print on the console on two separate lines how many **days** the mine has operated and the **total amount** of spice extracted.

Constraints

- The starting yield will be a **number** within range [0…228].

Examples

Input	Output	Explanation
`111`	2 134	**Day 1** we extract 111 spices and at the end of the shift, the workers consume 26, leaving 85. The yield drops by 10 to 101. **On day 2** we extract 101 spices, the workers consume 26, leaving 75. The total is 160 and the yield has dropped to 91. **Since** the expected yield is less than 100, we abandon the source. The workers take another 26, leaving 134. The mine has operated for 2 days.
`450`	36 8938	