# Base information about the Learning Unit

Fabian Stiewe

Academic Year 2024-2025

# Contents

# 1 School and Year

This learning unit is designed for the 4th year of the **LSSA - Liceo Scientifico Scienze Applicate**. It will be taught at the beginning of the academic year.

# 2 Topics to be covered

The main topic is theoretical computer science, inside this huge field we will focus on the following topics:

- Sorting algorithms

  - Bubble sort

  - Selection sort

  - Merge sort

– Sorting on presorted data
- Searching algorithms
  – Linear search
  – Binary search
  – Hash tables

## 2.1 Objectives Sorting Algorithms

### 2.1.1 Bubble Sort

The students should understand how the **bubble sort** algorithm works and be able to implement it in their preferred programming language. They should also be able to analyze the time complexity *(not with big-O notation)* of the algorithm and understand why it is not efficient for large data sets.

### 2.1.2 Selection Sort

The students should understand how the **selection sort** algorithm works and be able to implement it in their preferred programming language. They should also be able to analyze the time complexity *(not with big-O notation)* of the algorithm and understand why it is not efficient for large data sets.

### 2.1.3 Merge Sort

The students should understand how the **merge sort** algorithm works and be able to implement it in their preferred programming language. A time complexity is **not** required, but the students should understand that it is more efficient than bubble sort and selection sort.

### 2.1.4 Sorting on Presorted Data

Given different data sets, the students should be able to choose the best sorting algorithm for each data set. They should also be able to explain why a specific sorting algorithm is better for a specific data set.

In example: If the data set is already sorted, except for $k$ elements, which sorting algorithm should be used?

## 2.2 Objectives Searching Algorithms

### 2.2.1 Linear Search

The students should understand how the **linear search** algorithm works and be able to implement it in their preferred programming language. They should also be able to analyze the time complexity *(not with big-O notation)* of the algorithm and understand why it is not efficient for large data sets.

### 2.2.2 Binary Search

The students should understand how the **binary search** algorithm works and be able to implement it in their preferred programming language. They should also be able to analyze the time complexity *(not with big-O notation)* of the algorithm and understand why it is more efficient than linear search.

### 2.2.3 Hash Tables

The students should understand how **hash tables** work and be able to use them in their preferred programming language. They should understand why it is efficient for searching and inserting elements and be able to explain why it is more efficient than binary search (in special cases).

# 3   Perquisites

There are no formal prerequisites for this learning unit. However, it is recommended that students have a basic understanding of the following topics:

- Basic programming concepts

  - The programming language does not matter, each student can use the language they are most comfortable with
  - Variables, loops, functions, arrays, etc. should be known

- Basic mathematics concepts

  - Basic algebra
  - log function
  - Very basic probability (there won't be a proof for hash tables, but it helps to understand the concept)

# 4   Time plan for the Learning Unit

The learning unit will take 9 weeks. Each week will be divided into two parts: **theory** and **practice**. The theory part will be taught first, and the practice part will be taught (in the laboratory) in the second half of the week. During the practice part, the students will get a simple problem and should implement a solution using the algorithms they learned in the theory part.

## 4.1   Week 1: Bubble Sort

Using a presentation, the students will get taught how the bubble sort algorithm works. They will also get some examples and the "Pseudocode" of the algorithm.

In the laboratory, the students will get a simple problem and should implement a solution using the bubble sort algorithm.

### Problem for the laboratory

To start the course the students will get a simple problem: Given an array of integers, sort the array using the bubble sort algorithm. The array should be sorted in ascending order. *(This is a good start to warm up programming skills of each student.)*

## 4.2   Week 2: Selection Sort

Using a presentation, the students will get taught how the selection sort algorithm works. They will also get some examples and the "Pseudocode" of the algorithm. Also, an application example will be shown, quite similar to the problem for the laboratory.

### Problem for the laboratory

> You are given an array of strings `names`, and an array `heights` that consists of distinct positive integers. Both arrays are of length $n$. For each index $i$, `names`[$i$] and `heights`[$i$] denote the name and height of the $i$th person. Return `names` sorted in descending order by the people's heights.

The problem will should be solved using the selection sort algorithm. The students will get test cases to check if their implementation is correct.

## 4.3   Week 3: Merge Sort

Todo

## 4.4 Week 4: Sorting on Presorted Data

Todo

## 4.5 Week 5: Recap and evaluation of the sorting algorithms

Todo

## 4.6 Week 6: Linear Search

Todo

## 4.7 Week 7: Binary Search

Todo

## 4.8 Week 8: Hash Tables

Todo

## 4.9 Week 9: Recap and evaluation of the searching (and sorting) algorithms

Todo

# 5 Disabilities

Todo

# 6 Evaluation

The main part of the evaluation will be the programming during the laboratory. The students will get a problem and should implement a solution using the algorithms they learned in the theory part. The implementation should be correct and efficient, or at least the students should be able to explain why their implementation is not efficient.