Algorithmic Strategies 2024/25

Week 1 – Introduction

1 2 9 0

UNIVERSIDADE Đ
COIMBRA

Outline

1. Introduction
2. Assessment
3. Contents
4. Bibliography

Introduction

Luís Paquete

- T + TP + PL1 + PL2
- paquete@dei.uc.pt
- D.2.18 by e-mail
- Fri, 9h-13h30

Noé Godinho

- PL4+PL5+PL6
- noe@dei.uc.pt
- D.3.13 by e-mail
- Tue, 12h-13h, 18h-20h
- Thu, 12h-13h, 18h-19h

Henrique Branquinho

- PL3
- branquinho@dei.uc.pt

Introduction

Algorithmic Strategies: what is it about?

- Course about programming and algorithms
- Focused on problem solving through the study of algorithm paradigms and their application to problems in different domains: Optimization, Graphs, Geometry, Strings, . . .
- Re-uses programming techniques that students already know
- At the end of the semester, the student should know how to
  - Choose an approach to solve a given problem
  - Show that the approach is correct
  - Implement the code in an efficient way

Introduction

This can be useful to:

- Find the shortest path in a navigation system
- Detect path similarity from GPS traces
- Place readers in a building
- Find similarities between DNA sequences

Introduction

This can be useful to:

- Find the shortest path in a navigation system
- Detect path similarity from GPS traces
- Place readers in a building
- Find similarities between DNA sequences
- Job interviews!

Introduction

How it works?

- EA covers 8 algorithmic topics.
- T class: Introduction to the topic and to the programming problem.
- TP class: Summary, exercises, follow-up of the programming problem.
- PL class: Support on programming and written exercises, clarification of doubts and individual training.
- Assignments: programming problems and written exams.

Assessment

A) 4.5 points in two programming problems

B) 0.5 points in PL programming exercises

C) 15.0 points in one written exam.

Minimum requirements: 35% in C)

Retake and special exam only for C)

Beware of borrowing code from others or the Internet!

Assessment

A - Programming problems

- Two problems to be solved in teams of two with Mooshak, partial points are allowed.
- Test cases have different levels of difficulty.
- Students may need to orally defend the grade obtained on both programming problems.
- Register at mooshak with team name

  <student 1 ID>-<student 2 ID>

  Nonconforming registrations will be deleted from mooshak.

Assessment

B - PL Programming exercises

- Maximum 10 problems to be solved individually with Mooshak, 0.05 points (accepted in mooshak) each. Consider this to improve your coding skills.
- A problem is released every week after T class and stays open for two weeks. The solution to the problem is discussed in the PL classes in the first week after the release.
- Register at mooshak (EA2025_PL) with team name <student ID>. Nonconforming registrations will be deleted from mooshak.

Assessment

C - Written exam

- Totals 15 in the official scale.
- One exam to be solved individually
- Closed books

Calendar

Week	Day	Topic
1	6 Feb	Intro and problem solving
2	13 Feb	Recursion (no T in 20/2) & Problem 1 (18/2)
3	27 Feb	Backtracking
4	6 Mar	Dynamic programming
5	13 Mar	Dynamic programming
6	20 Mar	Greedy algorithms & 1st Deadline (19/3)
7	27 Mar	Branch and bound
8	3 Apr	Graph algorithms
8	10 Apr	Graph algorithms & Problem 2 (22/4)
9	24 Apr	Graph algorithms
10	30 Apr	Graph algorithms
11	8 May	Computational Geometry
12	15 May	Computational geometry
13	22 May	TBD & 2nd Deadline (23/5)

Deadlines for problems are at 23h59m

Assessment

- Teams: Students **must** split into groups of two. Students cannot change group after team assessment begins. Meet your team member at least once a week to solve problems.
- Programming languages: Use `C`, `C++`, `Java` and `Python`. However, some large test cases may not be suitable for `Java` and `Python`.
- Use mooshak, PLs, teacher attendance timetable, and slack for clarification of doubts. Note that the teaching staff may not answer all questions in the mooshak.
- Mooshak automatically opens and closes contests. No delays are tolerated!

Bibliography

Main references

- Algorithms, Jeff Erickson (draft available here)
- Introduction to Algorithms, T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein

Additional references

- Programming Challenges, S. Skiena, M. Revilla
- Algorithm Design, J. Kleinberg, E. Tardos
- The Algorithm Design Manual, S. Skiena (2nd edition)
- How to Think About Algorithms, J. Edmonds
- The Art of Computer Programming, D. Knuth
- C++ STL and JDK