# Exercises: Code Tuning and Optimization

This document defines the **in-class exercises** assignments for the ["High-Quality Code" course @ Software University](https://softuni.bg/courses/high-quality-code).

## Compare Sorting Algorithms

Write (or copy from the Internet) some implementations of sorting: insertion sort, selection sort, merge sort, quick sort. Compare their performance. You can look at the **System.Diagnostics.Stopwatch** class for a way to calculate the time a method takes to run.

Pass a few parameters and see how long it takes for the method to finish. Fill in the table below. Write "hangs" if the execution does not finish within 45-60 seconds.

You can optionally pass some more intermediate values and make a plot: place the parameter on the X axis, and the execution time on the Y axis.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | **n = 10** | **n = 50** | **n = 100** | **n = 1000** | **n = 10 000** | **n = 100 000** | **n = 1 000 000** | **n = 10 000 000** |
| **Insertion** |  |  |  |  |  |  |  |  |
| **Selection** |  |  |  |  |  |  |  |  |
| **Merge** |  |  |  |  |  |  |  |  |
| **Quick** |  |  |  |  |  |  |  |  |

## Compare Data Structures

Write a phone book application, containing people **names** and **phones**. Use two approaches: using a **List<Person>** (string name, string phone) and a **Dictionary<string, string>** (key: name, value: phone).

Write a method to search for a given person's phone. Using the list, you have to search in the entire list. Using the dictionary, you can select the index directly.

Generate a lot of people and perform a lot of searches (n = number of calls to the **Search(string personName)** method). See how long it takes for the method to finish. Fill in the table below. Write "hangs" if the execution does not finish within 45-60 seconds.

You can optionally pass some more intermediate values and make a plot: place the parameter on the X axis, and the execution time on the Y axis.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Approach** | **n = 10** | **n = 50** | **n = 100** | **n = 1000** | **n = 10 000** | **n = 100 000** | **n = 1 000 000** | **n = 10 000 000** |
| **List** |  |  |  |  |  |  |  |  |
| **Dictionary** |  |  |  |  |  |  |  |  |

Make a second table, where the number of searches is constant (say, 1000), and n = number of people. See how the results differ.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Approach** | **n = 10** | **n = 50** | **n = 100** | **n = 1000** | **n = 10 000** | **n = 100 000** | **n = 1 000 000** | **n = 10 000 000** |
| **List** |  |  |  |  |  |  |  |  |
| **Dictionary** |  |  |  |  |  |  |  |  |