## Arithmetics practice

Implement a script, which will produce simple arithmetic exercises for the user. Have it configurable by the range of values to appear in the exercises and operations practiced (+, -, \*, /, …).

You will need argparse for configuration and random for exercise generation.

You will practice basic control structures and simple user interaction.

Bonus: implement some basic statistics collection and provide the user with more examples in tasks they have troubles with.

sasha: “if you are done, you can share the [link with me by pasting it here](https://docs.google.com/document/d/1GVWY_NQyb5L_YHGjGF0V-XpE8fswrdq3x30dfFPMoHA/edit?usp=sharing)”

## Numerical derivatives

Implement a function, which takes a function func and a given point x’, and:

1. Plot func on some range around x’
2. Highlight the point (x, func(x’))
3. Compute numerical derivative at x’ (hint: derivative is defined as lim\_{h->0} [f(x+h) - f(x)] / h)
4. Plot the corresponding tangent line

You will need matplotlib for plotting. Optionally use numpy for computation, but refrain from advanced, differentiation-related functions.

You will practice plotting, computation and passing functions as arguments.

Bonus: Try different schemes (left, right, central); show error as a function of h for some known func.

([solution](https://colab.research.google.com/drive/1J2qK4HEC-XX8iXjCvw44bbzy_HFUwzw9?usp=sharing))

## Values with units

Design and demonstrate a system where every value is associated with its unit (e.g. 3 kilogram, 6.28 meters, 9.81 meters / second^2, etc.). Allow sensible computation on the units (i.e. same units can be added, different units can be multiplied).

You probably don’t need any specific tooling. Avoid libraries which already implement this ;-)

You will practice abstract thinking, operations with custom classes, operator overloading.

Bonus: Add an interface for dealing with a (single) nonmetric system (e.g.: distance in feet, weight in pounds); add an interface for estimating dimensions for pretty printing (e.g.: for 9.81 meters / second^2, you’ll be able to print Acceleration: 9.81 m/s^2, where Acceleration is inferred programmatically.)

## Simple clustering example

See the notebook diarization.ipynb.