# PHOT 110: Introduction to programming
# LECTURE 00

*Michaël Barbier, Spring semester (2025-2026)*

# COURSE INFORMATION

**Instructor**

Dr. Michaël Barbier
e-mail: michaelbarbier@iyte.edu.tr
Office – door on the right of Z5
Office Hours – Friday 11:00-13:00
(or via appointment)

**Teaching Assistants**

Volkan Bozkuş
e-mail: volkanbozkus@iyte.edu.tr
Office: Z-9B
Office hours: TBA

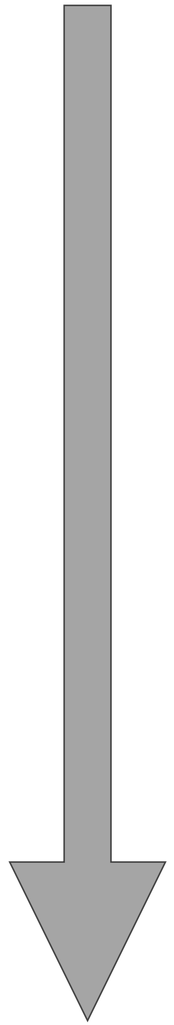
Metin Tan
e-mail: metintan@iyte.edu.tr
Office: Z9B
Office hours: TBA

**Course Schedule**

| | | |
|---|---|---|
| Thursday | 10:45 – 12:30 (Theory) | Ali Küçük Lab (computer lab on the 1st floor) |
| Thursday | 13:30 – 15:15 (Practice) | Ali Küçük Lab |

# CONTENTS OF THE COURSE

- How computers perform calculations
- How to program in Python
- Implementing numerical methods
- Handling input/output data, plotting graphs, tables
- Debugging, testing, and benchmarking your code



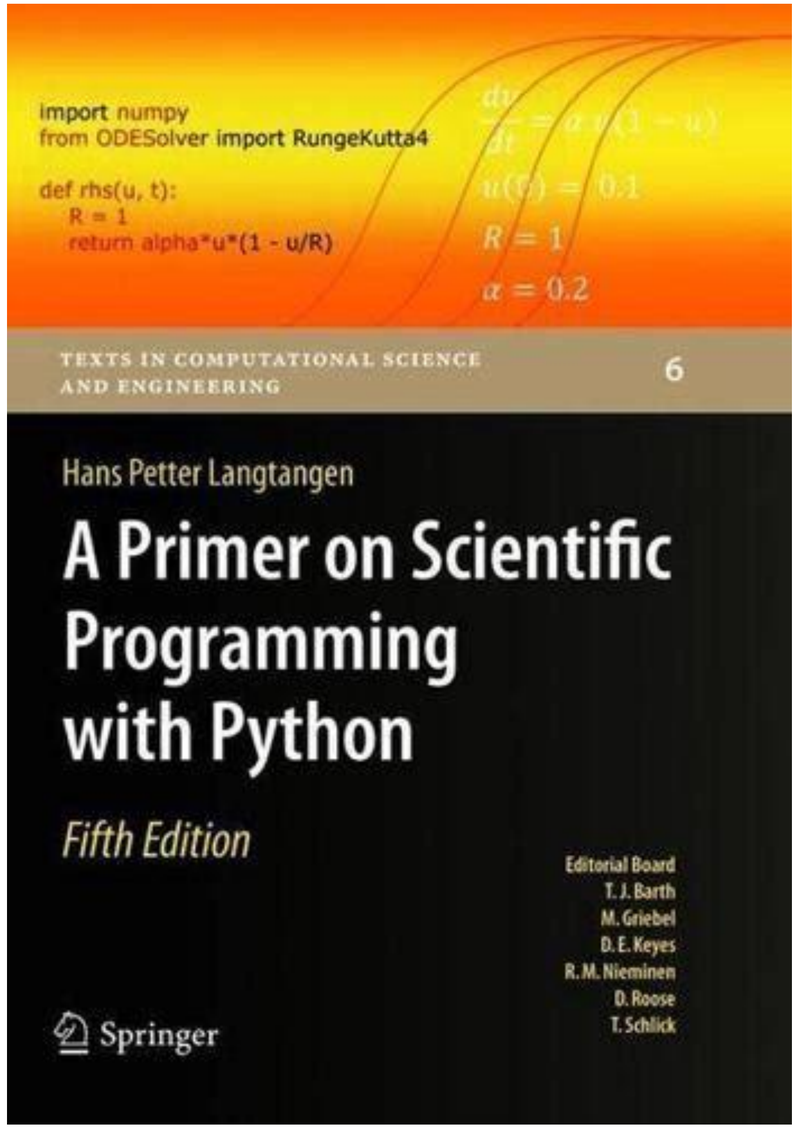
# COURSE MATERIALS

**Course book**

H.P. Langtangen, A primer on scientific programming with Python, Springer

Supporting slides and materials
http://hplgit.github.io/scipro-primer/slides/index.html

https://link.springer.com/book/10.1007/978-3-030-16877-3



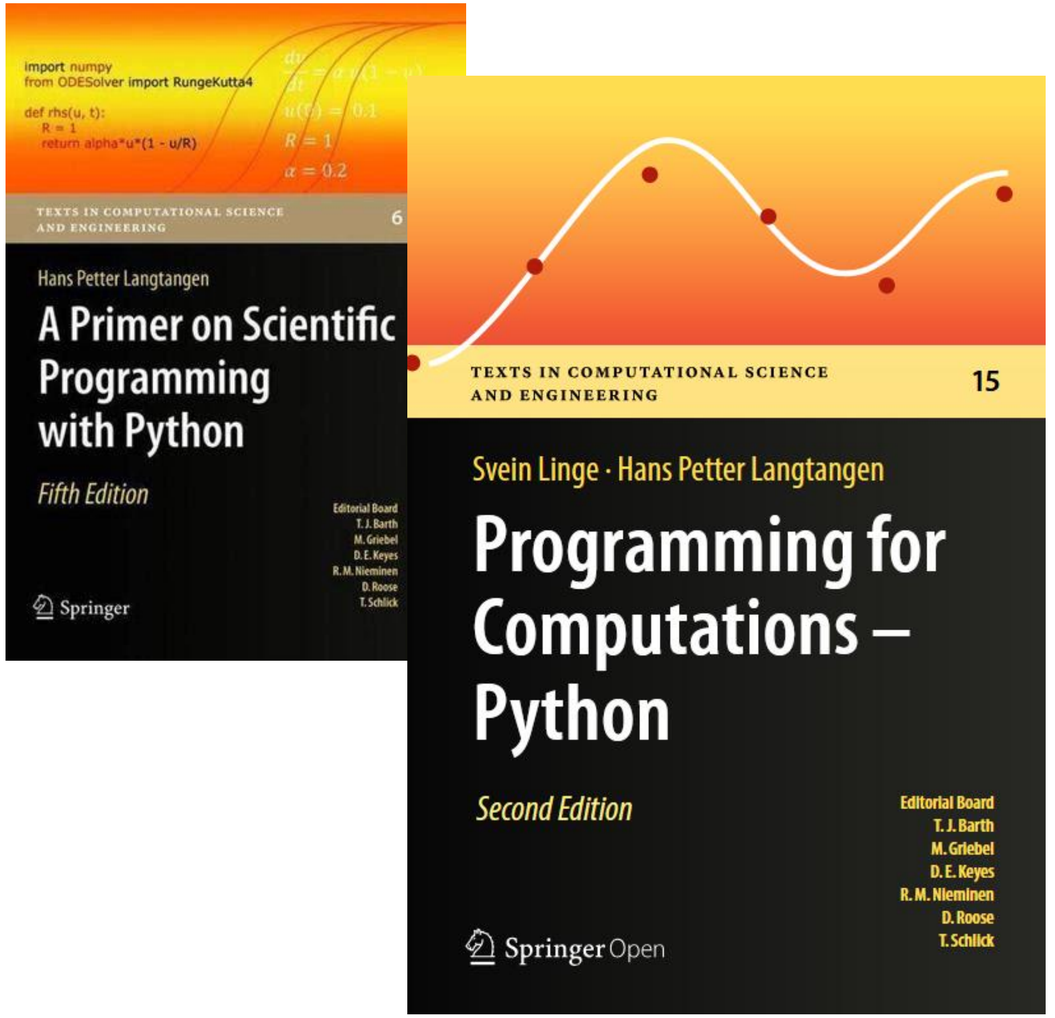
# COURSE MATERIALS

## Course book

H.P. Langtangen, A primer on scientific programming with Python, Springer

Supporting slides and materials
http://hplgit.github.io/scipro-primer/slides/index.html

Free book version with some overlap in content (first 5 chapters)



# OVERVIEW OF THE COURSE

| week | topic | Chapter |
|---|---|---|
| Week 1 | Computers and Python basics: interpreter, IDE, running a script | 1 |
| Week 2 | Expressions, variables, variable types | 1 |
| Week 3 | Program control flow: while-loop, for-loop, lists, and ranges | 2 |
| Week 4 | Program control flow (ctu'd): conditional execution. Further data types: nested lists, Tuples, Arrays | 3 |
| Week 5 | Functions, variable scope | 3 |
| Week 6 | Input/output and error handling | 4 |
| Week 7 | **Midterm exam** (Thursday, April 9) | |
| Week 8 | Modules and script documentation | 4 |
| Week 9 | Vectors, Arrays, implementation of numerical algorithms | 5 |
| Week 10 | Plotting graphs using Matplotlib | 5 |
| Week 11 | Dictionaries, Strings, tables with Pandas | 6 |
| Week 12 | Object Oriented Programming | 7 |
| Week 13 | Object Oriented Programming ctu'd | 9 |
| Week 14 | Unit tests, integrated testing performance, benchmarking, profiling | App. F |
| Week 15-16 | **Finals** | |

# COURSE SYLLABUS AND CLASS WORKFLOW

**Homework/projects**

- One project on an applied topic (20%)
- Working together on solutions allowed
- But .. individual written reports and code

**Exams**

- Midterm exam (10%) & final exam (70%)
- Solve problems on the computer by writing scripts

**Most course materials on webpage:**

https://web.iyte.edu.tr/michaelbarbier/education_phot110.html

# AI-ASSISTED CODING OR “VIBE CODING”

**Vibe coding:**

- Software applications with AI code generators
- Develop complex software applications as amateur
- The idea of VIBE coding is NOT to understand your code
- More information: **https://en.wikipedia.org/wiki/Vibe_coding**

**We will NOT employ vibe coding during our course:**

- Almost no educational value for beginners
- Current AI tools insufficient
- Clear drawback for the software community

**Follow the University rules on AI usage:**

Use it to explore knowledge & gain inspiration

Don’t use to avoid gaining knowledge

# EXAMPLES AI-ASSISTED CODING: PRINT NUMBERS

- AI-assisted coding can give intended result for **very small scripts**
- In this example the AI generated code was exactly the same

**Manual code**

```
for i in range(1, 5):
    print(i)
```

**AI-assisted code**

Prompt: print the numbers from 1 to 4

```
for i in range(1, 5):
    print(i)
```

# EXAMPLES AI-ASSISTED CODING

## Prompt: Plot a sine curve

## Manual code

```python
"""Plot a sine curve """
import matplotlib.pyplot as plt
from numpy import pi, sin, linspace

# Parameters
A = 1                   # Amplitude
f = 2                   # frequency

# Plot
x = linspace(0, 2, 1000)
y = A * sin(2 * pi * f * x)
fig, ax = plt.subplots()
ax.plot(x, y)
ax.set_xlabel("x")
ax.set_ylabel("y")
plt.show()
```

plot_sine.py > ...

```python
#!/usr/bin/env python3
"""Simple sine-curve plotting script.

Saves `sine.png` by default. Accepts optional CLI args for frequency,
amplitude, phase, duration, and sample rate.
"""
import argparse
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

def plot_sine(freq=1.0, amplitude=1.0, phase=0.0, duration=2.0, fs=1000,
              filename='sine.png', show=False):
    t = np.linspace(0, duration, int(duration * fs), endpoint=False)
    y = amplitude * np.sin(2 * np.pi * freq * t + phase)

    if not show:
        matplotlib.use('Agg')

    plt.figure(figsize=(8, 4))
    plt.plot(t, y, lw=1.5)
    plt.xlabel('Time (s)')
    plt.ylabel('Amplitude')
    plt.title(f'Sine wave — {freq} Hz')
    plt.grid(alpha=0.3)
    plt.tight_layout()
    plt.savefig(filename, dpi=150)
    print(f'Saved plot to {filename}')
    if show:
        plt.show()
    plt.close()


def main():
    p = argparse.ArgumentParser(description='Plot a sine curve')
    p.add_argument('--freq', type=float, default=1.0, help='frequency in Hz')
    p.add_argument('--amplitude', type=float, default=1.0, help='amplitude')
    p.add_argument('--phase', type=float, default=0.0, help='phase in radians')
    p.add_argument('--duration', type=float, default=2.0, help='duration in seconds')
    p.add_argument('--fs', type=float, default=1000.0, help='sample rate (Hz)')
    p.add_argument('--out', default='sine.png', help='output filename')
    p.add_argument('--show', action='store_true', help='display the plot interactively')
    args = p.parse_args()

    plot_sine(freq=args.freq, amplitude=args.amplitude, phase=args.phase,
              duration=args.duration, fs=args.fs, filename=args.out,
              show=args.show)


if __name__ == '__main__':
    main()
```