VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE | DEPARTMENT OF COMPUTER SCIENCE

Algorithms I – Subject Syllabus

Jiří Dvorský, Ph.D.

Presentation status to date February 24, 2025

Department of Computer Science
VSB – Technical University of Ostrava

Lecture outline

Algorithms I – Subject Syllabus

- About Algorithms I
- Fulltime Study
  - Teaching
  - Tasks and their Evaluation
- Software
- Study Literature

Algorithms I – Subject Syllabus

About Algorithms I

About Algorithms I

Attention

For all the latest information on the subject, please see

http://www.cs.vsb.cz/dvorsky/

This presentation is for introductory lecture purposes only and will not be updated.

About Algorithms I

- The subject covers basic algorithmic problem solving strategies (brute force, divide and conquer, etc.) and typical examples of their use.
- Lectures are focused on **theory**.
- Seminars are focused on problem solution **implementation** using a given strategy in C or C++.
- Algorithms I are related to other subjects:
    - Introduction to programming – C language,
    - Functional programming – recursion and
    - Object-oriented programming – probably no commentary needed.

Time Allocation, Evaluation of the subject

Time Allocation

- Subject is taught in the summer semester of the first year of the bachelors study.
- There are
  - 2 hours of lectures and 2 hours of exercises per week in full-time form and
  - 6 tutorials in the combined form of study.

Evaluation – marked credit

- Marked credit is not an exam, it follows different rules.
- Please read the Study and Examination Regulations for Study in Bachelor’S and Master’S Degree Programmes at VSB - Technical University of Ostrava, Article 12.

Subject Guarantor

doc. Mgr. Jiří Dvorský, Ph.D.

Office:	EA441
Email:	*jiri.dvorsky@vsb.cz*
Web:	*www.cs.vsb.cz/dvorsky*

What's the subject guarantor for?

The guarantor is responsible for the course of the entire subject, is responsible for teaching and correct evaluation of the assignments.

Prerequisities

- Prerequisites are a set of requirements that must be met in order for a student to enrol in a subject. Prerequisites are either formal or substantive.
- Formal prerequisites – none
- Substantive prerequisites:
  - knowledge from Introduction to Programming,
  - high school mathematics and
  - general orientation in IT.
- The subject Algorithms I is a mandatory prerequisity of the follow-up subject Algorithms II.

Attendance

Lectures

- Attendance at the lectures is **highly recommended**.

Seminars

- Attendance is **mandatory**.
- Attendance and activity at the seminars are evaluated.
- Sufficient scores must be obtained.

Consultation

- If you don't understand something in class, need help with something or solve a problem with a lecture, seminars, tests, your absence from class, etc. it is possible to arrange a **individual consultation**.
- The consultation must be arranged in advance, for example by e-mail.
- If you need help with the material, prepare the materials you have studied on the topic, write down what is clear to you and where you are "stuck" and need advice.
- You don't risk anything by consulting the teacher – at most you will learn what you need.

Algorithms I – Subject Syllabus

Fulltime Study

Lecture topics

1. General information about the subject
2. **Introduction**
   - 2.1 What Is an Algorithm?
   - 2.2 Fundamentals of Algorithmic Problem Solving
   - 2.3 Important Problem Types
   - 2.4 Fundamental Data Structures
3. **Fundamentals of the Analysis of Algorithm Efficiency**
   - 3.1 The Analysis Framework
   - 3.2 Asymptotic Notations and Basic Efficiency Classes
   - 3.3 Mathematical Analysis of Nonrecursive Algorithms
   - 3.4 Mathematical Analysis of Recursive Algorithms
4. **Brute Force and Exhaustive Search**

Lecture topics (cont.)

4.1 Selection Sort and Bubble Sort

4.2 Sequential Search and Brute-Force String Matching

4.3 Closest-Pair and Convex-Hull Problems by Brute Force

4.4 Exhaustive Search

4.5 Depth-First Search and Breadth-First Search

5. Decrease-and-Conquer

5.1 Insertion Sort

5.2 Topological Sorting

5.3 Algorithms for Generating Combinatorial Objects

5.4 Decrease-by-a-Constant-Factor Algorithms

5.5 Variable-Size-Decrease Algorithms

6. Divide-and-Conquer

6.1 Mergesort

Lecture topics (cont.)

6.2 Quicksort

6.3 Binary Tree Traversals and Related Properties

6.4 Multiplication of Large Integers and Strassen’s Matrix Multiplication

6.5 The Closest-Pair and Convex-Hull Problems by Divide-and-Conquer

Seminars

- Semminars corresponds to lectures.
- In the seminar, students implement given tasks in C++ language.
- It is also possible to consult the lecture material.
- The seminar is not a substitute for lecture!
    - The seminars are not a “brief lecture” for those who do not attend lectures.
    - It is necessary to be prepared for the seminars.
    - The purpose of the seminar is not to prepare for the final exam.

Tasks

- The assessment consists of three parts:
  1. **Ongoing activities on seminars**
  2. **Project defense**
  3. **Final written test**
- All assignments are mandatory.
- A minimum grade is required for each assignment.

Tasks – Ongoing activities on seminars

- This part of the assessment is done **ongoing throughout the semester**.
- At each exercise, your activity is evaluated by the teacher. The activity is graded using a colour code:
  - **green** – the student actively participated in the seminar, was familiar with the material, he/she was able to carry out the assigned tasks,
  - **orange** – the student was rather passive in the seminar, he/she was not very well prepared for the seminar, his/her knowledge was limited, he/she had problems with the implementation of the tasks, and

Tasks – Ongoing activities on seminars (cont.)

  - **red** – the student was rather passive in the seminar, he/she was unable to complete the assignments. Unexcused absence from the exercise also falls into this category.

- Each colour code corresponds to a certain weight, that is reflected in the overall evaluation of all seminars.

Color code	Weight
green	1
orange	0.5
red	0

- At the end of the semester, an average weight is calculated, multiplied by the maximum number of points possible (30), and the result is your score.

Tasks – Ongoing activities on seminars (cont.)

- It is clear that all green codes correspond to the maximum number of points (30), while all red codes correspond to zero points.
- Activity points cannot be redeemed.

Example

The student A received a green rating on five seminars, orange on three and red on two ones. The average weight is calculated as:

$$\frac{5 \times 1 + 3 \times 0.5 + 2 \times 0}{5 + 3 + 2} = \frac{6.5}{10} = 0.65.$$

So the final score is $0.65 \times 30 = 19.5 \approx 20$ points.

Tasks – Project defense

- The project assignment will be published on the subject website at the beginning of April.
- Deadline for submission will be around credit week. The exact date will be published in the project assignment.
- The method of submission will be determined later.
- Project defenses will take place during the credit week and the exam period.
- Regardless of when the project defences take place, the version that has been submitted by the deadline is defended.
- The project defence cannot be repeated and the project will not be returned for revision.

Tasks – Final written test

- The test will take place during the exam period.
- All test dates will be announced in Edison system.
- Only students who have scored at least 10 points on their first attempt will be allowed to retake the test.

Number of points in the first attempt	Retake the test
0 to 9	no
10 to 20	yes
more than 21	not necessary

- You are allowed to write the final test a total of **two times**, in other words you are entitled to **one correction**. The course is completed with a marked credit not an exam – the rules are different.

Algorithms I – Subject Syllabus

Software

Software

Primary Software

- C++ Development Environment
- C++ Documentation

Additional Software

- Doxygen Documentation System, *www.doxygen.org*
- Typography System LaTeX, *www.ctan.org*

Development Environment for C++

- Microsoft Visual Studio Community 2022 is available for classroom use.
- I recommend this development environment for home study.
- In general, any development environment with a compiler that supports at least the C++17 specification can be used.

Development Environment for C++ (cont.)

Remarks

1. The **Microsoft Visual C++** compiler and the **C++17** language specification will be used to evaluate your projects.
2. The C language is not identical to C++!
3. Beware of non-standard C++ language extensions implemented in the GNU C++ compiler.
    - For example, a variable length array is such an extension.
    - It is recommended to compile with the *`-pedantic-errors`* option enabled, see Options to Request or Suppress Warnings.

Algorithms I – Subject Syllabus

Study Literature

Study Literature

The study literature can be divided into two groups:

- **mandatory literature** – strategies of algorithmic problems solving and
- **recommended literature** – C++ programming language.

The study literature is shared across Algorithms I and Algorithms courses.

Mandatory Study Literature

1. LEVITIN, Anany. *Introduction to the Design and Analysis of Algorithms*. 3rd ed. Boston: Pearson, 2012. ISBN 978-0-13-231681-1.
2. CORMEN, Thomas H., Charles Eric LEISERSON, Ronald L. RIVEST a Clifford STEIN, 2022. *Introduction to algorithms*. Fourth edition. Cambridge, Massachusetts: The MIT Press. ISBN 978-026-2046-305.
3. SEDGEWICK, Robert, 1998. *Algorithms in C++*. 3rd ed. Reading, Mass: Addison-Wesley. ISBN 978-020-1350-883.

Recommended study literature

1. STROUSTRUP, Bjarne., 2013. The C++ programming language. Fourth edition. Upper Saddle River, NJ: Addison-Wesley. ISBN 978-0321563842.
2. CADENHEAD, Rogers a Jesse LIBERTY, 2017. Sams teach yourself C in 24 hours. Sixth edition. Indianapolis, Indiana: Pearson Education. ISBN 978-0672337468.

Thanks for your attention