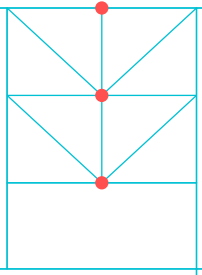
Course Introduction

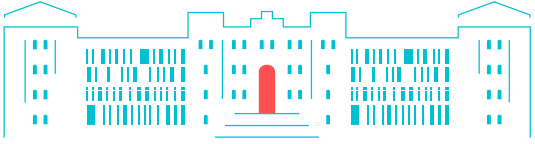
Programmierparadigmen – Objektorientierte Programmierung (OOP)

Week 1/14 – version 26.0

TUHH
Hamburg
University of
Technology

Institute for Autonomous Cyber-Physical Systems





Thibaut Lunet

What is this course ?

Module for second semester bachelor students

- Mandatory for CS, DS, IIW, TM / Vertiefung for AI, ES
- Available for any voluntary ...

Main goals

Continuation of "Prozedurale Programmierung für Informatiker" (C programming)

- C++ (∼2/3 of the content) and Python (∼1/3 of the content)
- learning **Object-Oriented Programming** (and a bit about other Programming Paradigms)
- consolidation with Git, compilation and debug of programs, project development, ...

Note :

lectures will be (mostly) in English, but you can also communicate with me in German or French ...

How is it going to work ?

General organization

- 1 lecture (Vorlesung – VL) + 1 tutorial (Hörsaalübung - HÜ) every week
- 1 exercise session (Praktikum – PR) every weeks starting next week, with *bonus points*
- one programming project, starts after the mid-semester holidays
- exam at the end of this semester, *second chance exam end of next winter semester*
  → see webpage for exam dates ...

Official StudIP Course → *you **have to register** for notifications + blubber*

Gitlab Repository (Repo) for **up-to-date material and infos**
→ *access with your StudIP login (**LoginID**, e.g* cpf5546*) and password*

Organisation – Lectures

Lectures

Vorlesung : **Thursday 8:00** → **9:30** in Audimax I (except 23.04.2026 in Audimax II)
Hörsaalübung : **Friday 8:45** → **9:30** in Audimax II

- slides and example codes uploaded on Gitlab
- mostly in English, with a touch of German
- possible zoom link (with good reason) but **no recording**
- exceptional session in zoom only (**if any**, will be specified in advance)
- some weeks won't have lecture (cf. StudIP and TUNE)
- small (free) knowledge test during Hörsaalübung (MCQ) on content from previous lecture

*Slides are on Gitlab (some will be reworked during the semester, **check the version !**)*
→ replace `blob` by `raw` in slide's url to get a **pdf view**

Organisation – Exercises

TUHH

Exercise Sessions

All on Friday : 11 groups **9:45** → **11:15**, 1 group **13:15** → **14:45**

→ ∼ 18 students per groups

- **work in teams** (3 or 4 students), using a shared Gitlab repository (more details later ...)
- instructions in English, support by your tutor (German, English)
- bonus points for the exams (more details later ...) :
  1. **Problem solutions** : solve a few issues with small code snippets (checked by tutors → 2 pts)
  2. **Programming tasks** : implement a program for a given task (automatically checked → 2 pts)
- timeline :
  1. April 10th : Team registration & Support-at-start (tutors will be in their room, see Gitlab)
  2. April 17th : first session (S01)
- **personal laptop needed** (if not possible, come see me at the end ...)

Organisation – Exercise Groups and Teams

Groups and team distribution – see rules on Gitlab ...

- formed during the first week (deadline on the HÜ before the first session)
- required for exercise and project evaluation
- possible adaptations in function of constrains

1 – Form your team (3–4 people) by talking to others

2 – Check which groups are still open on Gitlab and choose one

3 – Communicate all your StudIP LoginIDs and your team name to the group's tutor

Organisation – Exercise Groups and Teams

Groups and team distribution – see rules on Gitlab ...

- formed during the first week (deadline on the HÜ before the first session)
- required for exercise and project evaluation
- possible adaptations in function of constrains

1 – Form your team (3–4 people) by talking to others

2 – Check which groups are still open on Gitlab and choose one

3 – Communicate all your StudIP LoginIDs and your team name to the group's tutor

Note : alone or only two, don't know anyone to form a team ? Use the **StudIP blubber**

1. quickly present yourself, motivations, current level
2. contact other people who also used the blubber

Full instructions on Gitlab

Tutors presentation

Quick round

- which semester and field of study ?
- spoken languages and tutor experience ?
- programming languages and operating system affinity ?
- what did you dislike the most when attending OOP as a student ?

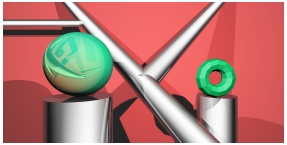
Organisation – Project

Programming Project

Implement a program for 3D visualization using "raytracing"

- combines a bit of math, optical physics and practical use of OOP
- collaborative project within exercise teams, official start after semester holidays
- additional bonus points for the exam

Some examples from last year's projects ...




Full project instructions

Exam and Grading

TUHH

Main written exam – 6 ECTS (Leistungspunkte)

- 90 min. duration, 40 exam points in total :
  → 20 points to pass (4.0 grade), 38 points for best grade (1.0 grade)
- questions in English, answers in English or German, **no tools neither aids**
- first date after Summer Semester, second after Winter Semester

Exam and Grading

TUHH

Main written exam – 6 ECTS (Leistungspunkte)

- 90 min. duration, 40 exam points in total :
  → 20 points to pass (4.0 grade), 38 points for best grade (1.0 grade)
- questions in English, answers in English or German, **no tools neither aids**
- first date after Summer Semester, second after Winter Semester

Exam bonuses (only applied if passing with min. 20 points)

- 10% from mean of exercises evaluations (max. 4 exam points)
- 10% of project evaluation (max. 4 exam points)

→ *getting all bonus points get your grade from 4.0 to 2.7 (for instance)*

Exam and Grading

TUHH

Main written exam – 6 ECTS (Leistungspunkte)

- 90 min. duration, 40 exam points in total :
  → 20 points to pass (4.0 grade), 38 points for best grade (1.0 grade)
- questions in English, answers in English or German, **no tools neither aids**
- first date after Summer Semester, second after Winter Semester

Exam bonuses (only applied if passing with min. 20 points)

- 10% from mean of exercises evaluations (max. 4 exam points)
- 10% of project evaluation (max. 4 exam points)

→ *getting all bonus points get your grade from 4.0 to 2.7 (for instance)*

Already did the exercises last semester and collected bonus points ? ...

- highly recommended to do them again (different problems and tasks, practice opportunity)
- bonus points are kept over semesters, only best semester is used

More details and **reference solutions** for the last exams on Gitlab

Course content

TUHH

First part : Object Oriented Programming (OOP) in C++ (5 weeks)

- *Introduction to C++ and basic OOP (scopes, shadowing, overloading)* – 3 weeks
- *Advanced OOP : Inheritance and Polymorphism* – 2 weeks

Second part : Advanced C++ concepts (3 weeks)

- *Memory Management / Error Handling / Move Semantic & Template*

Third part : Python & Applications (6 weeks)

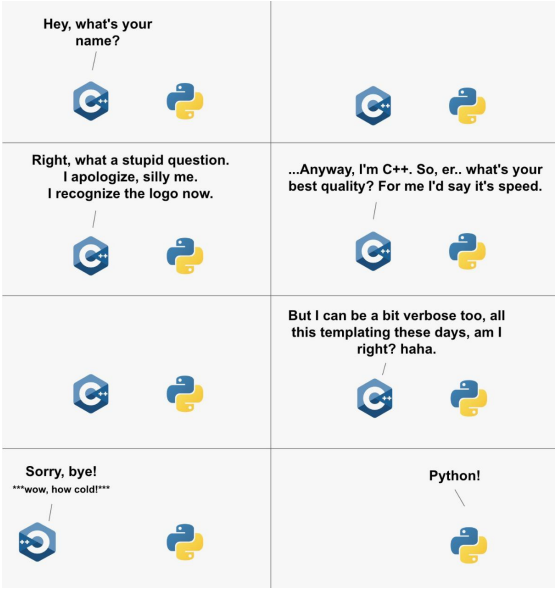
- *From C++ to Python, OOP in Python, other fancy stuff in C++ & Python*

In parallel : data versioning and collaboration with Git, code development with VSCode

Full detailed timeline on Gitlab, as usual ...

Some explanations ...

Why learning C++ and then Python ?

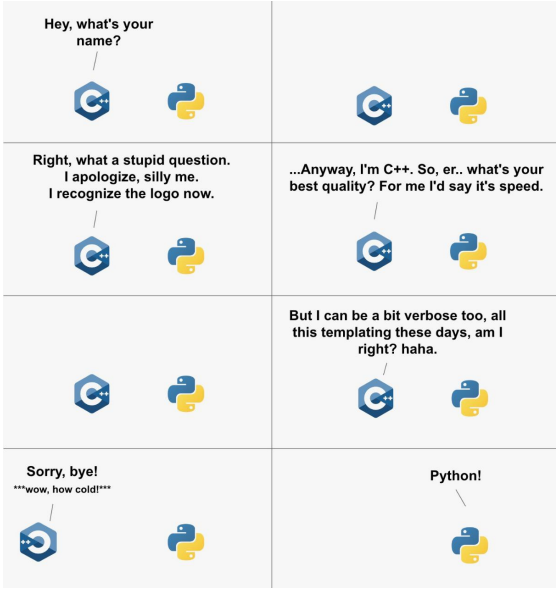


Some explanations ...

Why learning C++ and then Python ?

1) Both are Object Oriented, but :

- C++ is **very fast, but quite complex**
- Python is **way simpler, but very slow**



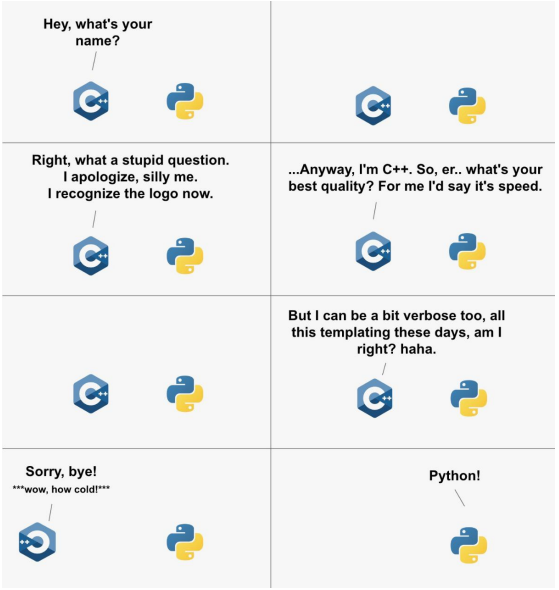
Some explanations ...

Why learning C++ and then Python ?

1) Both are Object Oriented, but :

- C++ is **very fast, but quite complex**
- Python is **way simpler, but very slow**

2) Python is the ground basis of many applications
→ data science, scientific computing, web, ...



Some explanations ...

Why learning C++ and then Python ?

1) Both are Object Oriented, but :

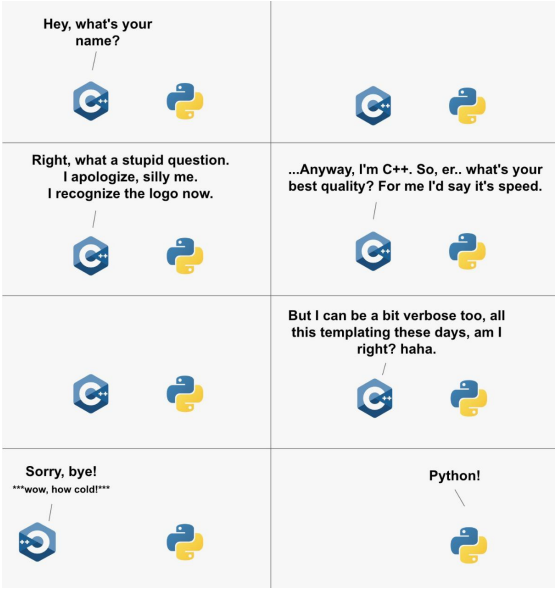
- C++ is **very fast, but quite complex**
- Python is **way simpler, but very slow**

2) Python is the ground basis of many applications
→ data science, scientific computing, web, ...

3) Knowing C++ makes Python learning easier !

"Hard Training, Easy War"

Proverb of the French Foreign Legion



Some explanations ...

Why learning C++ and then Python ?

1) Both are Object Oriented, but :

- C++ is **very fast, but quite complex**
- Python is **way simpler, but very slow**

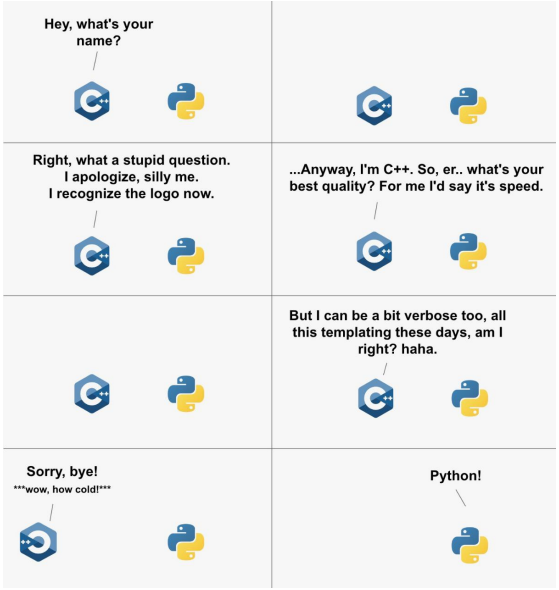
2) Python is the ground basis of many applications
→ data science, scientific computing, web, ...

3) Knowing C++ makes Python learning easier !

"Hard Training, Easy War"

Proverb of the French Foreign Legion

4) Competences in C, C++ and Python
⇒ ground basis for any other progr. languages



Support and communication during the semester

A) Everything is on Gitlab !

- access to up-to-date information and materials (slides, codes, exercises, previous exams, ...)
- keeping track of the main changes with the news page
- forum for questions and problems with Gitlab Issue (demonstration ...)
- ... but you still **have to register on StudIP** to receive group emails and notifications !

Support and communication during the semester

A) Everything is on Gitlab !

- access to up-to-date information and materials (slides, codes, exercises, previous exams, ...)
- keeping track of the main changes with the news page
- forum for questions and problems with Gitlab Issue (demonstration ...)
- ... but you still **have to register on StudIP** to receive group emails and notifications !

B) Move yourself too !

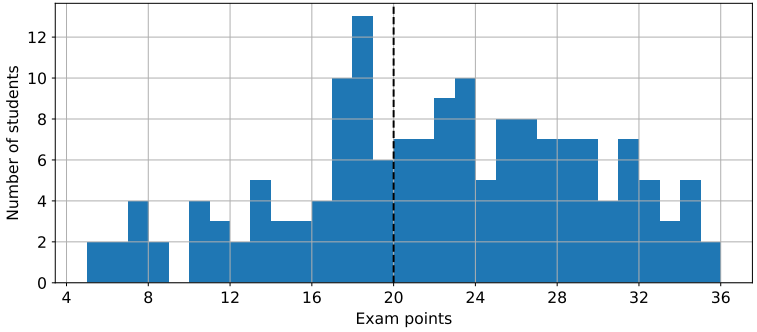
- do not hesitate to contact your tutors → list on Gitlab
- you can write me at thibaut.lunet@tuhh.de – **please don't use the messaging system of StudIP !**

Important detail : I'm still young and motivated ...

- you can drop by **any time** in my office CH4 1.021 (ACPS Institute) ...
- *... but to be sure that I'll be there and available, just* ***send me an email before !***

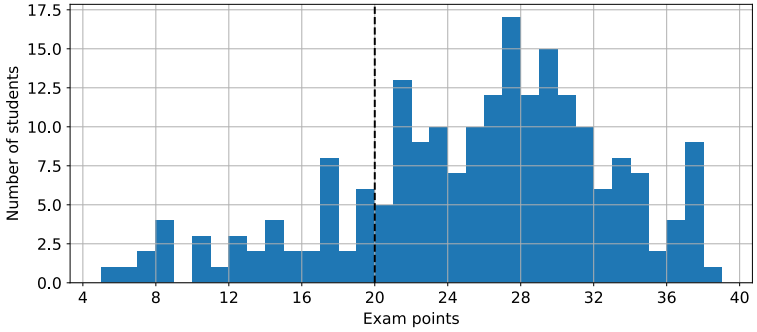
Time to wake up for the important warnings !

Grade distribution from SoSe23 exam : 164 corrected exams, 38.4% fail rate



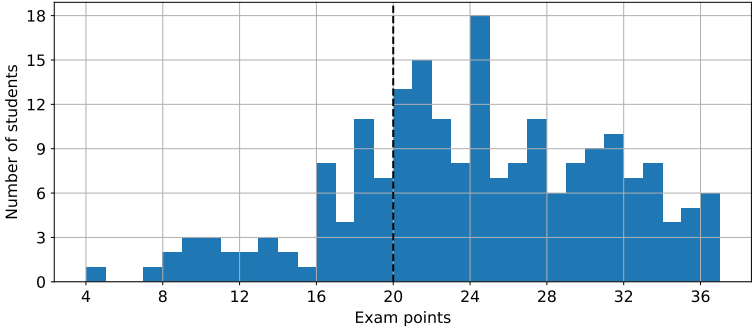
Time to wake up for the important warnings !

Grade distribution from SoSe24 exam : 210 corrected exams, 19.5% fail rate



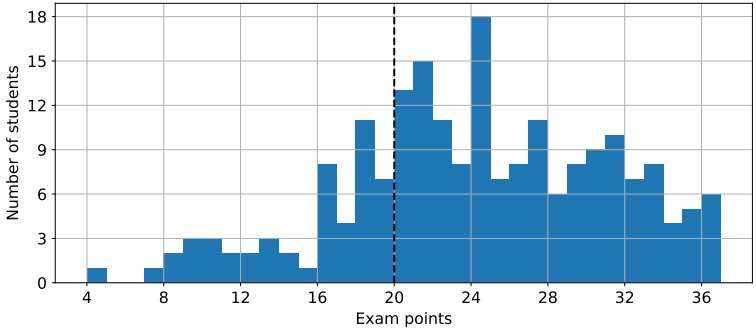
Time to wake up for the important warnings !

Grade distribution from SoSe25 exam : 204 corrected exams, 24.5% fail rate



Time to wake up for the important warnings !

Grade distribution from SoSe25 exam : 204 corrected exams, 24.5% fail rate



- **don't wait** for the last weeks before the exams to work !
- 6 ECTS ⇒ you are supposed to work ∼**9 hours** per week (in addition to VL, HÜ and PR ...)
- **regularly doing the exercises helps a lot**, see statistics ...

Note : tutorial on *"how to prepare for the exam"* at the end of the semester

For those who struggle ...

A bit of advices :

1. *never give up !*

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture
   - propose responses or comment Gitlab Issues opened by other students

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture
   - propose responses or comment Gitlab Issues opened by other students
   - use the student rooms (e.g in Channel 4, first floor) to work in groups

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture
   - propose responses or comment Gitlab Issues opened by other students
   - use the student rooms (e.g in Channel 4, first floor) to work in groups
3. *focus on the "one step at a time (ein Schritt nach dem anderen)"*

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture
   - propose responses or comment Gitlab Issues opened by other students
   - use the student rooms (e.g in Channel 4, first floor) to work in groups
3. *focus on the "one step at a time (ein Schritt nach dem anderen)"*
   - study **continuously over the semester**, don't work only one week before the exam

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture
   - propose responses or comment Gitlab Issues opened by other students
   - use the student rooms (e.g in Channel 4, first floor) to work in groups
3. *focus on the "one step at a time (ein Schritt nach dem anderen)"*
   - study **continuously over the semester**, don't work only one week before the exam
   - organize your work (week program, fixed hours for work, ...)

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture
   - propose responses or comment Gitlab Issues opened by other students
   - use the student rooms (e.g in Channel 4, first floor) to work in groups
3. *focus on the "one step at a time (ein Schritt nach dem anderen)"*
   - study **continuously over the semester**, don't work only one week before the exam
   - organize your work (week program, fixed hours for work, ...)
   - evaluate yourself, and what you still need to work on (with your tutor or professor)

For those who struggle ...

A bit of advices :

1. *never give up !*
   - use the Gitlab Issues (eventually private), your tutors (and your professor if needed)
   - continue attending lectures (VL), tutorials (HÜ) and exercise sessions (GÜ), **even if you're lost**
   - **don't wait for later** to solve your problems
2. *work with other students*
   - meet with your team (or others), and try to explain to them parts of the lecture
   - propose responses or comment Gitlab Issues opened by other students
   - use the student rooms (e.g in Channel 4, first floor) to work in groups
3. *focus on the "one step at a time (ein Schritt nach dem anderen)"*
   - study **continuously over the semester**, don't work only one week before the exam
   - organize your work (week program, fixed hours for work, ...)
   - evaluate yourself, and what you still need to work on (with your tutor or professor)

Checkout the "success stories" to get inspiration ...

Use of Generative AI (ChatGPT & friends)

Allowed, and even recommended in some cases ... **BUT**

1. **do not use it** as auto-completion tool (*e.g* GitHub Copilot) ... yet !

   → *first improve your coding skills (for 1 or 2 years), then* ***use AI as a skill multiplier***

Use of Generative AI (ChatGPT & friends)

Allowed, and even recommended in some cases ... **BUT**

1. **do not use it** as auto-completion tool (*e.g* GitHub Copilot) ... yet !
   → *first improve your coding skills (for 1 or 2 years), then* ***use AI as a skill multiplier***
2. it should never be the first "person" you ask !
   → *first read the course material, then ask others if you can (students, tutors, profs), then try the AI ...*

Use of Generative AI (ChatGPT & friends)

Allowed, and even recommended in some cases ... **BUT**

1. **do not use it** as auto-completion tool (*e.g* GitHub Copilot) ... yet !
   → *first improve your coding skills (for 1 or 2 years), then* ***use AI as a skill multiplier***
2. it should never be the first "person" you ask !
   → *first read the course material, then ask others if you can (students, tutors, profs), then try the AI ...*
3. always consider it as **potentially wrong or approximated**, and **talking way too much**
   → *use your own brain as an AI checker, summarize the output, optimize your prompts*

PS : even I can make some mistakes in the lecture content ... even intentionally !

Use of Generative AI (ChatGPT & friends)

Allowed, and even recommended in some cases ... **BUT**

1. **do not use it** as auto-completion tool (*e.g* GitHub Copilot) ... yet !
   → *first improve your coding skills (for 1 or 2 years), then* ***use AI as a skill multiplier***
2. it should never be the first "person" you ask !
   → *first read the course material, then ask others if you can (students, tutors, profs), then try the AI ...*
3. always consider it as **potentially wrong or approximated**, and **talking way too much**
   → *use your own brain as an AI checker, summarize the output, optimize your prompts*

PS : even I can make some mistakes in the lecture content ... even intentionally !

Usage recommendations

- generate questions / problems on a topic, and check your answers
- try different AI to compare their answers (like Le Chat, DeepSeek, Claude, ...)

... and always work on your capacity to synthesize !

Some final quotes ...

"Only grumpy narrow-minded people think they have nothing left to learn"

Some final quotes ...

"Only grumpy narrow-minded people think they have nothing left to learn"

"I'm not here to give you knowledge ..."
"... I'm here to help you improve your capacity to learn"

Some final quotes ...

"Only grumpy narrow-minded people think they have nothing left to learn"

"I'm not here to give you knowledge ..."
"... I'm here to help you improve your capacity to learn"

"Understanding is what makes you better than machine learning"

Some final quotes ...

"Only grumpy narrow-minded people think they have nothing left to learn"

"I'm not here to give you knowledge ..."
"... I'm here to help you improve your capacity to learn"

"Understanding is what makes you better than machine learning"

"There are no stupid questions ...
... just grumpy narrow-minded people who cannot answer properly"

Some final quotes ...

"Only grumpy narrow-minded people think they have nothing left to learn"

"I'm not here to give you knowledge ..."
"... I'm here to help you improve your capacity to learn"

"Understanding is what makes you better than machine learning"

"There are no stupid questions ...
... just grumpy narrow-minded people who cannot answer properly"

Bref : *I'm willing to give you as much as I can, but I won't chase you for that ...*
→ ***you'll have to do the first step !***