Material Summary: High-school Maths

1. Math in Real Life

1.1 Mathematics in Nature

- Honeycomb cells
  - Bees produce wax by consuming some of the honey they've made
  - Wax production takes time and energy (honey)
  - The hexagonal cells leave no unused space, and consume the least amount of wax and energy
- Snowflakes
  - All snowflakes are unique but they are perfectly symmetrical
  - Each arm (unless damaged) is identical
  - This makes them strong enough to stay together
- Romanesco broccoli
  - Each little floret looks exactly like the whole plant
  - This is called a fractal
  - Seen from above, the florets form a spiral
- Fibonacci spiral
  - Fibonacci spirals everywhere
  - Flowers, pinecones
  - Animal shells
  - Hurricanes
  - Galaxies

1.2 Mathematics in Music

- Sound is a combination of waves travelling through the air
  - Each sound wave has a frequency (pitch)
  - Every note is associated with a certain frequency, for example A4 produces 440 oscillations every second (440 Hz)
- Some combinations of tones sound pleasant, others sound harsh
  - Our ears like simple frequency ratios, e.g. $2:3$ is better than $160:231$
  - All "good sounding" combinations of tones have simple ratios
- Example: "A major" chord
  - A4: $440\ Hz$, C#5: $554{,}37\ Hz$, E5: $659{,}25\ Hz$

2. Methods

2.1 Divide and Conquer

- Useful for any kind of problem:
  - Especially in algorithms and debugging
  - When invading countries
- Assumption: complicated things are a combination of many, very simple things
  - Algorithms: Merge sort, Discrete Fourier transform
  - Software architecture:

            - "I want to build an ecommerce system"
              ⇒ I want shop owners to add new products
              ⇒ I want to store products in the DB ⇒ …
              ⇒ def save_product(name, price)
    - Debugging
        - The bug is somewhere in my code ⇒ …
          ⇒ the bug is ">=" instead of ">" on line 45 in user.py

2.2 The Scientific Method Steps

- Ask a question
- Do some research
- Form a hypothesis
- Test the hypothesis with an experiment
    - Experiment works ⇒ Analyze the data
    - Experiment doesn't work ⇒ Fix experiment
- Results align with hypothesis ⇒ OK
- Results don't align with hypothesis ⇒ new question, new hypothesis
- Communicate the results

2.3 Why use the Scientific Method Steps

- Useful when we're exploring something new
    - A new algorithm
    - A new codebase we've just been hired to work on
- Based on common logic
- Experiments
- Example: performance testing
    - Research: My logs show that this Web page on my server takes too much time to load
    - Hypothesis: This piece of code is too slow. I need to improve it
    - Control: Measure the runtime (in seconds)
    - Experiment: Try to fix the problem and repeat the runtime test
        - Did the fix bring a considerable performance gain?
    - Communication: Show the results and implement the fix

3. Set Up Our Environment

3.1 Anaconda

- You can install the Python interpreter
  and all libraries manually
    - Hard, boring and repetitive work
    - Error-prone
- Easy solution: platforms like Anaconda
    - Everything you need to get started with Python for science:
      Python interpreter, packages (720+), package manager, IDE
- Download from the Anaconda website
- Current version (March 2023): Anaconda 2022.10 (Py3.9)
    - Choose your platform (Windows, Linux, or MacOS)

SoftUni

Follow us:

    - Follow the installer

3.2 Setting Up an IDE

- You can use the built-in IDE called Spyder
    - You can even use Notepad if that's your thing
- If you want to use another IDE,
  you need to configure it to work with Python
    - Syntax highlighting, autocomplete, etc.
- Visual Studio Code
    - My preferred editor / IDE
    - [Python in VSCode – tutorial]()
    - [Python extension]()
- Visual Studio
    - [Python Tools]()

3.3 Python Online

- There are places where you can execute your code online
    - If you don't have access to Anaconda
    - Or you want to test something very quickly
- https://www.python.org/shell/
    - Provides a Python shell (CLI)
- https://www.pythonanywhere.com/try-ipython/
    - Provides an implementation of IPython (Interactive Python)
        - REPL (Read – Execute – Print Loop)
    - No major difference to the Python shell
- To share your code, you can use
    - http://ideone.com
    - https://gist.github.com/
    - http://pastebin.com/

3.4 Jupyter Notebook

- A very nice and clean way to document your research
- Included in Anaconda
- Can create documents that contain live code, equations, visualizations and explanatory text
    - HTML / CSS / JavaScript
    - Markdown
    -
    - Python
- Start:
    - use the Anaconda shortcut
    - type into the Command Prompt: **`jupyter notebook`**

3.5 How to use Jupyter Notebook?

- Create a new notebook
    - New -> Python 3
- Every piece of text or code is in a cell
    - Text cells just contain text or Markdown

    - Code cells contain code (obviously)
    - Code can be executed
    - Jupyter "remembers" the code
- Execute cell: **Ctrl + Enter**
    - Or use the menus

4. Math Notation

- The basic symbols we use are numbers and letters
    - Usually English or Greek letters
- Special symbols: $=, \geq, \in, \to, \nabla, \infty, \int$
- Indices: $\sum_{n=0}^{10}, \lim_{x \to 0}$

4.1 Other Useful Notations

- Scientific notation
    - Used for very large or very small numbers
    - Numbers are expressed as decimals with **exactly one** digit before the decimal point
    - All other digits are expressed as a power of 10
    - $15\ 000 = 1{,}5.10^4$
    - $0{,}000015 = 1{,}5.10^{-5}$
- Summation notation ("sigma" notation)
    - Used as a shorthand for writing long sums of numbers / symbols
        - Very similar to a for-loop
        - Greek capital "sigma" denotes the sum, the two numbers below and above it denote the start and end points

$$\sum_{i=1}^{5} i = 1 + 2 + 3 + 4 + 5 \qquad \sum_{k=1}^{n} x_k = x_1 + x_2 + \cdots + x_n$$

4.2 Equality Sign

- Important as it has different meanings
    - Like programming: **"=", "==" and "==="**
- Identity
    - The two statements around "=" are always equal: $x(x+3) = x^2 + 3x$
        - We can also use the "identity" symbol: $(a+b)^2 \equiv a^2 + 2ab + b^2$
    - … for all "valid" symbols: $\frac{4x^2}{x} = 4x,\ x \neq 0$
- Equation
    - The two statements are true only for specific values of the symbols

$$2x + 5 = 4,\ x = -0.5 \qquad x^2 - 1 = 0,\ x = \pm 1 \qquad \frac{\mathrm{d}x}{\mathrm{d}t} = 5x - 3$$

- Definition (we can also use $:=$ or $\stackrel{\text{def}}{=}$, or even $\equiv$)

$$\sum i := \sum_{i=1}^{n} i := 1 + 2 + 3 + \cdots + n$$

5. Linear Equations

- Equations of a variable x
- x is "on its own"
  - Not inside a function
  - No powers
- General form: $ax + b = 0$
  - a and b: fixed numbers (parameters)
- Examples:
  - $2x + 3 = 0$
  - $2(2x + 3) - 3x - 3(-4 + 3x) = 12$

- Solutions of the parametric equation
  - $a = 0, b = 0 \Rightarrow 0.x = 0, \ \forall x$ (every x is a solution)
  - $a = 0, b \neq 0 \Rightarrow 0.x = -b$ (no solution)
  - $a \neq 0, \Rightarrow x = -b/a$ (one solution, regardless of b)
- Many simultaneous equations
  - To solve the system, we need to find values of the variable(s) which satisfy **all equations** at once
  - Even if all individual equations have solutions, the system may have no solution
- Solution:
  - Method 1: Solve one equation and substitute
  - Method 2: Use sum of equations
  - Later, we'll learn a faster way of solving these systems
- Example:

$$\left| \begin{array}{rcl} 4x + 3y & = & 7 \\ 3x + 5y & = & 8 \\ x - 2y & = & -1 \end{array} \right.$$

$$\left| \begin{array}{rcl} 4x + 3y & = & 7 \\ 3x + 5y & = & 8 \\ x - 2y & = & -1 \end{array} \right.$$

$$(\mathbf{3}) : x = -1 + 2y$$

$$(\mathbf{3}) \to (\mathbf{2}) : 3(-1 + 2y) + 5y = 8$$

$$-3 + 6y + 5y = 8$$

$$11y = 11$$

$$\boxed{y = 1}$$

$$(\mathbf{2}) \rightarrow (\mathbf{3}) : x = -1 + 2.1$$

$$\boxed{x = 1}$$

$$(\mathbf{1}) : 4.1 + 3.1 = 7$$

$$\Rightarrow (x, y) = (1, 1) \text{ is the only solution of the system}$$