# Exercises: Data Types and Variables

Submit your solutions in the SoftUni judge system at: <https://alpha.judge.softuni.org/contests/data-types-and-variables-exercise/1229>

## 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 |

## ASCII Values of Reversed Characters

Write a **function**, which receives **3 parameters**. Each parameter is a single character. Each parameter is a **single character**. **Read** the characters and print them on **one line**, in **reverse** order.

On the next line, print the **ASCII values** of **each character** in the reversed string, separated by a **space**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 'a',  'b',  'c' | cba  99 98 97 |
| '%',  '2',  'o' | o2%  111 50 37 |
| '1',  '5',  'p' | p51  112 53 49 |

## 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).

First, **validate** the input data – the town **name** must be **at** **least 3 characters** long, and the **population** and the **area** must be **positive numbers**.

* If the town **name** is **too short**, print the following message:

**"****Town name must be at least 3 characters!"**

* If the **population** or the **area** is **not** a **positive** number, print the following message:

**"{Population/Area} must be a positive number!"**

* If **all** the **data** is **valid**, print it 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. |
| 'LA',  1481353,  512 | Town name must be at least 3 characters! |
| 'Plovdiv',  -45000,  100 | Population must be a positive number! |
| 'Ka',  3600,  -50 | Town name must be at least 3 characters!  Area must be a positive number! |

## Convert Distance

You will be given a **number** that will be the distance in **meters**. Write a program that converts **meters** to **kilometers** and then **kilometers** to **miles**. Print the output in the following format:

* **"{meters} meters is equal to {kilometers} kilometers."**
* **"{kilometers} kilometers is equal to {miles} miles."**

Format the **miles** to **two places** after the decimal point.

**Hint**: **1** kilometer = **0.621371** miles

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1852 | 1852 meters is equal to 1.852 kilometers.  1.852 kilometers is equal to 1.15 miles. |
| 798 | 798 meters is equal to 0.798 kilometers.  0.798 kilometers is equal to 0.50 miles. |

## 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 |

## Reversed String

Write a **function** that receives 1 parameter – a **string**, and prints it **reversed**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 'Information' | noitamrofnI |
| 'star' | rats |
| 'racecar' | racecar |

**Side quest**: did you notice that the **reverse string** in the **last** example looks **identical** to the **original** string? Use your **googling skills** to find out what this phenomenon is called and learn more about it.

## 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 |

## \*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 |

## \*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 |  |

## \*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 |  |