**Algorithmics and Programming II: Introduction**

Jordi Cortadella and Jordi Petit
Department of Computer Science

**Material**

- Slides, exercises:
  [https://www.cs.upc.edu/~jordicf/Teaching/AP2](https://www.cs.upc.edu/~jordicf/Teaching/AP2)
- Jutge (for lab sessions):
  [https://jutge.org](https://jutge.org)
- Lliçons (by J. Petit and S. Roura):
  [https://lliçons.jutge.org](https://lliçons.jutge.org)

**Lecturers:**
- Jordi Cortadella ([jordi.cortadella@upc.edu](mailto:jordi.cortadella@upc.edu))
- Emma Rollón ([erollon@cs.upc.edu](mailto:erollon@cs.upc.edu))
- Jordi Petit ([jordi.petit-silvestre@cs.upc.edu](mailto:jordi.petit-silvestre@cs.upc.edu))

**Sessions:**
- Theory & (Jordi C.)
- Lab (Emma & Jordi P.)
Evaluation

- Evaluation items:
  - Projects (Proj), Parcial Lab (PLab), Final Theory (FTh), Final (FLab).

- Grading:
  - $N_1 = 0.2 \text{Proj} + 0.25 \text{PLab} + 0.25 \text{FLab} + 0.3 \text{FTh}$
  - $N_2 = 0.2 \text{Proj} + 0.4 \text{FLab} + 0.4 \text{FTh}$
  - $N = \max(N_1, N_2)$

First project: Containers

- Design a class to manage containers.
- Language: Python.

Peer and self assessment

- The project will be evaluated by the students themselves.
- Each project will be evaluated by three students. The grade will be calculated as the average grade given by the students.
- The evaluation will be completely blind.
- Biased evaluations will be detected and penalized.
- Each student will have the right to request the evaluation by the professor (who can upgrade or downgrade the evaluation given by the students).

Donald Knuth (Turing award, 1974)

- “Programming is an art of telling another human what one wants the computer to do.”
- “An algorithm must be seen to be believed.”
- “The real problem is that programmers have spent far too much time worrying about efficiency in the wrong places and at the wrong times; premature optimization is the root of all evil (or at least most of it) in programming.”
Second Project: GPS

Language: Python

Objective of the course

Confronting large and difficult problems. How?

- Skills for abstraction and algorithmic reasoning.
- Design and use of complex data structures.
- Techniques for complexity analysis.
- Methodologies for modular programming.
- High-quality code.

Problems on polygons

Compute the convex hull of \( n \) given points in the plane.

The Closest-Points problem

- **Input**: A list of \( n \) points in the plane \( \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \)
- **Output**: The pair of closest points
- **Simple approach**: check all pairs \( \rightarrow O(n^2) \)
- We want an \( O(n \log n) \) solution!
Navigation: find the shortest path

How to encrypt messages?

The secret: training, training, training ...

... up to the finish line