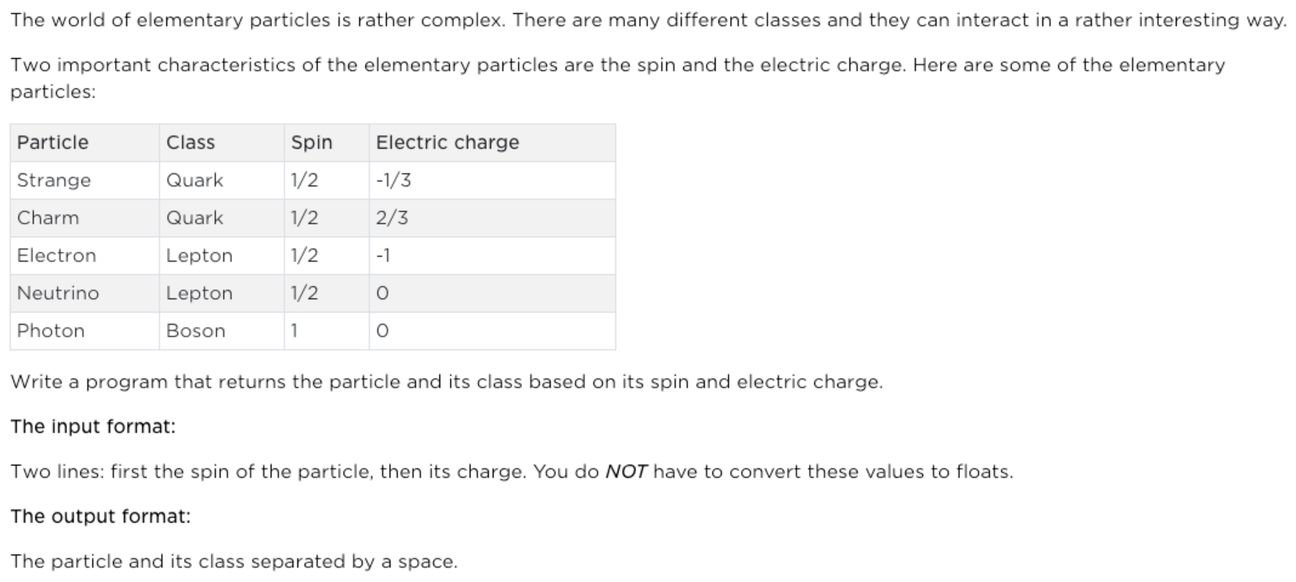
219114/115 Programming I

Exercise 2

All problems should be done in a file named ex2.py. Use Python’s comments to separate the code in each problem

1. Particles



The world of elementary particles is rather complex. There are many different classes and they can interact in a rather interesting way.

Two important characteristics of the elementary particles are the spin and the electric charge. Here are some of the elementary particles:

Particle	Class	Spin	Electric charge
Strange	Quark	1/2	-1/3
Charm	Quark	1/2	2/3
Electron	Lepton	1/2	-1
Neutrino	Lepton	1/2	0
Photon	Boson	1	0

Write a program that returns the particle and its class based on its spin and electric charge.

The input format:

Two lines: first the spin of the particle, then its charge. You do ***NOT*** have to convert these values to floats.

The output format:

The particle and its class separated by a space.

Sample Input 1:

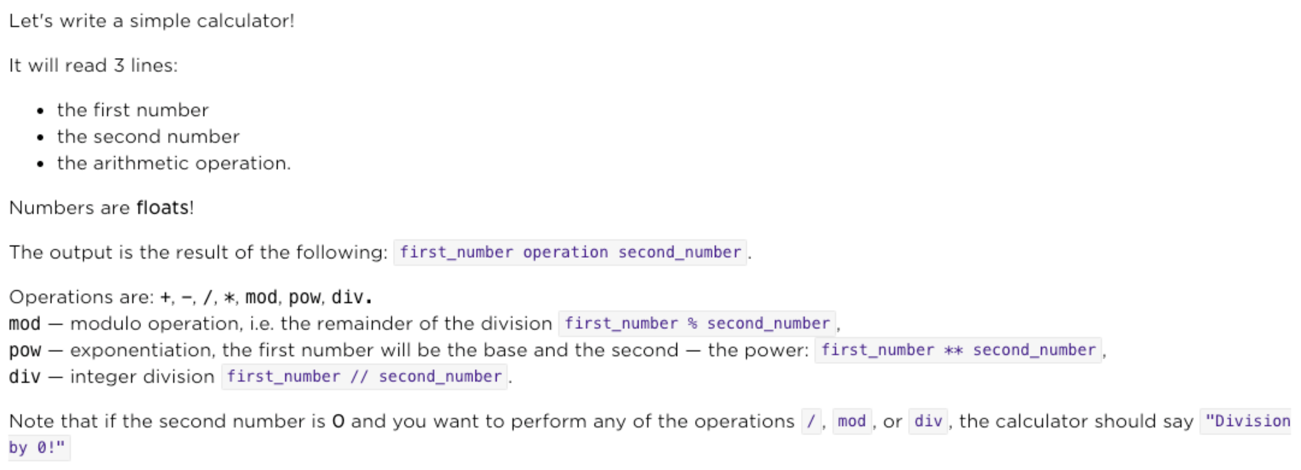
```
Spin: 1/2

Charge: -1/3
```

Sample Output 1:

```
Strange Quark
```

2. Calculator



Let's write a simple calculator!

It will read 3 lines:

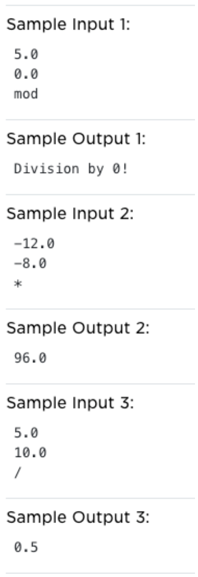
- the first number
- the second number
- the arithmetic operation.

Numbers are **floats**!

The output is the result of the following: `first_number operation second_number`.

Operations are: **+**, **−**, **/**, **∗**, **mod**, **pow**, **div.**
mod — modulo operation, i.e. the remainder of the division `first_number % second_number`,
pow — exponentiation, the first number will be the base and the second — the power: `first_number ** second_number`,
div — integer division `first_number // second_number`.

Note that if the second number is **0** and you want to perform any of the operations `/`, `mod`, or `div`, the calculator should say `"Division by 0!"`



Sample Input 1:

```
5.0
0.0
mod
```

Sample Output 1:

```
Division by 0!
```

Sample Input 2:

```
-12.0
-8.0
*
```

Sample Output 2:

```
96.0
```

Sample Input 3:

```
5.0
10.0
/
```

Sample Output 3:

```
0.5
```

3. Farm

In a farming game, you can buy certain animals for a specific price. As a player, you want to buy the most useful (i.e. the most expensive) animal. Here are the animals and their prices:

Item: Price

Chicken: 23

Goat: 678

Pig: 1296

Cow: 3848

Sheep: 6769

Write a program that determines what is the most expensive animal that the user can buy with their money and how many of them they can buy.

Note that you should only find one kind of animal to buy (the most expensive). You don't need to mention the alternative options.

The input format:

The money that the user has.

The output format:

How many animals the user can afford, for example, 20 chickens. If the user cannot afford any animal, write None.

Pay attention to the number of nouns. Also, keep in mind that the word "sheep" has the irregular plural form "sheep".

Sample Input 1:

```
Your money: 25
```

Sample Output 1:

```
1 chicken
```

Sample Input 2:

```
Your money: 8
```

Sample Output 2:

```
None
```

Sample Input 3:

```
Your money: 668
```

Sample Output 3:

```
29 chickens
```

Sample Input 4:

```
Your money: 3000
```

Sample Output 3:

```
2 pigs
```

4. Day

Today the whole world uses the Coordinated Universal Time (UTC) to distinguish between the time zones. The UTC is considered to be the 0, and the rest of the time zones are expressed using positive or negative offsets from the UTC. For instance, London is in the zone UTC+00:00 (or GMT) and Moscow is in the zone UTC+3:00.

There are 14 positive offsets (from UTC+1:00 to UTC+14:00) and 12 negative offsets (from UTC-12:00 to UTC-1:00). This also means that at a particular hour, three calendar days are observed on the planet. For example, if now it's Sunday, 11:30 in the morning in London, then in the time zone with +14:00 offset people are already living in the "next day", Monday, because their time is 14 hours ahead of London.

Your task is stated as follows:

- the reference time point is Tuesday, 10:30 in the morning in London (UTC+00:00)
- read the input string containing the number and the sign of this number (for example, +4, -10). Note, however, that there will be no sign if the number is 0. The number is always an integer.
- this number is the offset for some time zone.
- your program should calculate the day of the week in the time zone for which you were given the offset. The reference time point for your calculations is mentioned above.

- output the day of the week in the given time zone.

For example, if the input is -11, then, relatively to London, it's "the day before" in this time zone, that is, it's still Monday, but if the input is +3, then it's Tuesday, the same as in London.

The input format:

The value of offset with the sign (e.g. +3 or -9).

The output format:

The day of the week in that timezone.

Sample Input 1:

```
Offset: 0
```

Sample Output 1:

```
Tuesday
```

Sample Input 2:

```
Offset: -11
```

Sample Output 2:

```
Monday
```

For each of the following problems, write three test cases that make a call to the function in question. If the function does return a value, print out the result as well.

5. Write a Python function `is_even` that takes as input the parameter `number` (an integer) and returns `True` if `number` is even and `False` if `number` is odd.

   Hint: Apply the remainder operator to `n` (i.e., `number % 2`) and compare to zero.

6. Write a Python function `is_leap_year` that take as input the parameter `year` and returns `True` if `year` (an integer) is a leap year according to the Gregorian calendar and `False` otherwise. The Wikipedia entry for leap years contains a simple algorithmic rule for determining whether a year is a leap year. Your main task will be to translate this rule into Python.

   "Every year that is exactly divisible by four is a leap year, except for years that are exactly divisible by 100, but these centurial years are leap years if they are exactly divisible by 400. For example, the years 1700, 1800, and 1900 are not leap years, but the years 1600 and 2000 are."

7. Write a Python function `interval_intersect` that takes parameters `a`, `b`, `c`, and `d` and returns `True` if the intervals $[a,b]$ and $[c,d]$ intersect and`False` otherwise. While this test

may seem tricky, the solution is actually very simple and consists of one line of Python code. (You may assume that $a\leq b$and $c\leq d$.)

8. Write a Python function `print_digits` that takes an integer `number` in the range $[0,100)$ and prints the message `"The tens digit is % and the ones digits is %"` where the percents should be replaced with the appropriate values. The function should include an error check for the case when `number` is negative or greater than or equal to 100. In those cases, the function should instead print `"Error: Input is not a two-digit number.”`.

9. Given numbers a, b, and c, the quadratic equation $ax^2+bx+c=0$ can have zero, one or two real solutions (i.e; values for x that satisfy the equation). The *quadratic formula*
$$x=(-b\pm\surd(b^2-4ac))/2a$$
can be used to compute these solutions. The expression b^2-4ac is the *discriminant* associated with the equation. If the discriminant is positive, the equation has two solutions. If the discriminant is zero, the equation has one solution. Finally, if the discriminant is negative, the equation has no solutions.
Write a Python function `smaller_root` that takes an input the numbers `a`, `b` and `c` and returns the smaller solution to this equation if one exists. If the equation has no real solution, print the message `"Error: No real solutions"` and simply return. Note that, in this case, the function will actually return the special Python value `None`.

10. Write a Python function `there_is_odd` that takes 3 integers `x, y, and z` and prints the message `"There is an odd number whose value is % "` where the percents should be replaced with the appropriate values when there is an odd value among the three integers. The function prints `"There is no odd number"` when there is no odd integer among the three input integers.

11. Write a Python function `list_all_odds` that takes 4 integers `w, x, y, and z` and prints the message `“This value is odd % "` where the percents should be replaced with the appropriate values when an odd value is found among the four integers. The number of times this message gets printed out equals the number of odd values that exist among the four input integers. The function prints `"There is no odd number"` when there is no odd integer among the four input integers.

12. Write a Python function `max_of_three` that takes 3 integers `x, y, and z` and prints the message `“The max value is % "` where the percents should be replaced with the maximum value among the three input integers.

Submission:

- ***Create StudentID_Firstname_ex2 folder, where StudentID is your KU ID and Firstname is your given name***
- ***Put the files to submit, ex2.py, into this folder***
- ***Zip the folder and submit the zip file to the course's Google Classroom before the due date***

Grading:

1. Correctness (50%); your code must run and produce correct outcomes; code that does not run because of, for example, syntax errors or name misspelling receives zero credit.

2. Cleanliness (50%): your code must be clean, following PEP 8 style guide; variable names must be meaningful, following PEP 8 convention; comments must be put in for others to be able to read and understand your code, again following PEP 8 convention. For this exercise, which involves relatively simple and short programs, extensive comments may not be necessary.