# Algorithms and Data Structures Review

Angelos-Christos Anadiotis

*Practice Session*

# Integrated Development Environment

- Software which helps into building software
- Typically includes an editor bundled with a lexical and syntax analyzer for various programming languages
- Modern IDEs also include compilers, debuggers, profilers, etc
- Popular IDEs:
  - Intellij IDEA (Java, Scala, Python, Web, etc)
  - CLion (C/C++)
  - Eclipse (most popular programming languages with different plugins)

# Install or Start your favorite IDE

- I use Java and IntelliJ IDEA, but you are free to use the PL and IDE of your choice
- Today's session mostly focuses on algorithmic aspects and requires knowledge of only fundamental data structures provided by every PL

# Exercise 1

Suppose that you have two strings as input from the user. Write a program that checks whether one string is permutation of the other.

Note: A permutation of a string has exactly the same characters, but they may appear in different order.

# Exercise 2

Suppose that you have one string as input from the user. Write a program which checks whether there exists another string that is a permutation of the input and is a palindrome.

Note: Palindrome is a string which can be read in the same way from the beginning and from the end.

# Exercise 3

Suppose that you have a linked list of integers. Every element of that list may appear several times. Write a program which returns a new linked list including only one appearance of every element of the original list.

Bonus question: How would you implement this in SQL, supposing that instead of linked list, you have a relational database table?

# Exercise 4

Write a program which implements a stack of integers. Apart from the push and pop, the stack must also support a min operation that returns the minimum value stored in the stack. The computational complexity of all operations must be O(1).

# Exercise 5

Write a program that implements a queue using stacks.

Note: Stack implements LIFO order and Queue implements FIFO order.

# Exercise 6

Write a program that implements a tree data structure. The design should allow extensions to support standard trees like BSTs, AVL trees, etc.

# Exercise 7

Write a program that checks whether a binary search tree is balanced.