# Exercises: Data Types and Variables

Problems for exercises for the ["Technology Fundamentals " course @ SoftUni](https://softuni.bg/modules/57/tech-module-4-0).   
Submit your solutions in the SoftUni judge system at: [Data-Types-and-Variables-Exercise](https://judge.softuni.bg/Contests/1229/Data-Types-and-Variables-Exercise)

## Sum Digits

You 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 |

## Chars to String

Write a function that 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 |

## 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). Use the correct data types and 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. |

## Convert Meters to Kilometres

You will be given a **number** that will be 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 |

## Pounds to Dollars

Write a JS function that converts British **pounds** to **dollars** formatted to **3th decimal point**.

1 British Pound = 1.31 Dollars

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 80 | 104.800 |
| 39 | 51.090 |

## 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 |

## Lower or Upper

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

1. **\*Gladiator Expenses**

As a gladiator, Pesho has to repair his broken equipment when he loses a fight. His equipment consists of helmet, sword, shield and armor. You will receive the Pesho`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 brakes**.

**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:

* First parameter – **lost fights count** – integer in the range **[0, 1000]**.
* Second parameter – **helmet price** - floating point number in range **[0, 1000]**.
* Third parameter – **sword price** - floating point number in range **[0, 1000]**.
* Fourth parameter – **shield price** - floating point number in range **[0, 1000]**.
* Fifth parameter – **armor price** - floating point number in range **[0, 1000]**.

**Output**

* As output you must print Pesho`s total expenses for new equipment: **"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 constant threat of giant sand worms. To make the work as efficient as possible, the Duke has tasked you with the creation of a management software.*

Write a JS 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 spice is expected in a day, abandon the source.

The mining crew **consumes** 26 spice **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 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 spice and at the end of the shift, the workers consume 26, leaving 85. The yield drops by 10 to 101.  **Day 2** we extract 101 spice, 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 2 days. |