High-School Maths

Establish a workflow, get to know our tools, review basic concepts

Yordan Darakchiev

Technical Trainer

Software University

SoftUni

Software University

https://softuni.bg

Have a Question?

Software University

sli.do

#MathForDevs

Table of Contents

Mathematics in Real Life

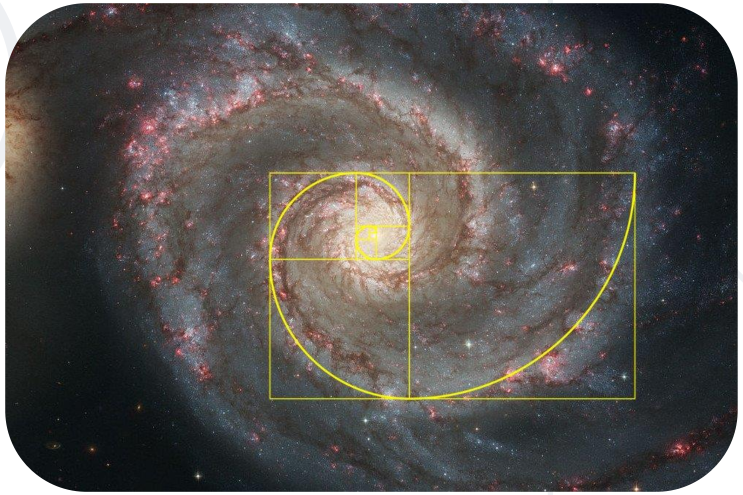
Mathematics in Nature

- **Honeycomb cells**
  - 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**
  - This makes them strong enough to stay together

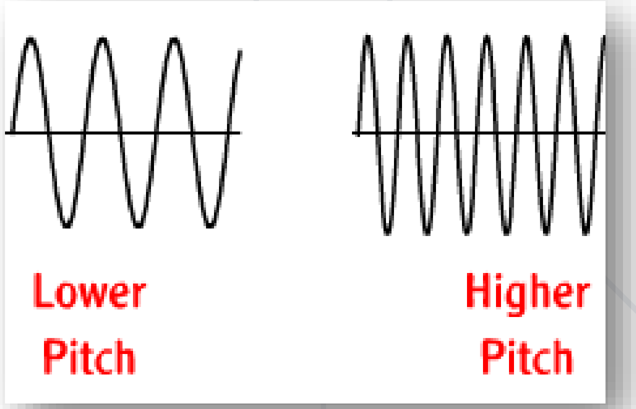
Mathematics in Nature

- **Romanesco broccoli**
  - Each little floret looks exactly like the **whole plant**
  - Seen from above, the florets form a **spiral**
- **Fibonacci spirals everywhere**
  - Flowers, pinecones
  - Animal shells
  - Hurricanes
  - Galaxies



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**
    - **A4 produces 440 oscillations every second ($440\ Hz$)**
  - Some combinations of tones sound **pleasant**, others sound **harsh**
  - **Example**: "A major" chord
    - A4: $440\ Hz$, C#5: $554{,}37\ Hz$, E5: $659{,}25\ Hz$
    - $A4: C\#5: E5 \approx 4:5:6$
    - $A4: E5 \approx 2:3$



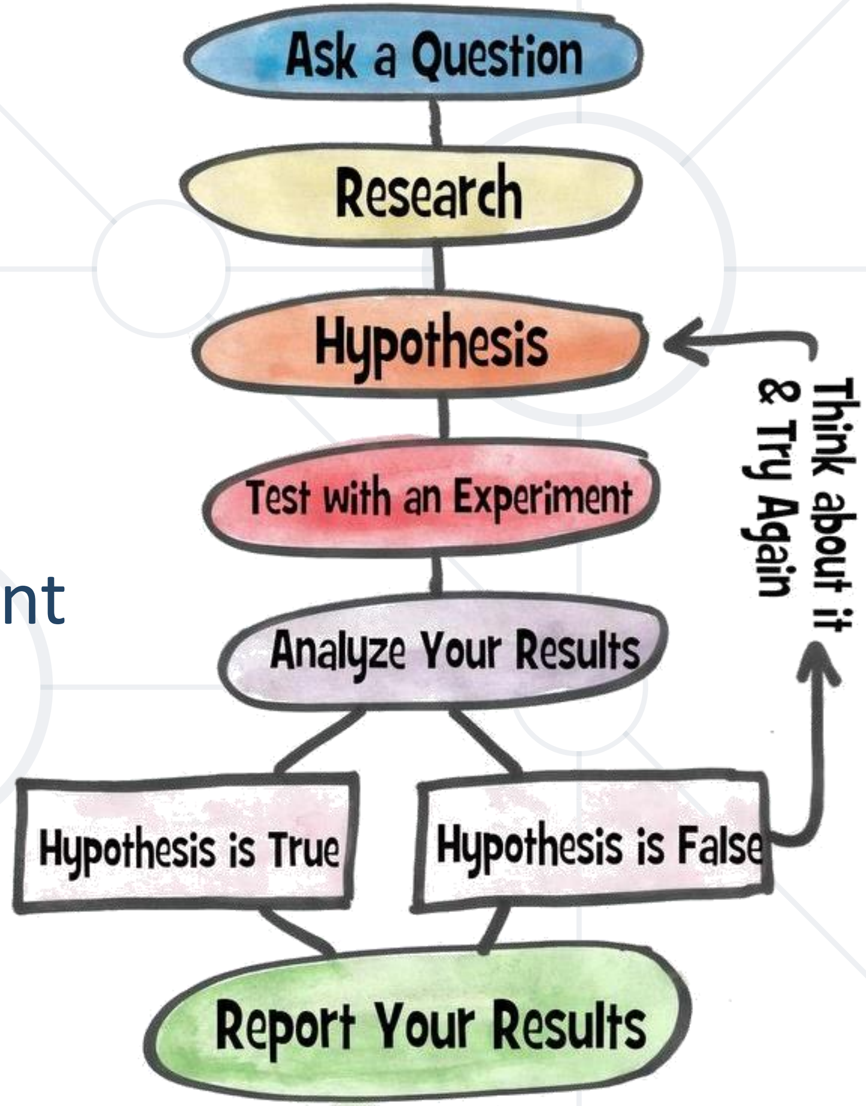
Methods

Divide and Conquer

- Useful for any **kind of problem**
- **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`

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**



Why Use the Scientific Method?

- Useful when we're **exploring** something new
- Based on common logic
- Experiments
- **Examples**
  - **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
  - **Communication:** Show the results and implement the fix

Setting Up Our Environment

Getting ready to conquer math, science and programming

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
    - Jupyter lab
- Download from [the Anaconda website]

Setting Up an IDE (Optional)

- You can use the built-in IDE called **Spyder**
- If you want to use another IDE,
  you need to configure it to work with Python
- **Visual Studio Code**
  - Preferred editor / IDE
  - Python in VSCode – tutorial
  - Python extension
  - Jupyter extension
- **Visual Studio**
  - Python Tools

Python Online

- There are places where you can execute your code online:
  - https://www.python.org/shell/
  - https://www.pythonanywhere.com/try-ipython/
  - https://www.kaggle.com/code
- To share your code, you can use:
  - https://gist.github.com/
  - http://pastebin.com/

Jupyter Lab

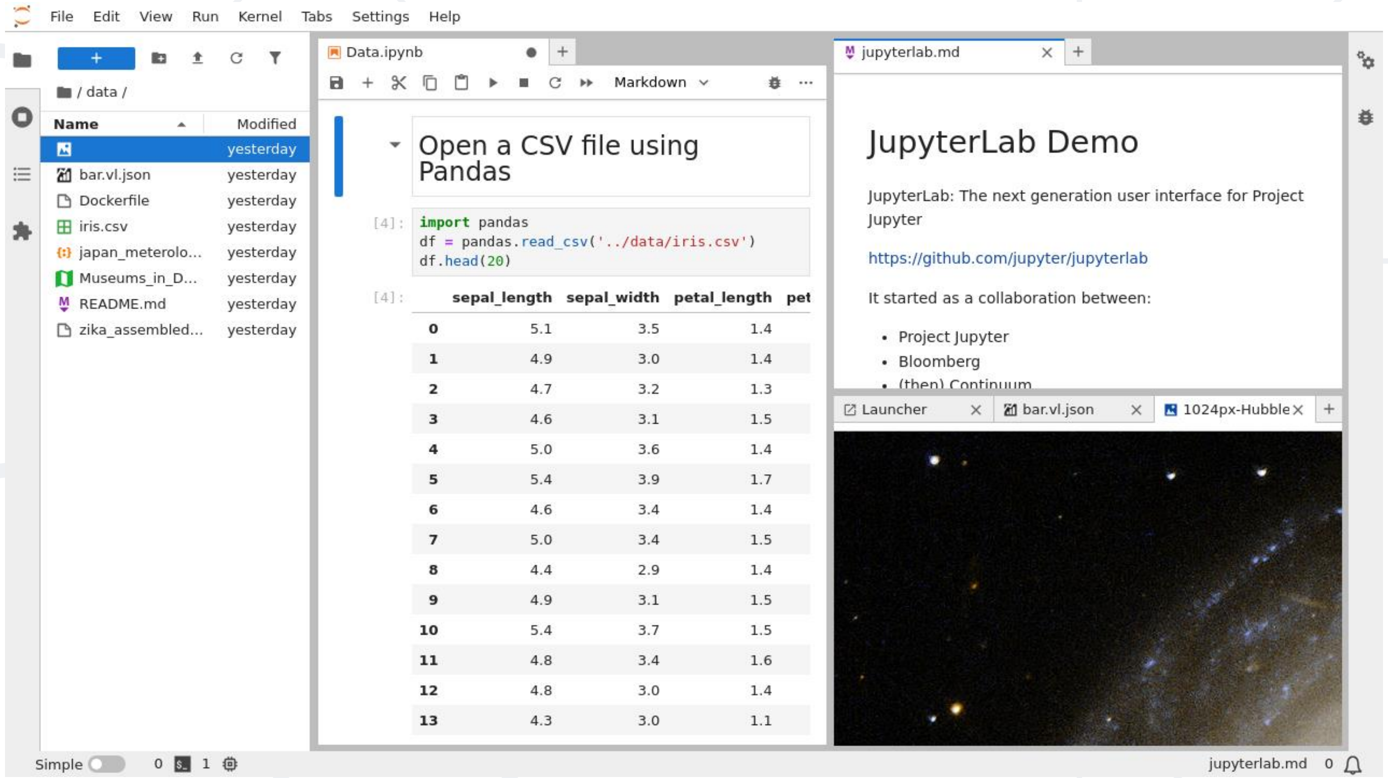
Software University

- 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
  - LaTeX
  - Python
- Start
  - use the **Anaconda shortcut**
  - type into the **Command Prompt**

```
jupyter lab
```

How to Use Jupyter Lab?

- Create a **new notebook**
- Every piece of text or code exists within a **cell**
  - Text cells
  - Code cells
  - You can run (execute) code cells
  - Jupyter "remembers" the code that you already ran
- Execute cell: **`Ctrl + Enter`**

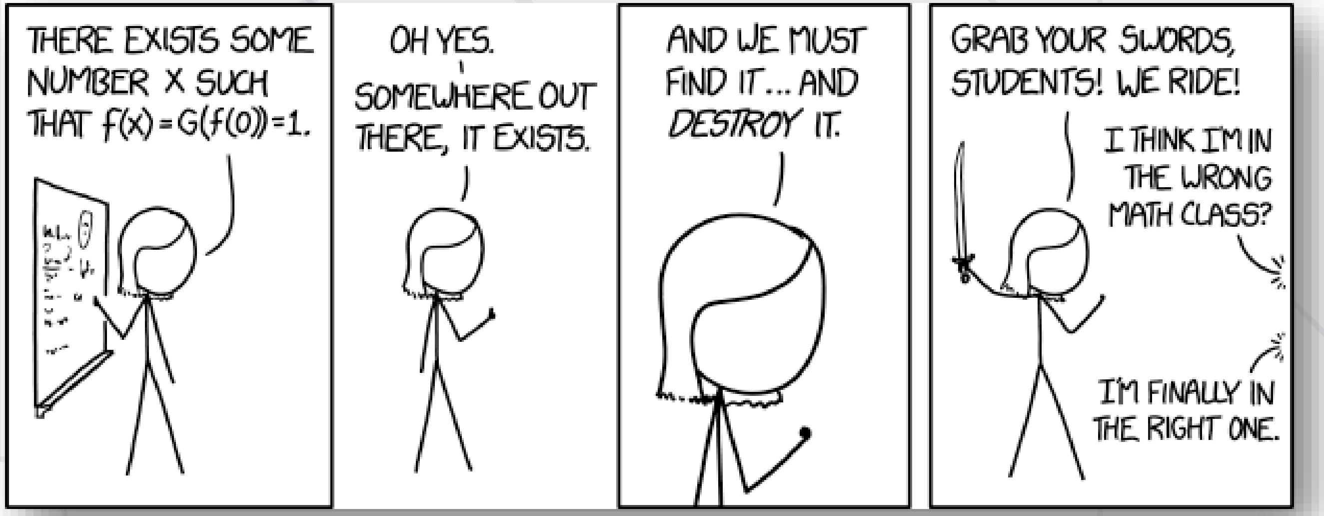


Math Notation

How to write more quickly and concisely

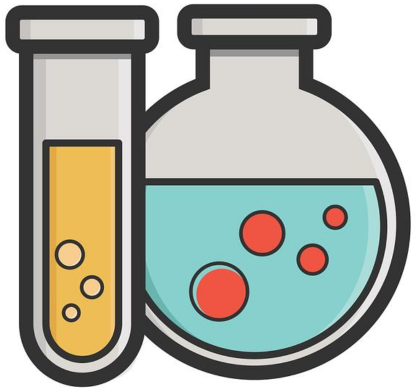
Math Notation

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



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**
- $\mathbf{15\,000 = 1,5.10^{4}}$
- $\mathbf{0,000015 = 1,5.10^{-5}}$



Summation Notation

- "Sigma" notation
- Used as a **shorthand** for writing long sums of numbers or 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

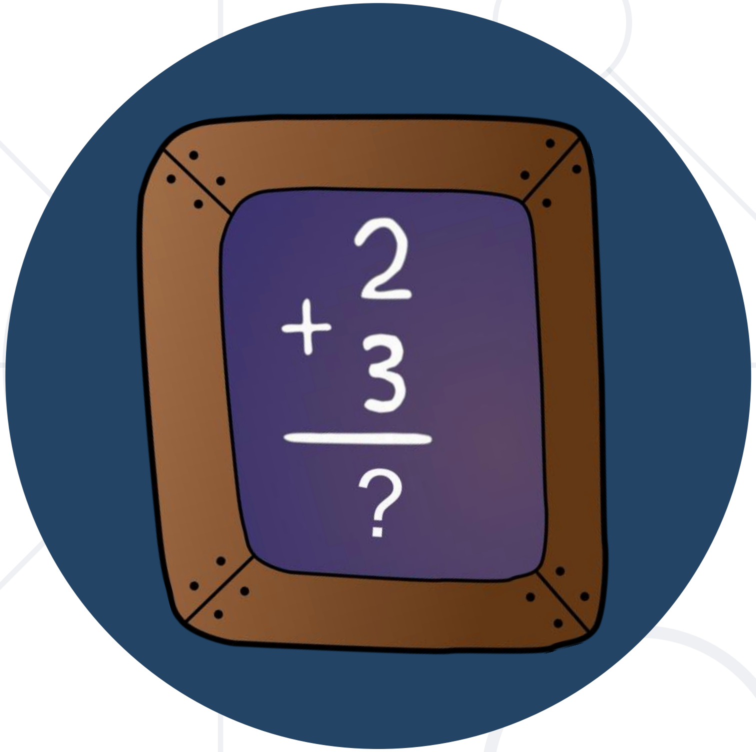
$$\sum_{i=1}^{5} i = 1 + 2 + 3 + 4 + 5 \qquad \sum_{k=1}^{n} x_k = x_1 + x_2 + \cdots + x_n$$

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: $(a+b)^2 \equiv a^2 + 2ab + b^2$
- **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** $\sum i := \sum_{i=1}^{n} i := 1 + 2 + 3 + \cdots + n$



Linear Equations

Simple, yet very useful

Review

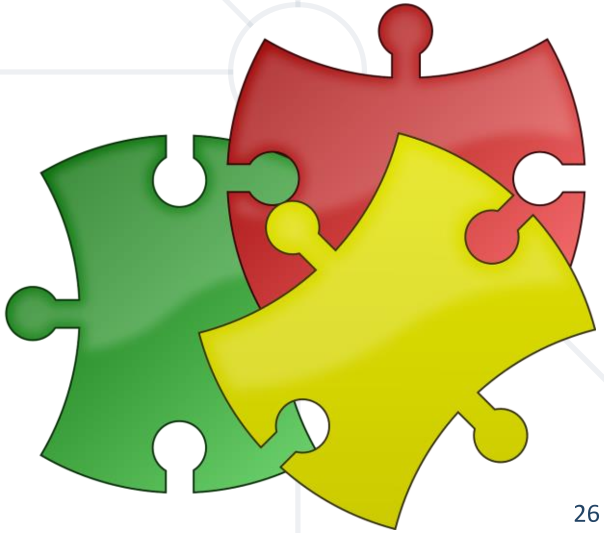
- Equations of a **variable** x
- x is "on its own"
  - Not inside a function
  - No powers
- General form: $ax + b = 0$
  - a and b: fixed numbers (**parameters**)

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)

Exercise: Linear Equations

- Write a Python function which solves a linear equation given the definition from the previous slide
- The function should accept the **a** and **b** as arguments
- The function should return
  - The solution, if there is only one
  - Empty list **[ ]** if no solution or all x satisfy the equation

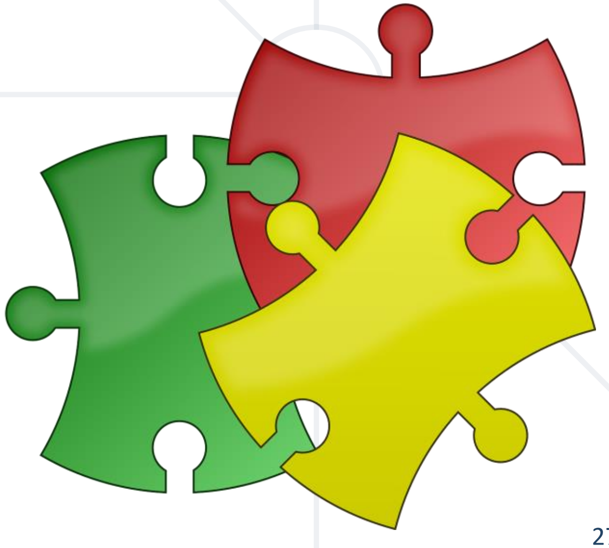


Exercise: Linear Equations

```python
import math

def solve_linear_equation(a, b):
  if a == 0:
      return []
  else:
    return -b / a
```

```python
# Test cases
solve_linear_equation(0, 0) # []
solve_linear_equation(0, 5) # []
solve_linear_equation(5, 0) # 0.0
solve_linear_equation(5, 5) # -1.0
solve_linear_equation(2.5, -5.3) # 2.12
```



Linear Systems of Equations

- 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

Example

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

$$\begin{aligned}
&(\mathbf{3}) : x = -1 + 2y \\
&(\mathbf{3}) \to (\mathbf{2}) : 3(-1+2y) + 5y = 8 \\
&\qquad -3 + 6y + 5y = 8 \\
&\qquad 11y = 11 \\
&\qquad \boxed{y = 1} \\
&(\mathbf{2}) \to (\mathbf{3}) : x = -1 + 2.1 \\
&\qquad \boxed{x = 1} \\
&(\mathbf{1}) : 4.1 + 3.1 = 7 \\
&\Rightarrow (x, y) = (1, 1) \text{ is the only solution of the system}
\end{aligned}$$

Note: The numbers of equations and variables matter!

Summary

- Maths in real life
  - "Pause and ponder"
- The scientific method as a "guiding light"
- Tooling
- Math notation
- Linear equations
  - Does an equation always have a solution?
  - How about infinitely many solutions?
- Systems of linear equations
  - Substitution method

Software University

Questions?

SoftUni

- Software University
- SoftUni Creative
- SoftUni Digital
- SoftUni Foundation
- SoftUni Kids
- СофтУни БУДИТЕЛ
- Finance Academy

SoftUni Diamond Partners

Software University

SmartIT

createX

VIVACOM

INDEAVR
Serving the high achievers

Coca-Cola HBC Bulgaria

Postbank

top EMPLOYER
България Bulgaria 2024
FOR A BETTER WORLD OF WORK

AMBITIONED

SUPER HOSTING .BG

DRAFT KINGS
THE CROWN IS YOURS

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

Software
University

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg

Software University