NICOLAUS COPERNICUS UNIVERSITY IN TORUŃ

RESEARCH UNIVERSITY
EXCELLENCE INITIATIVE

# Python3 7404-FIZ-KKP – Introduction

**Iulia Emilia Brumboiu**

iubr@umk.pl

20 October 2025

# Today

- **About the course**
  - How to register
  - Course overview: objectives, schedule, and topics covered
  - Course website
  - Project and grading
- **Course questionnaire**
- **Code prototyping with Jupyter notebooks**
  - Working with conda environments
  - Jupyter notebook basics

# Part 1: Course Overview

# How to register

Send an e-mail to: **ast@umk.pl** and ask to be registered for Python3 7404-FIZ-KKP

# Course overview

Send an e-mail to: **ast@umk.pl** and ask to be registered for Python3 7404-FIZ-KKP.

- **Course objective**: introduce the tools necessary to design, develop and maintain a Python-based project.
- **ECTS**: 3.
- **Workload**: 30h class + ~30h homework/project work.
- **Classes**: usually Mondays 11:00 – 13:00, but a few extra classes on Thursdays 08:00 – 10:30.
- **Class start**: **11:15** (Mondays, 15 min. break @ 12:00), **08:30** (Thursdays, & we skip the break).

# Course overview

Send an e-mail to: **ast@umk.pl** and ask to be registered for Python3 7404-FIZ-KKP.

- Short lectures combined with **practical project-based tasks**.
- You are welcome to **bring and use your own laptop**, or use the computers in B.2.22.
- If you use the lab computers, **be careful where you save your data**!
  - Partitions are cleaned regularly, so
  - Follow the instructions in the pop-up that shows up when you start Windows.

# Course overview

Send an e-mail to: **ast@umk.pl** and ask to be registered for Python3 7404-FIZ-KKP.

- **Course website**: https://sites.google.com/view/python3-umk
- You'll find the **schedule** and **topics**, as well as course material.

# Grading

30% active class participation

- attend classes,
- complete project tasks,
- complete homeworks.

70% project work: develop your own Python software package.

**Grading system:**

0% - 49% - grade: 2

50% - 60% - grade: 3
61% - 70% - grade: 3+
71% - 80% - grade: 4
81% - 90% - grade: 4+
91% - 100% - grade: 5

# Project: Develop your own Python software

You can either choose to work on:

1. A Python project which you **design yourself,**
2. One of the Python projects **suggested** by me.

If you would like to work on **your own idea**, try to define a:

- Self-contained project,
- Useful for your (PhD) work,
- Not too complicated.

# Project: Develop your own Python software

**Project evaluation**:

1. Github or Gitlab **repository**:
    - Repository should contain the following components:
        - README with installation instructions,
        - Proper project structure,
        - Python tests and script/notebook examples,
        - Software manual,
        - One C/C++ class which is exposed to Python.
    - I should be able to install and run your software,
    - The code should follow Python coding conventions,
    - The Git commits should follow (more or less) a set of commit conventions.

2. Short **project presentation**:
    - 2026/02/02
    - 10 min. Jupyter notebook demonstration of your software

# Project: Develop your own Python software

**Project evaluation**:

1. Github or Gitlab **repository**:
    - Repository should contain the following components:
        - README with installation instructions,
        - Proper project structure,
        - Python tests and script/notebook examples,
        - Software manual,
        - One C/C++ class which is exposed to Python.
    - I should be able to install and run your software,
    - The code should follow Python coding conventions,
    - The Git commits should follow (more or less) a set of commit conventions.

- We will cover each of these topics during the semester,
- Most of the practical tasks during classes will be directly related to your project.

# Project: Develop your own Python software

**By/on 13 November**:

- Decide which project you would like to do:
  - Select from my list of projects (list to be posted on course website),
  - Design your own.
- During the **class on 13 November**, we'll:
  - Set the projects,
  - Discuss the corresponding code structure.

# Part 2: Course Questionnaire

# Prior experience with Python

Complete the questionnaire:

http://bit.ly/3KVWp6s

# Part 3: Conda and Jupyter Notebooks

# Virtual Environment

**Isolated** environment which allows to manage **project-specific dependencies** and **prevents version conflicts** between projects.

**Python Virtual Environment**

- Isolates Python packages
- Available with standard Python

Recommended for **pure Python** projects.

**Conda Environment:**

- Isolates both Python and other languages (C, C++, Fortran, etc.)
- Needs a conda interpreter like Miniconda or Anaconda

Recommended for **multi-language** environments.

# Virtual Environment

**Isolated** environment which allows to manage **project-specific dependencies** and **prevents version conflicts** between projects.

**Python Virtual Environment**

- Isolates Python packages
- Available with standard Python

Recommended for **pure Python** projects.

**Conda Environment:**

- Isolates both Python and other languages (C, C++, Fortran, etc.)
- Needs a conda interpreter like Miniconda or Anaconda

Recommended for **multi-language** environments.

**Recommendation: Always work in a virtual environment!**

# Jupyter Notebook

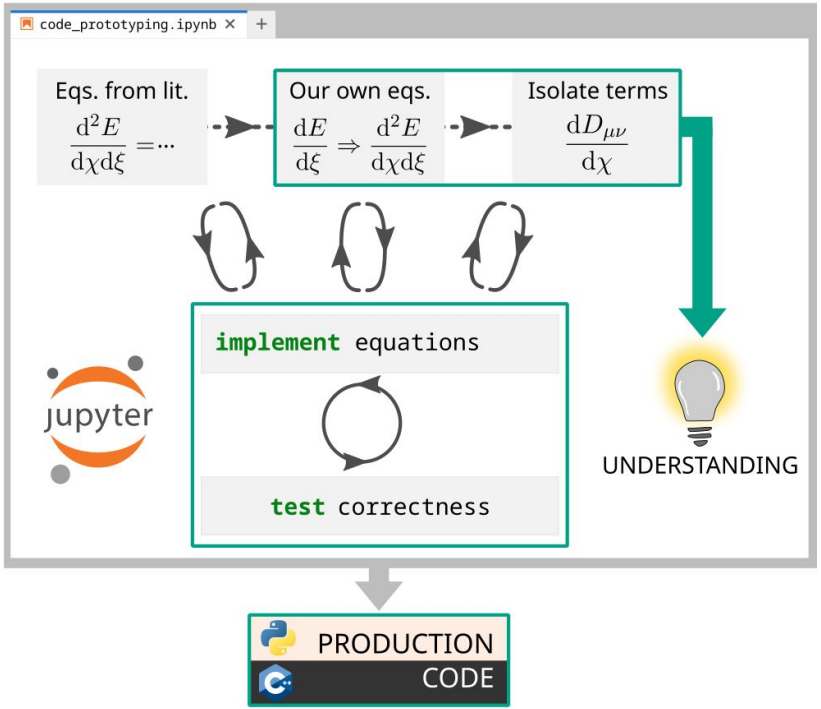
- **Browser**-based **interactive** computational notebook.
- Multiple programming languages (**Ju**lia, **Pyt**hon, **R**, and more, e.g. shell scripting).
- Input cells can be:
  - **Code** cells
  - **Markdown** cells with equations, text, images.
- Allows **side-by-side** comparison of **equations** and **code**.
- Code **runs in real time**, variables can be directly accessed, printed, or plotted.

https://jupyter.org/

# Jupyter Notebook

**Development Strategy:**

- Prototype code using notebooks.
- Debug and test if the code is correct.
- Implement as production code in Python project.




M. Hodecker *et al.*, Chem. Methods 2500033 (2025). https://doi.org/10.1002/cmtd.202500033

# Tasks for today

- Follow the instructions on the course website to install the conda environment defined by the **myenv.yml** file,
- Download the **jupyter.ipynb** notebook from Course Material,
- Run the notebook and **fill in the missing code**,
- When you are finished, save the notebook, rename it with your name, and send it in.

# Summary

- Introduction to using conda and working in a conda environment,
- Introduction to interactive programming with Jupyter notebooks.

**Programming strategy for the rest of the course:**

- Always work in a conda environment, i.e. always run `conda activate my-env` before programming for the project.
- Prototype your routines first (e.g. using notebooks), then implement them in the source code of your project.

# Homework

- If you haven't finished, complete the Jupyter notebook.
- Send me the completed notebook by **Friday 25/10**.

# Next time we will:

- Create the Git repository for our projects,
- Get an introduction to Git version control.

Thank you for today!

Questions?