Object-oriented Programming

in (C and) C++

Bachelor of Science - École polytechnique

gael.thomas@inria.fr

Goal of the course

■ Master the C and the C++ programming languages

- Imperative programming: type, array, pointer, function, dynamic memory management, modular programming
- Object-oriented programming: class, reference, template

■ Understand how a computer runs applications at low level

Object-oriented programming in C++ Introduction

Why C and C++

- C++ is a kind of "super" C => learning both languages at the same time make sense

- Most popular programming language (Tiobe index 2024/08)
  1. Python: 18.04%
  2. **C++: 10.04%**
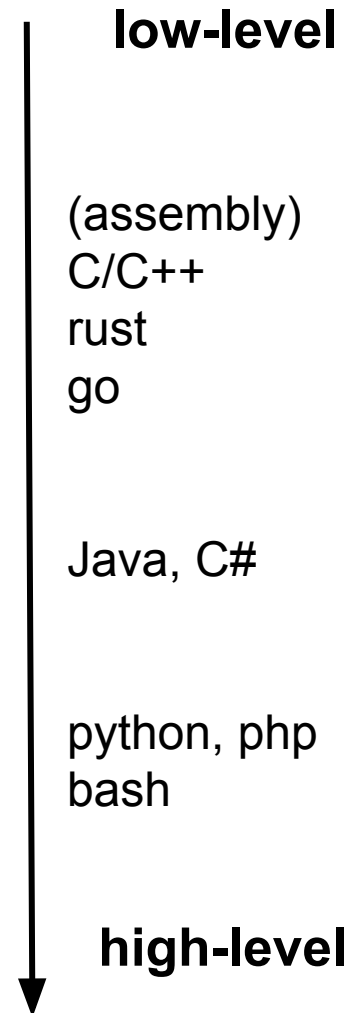 3. **C: 9.17%**
  4. Java: 9.16%

19.21% => C/C++ remain very popular programming languages

Object-oriented programming in C++   Introduction

Low-level vs high-level languages

- Low-level programming language
  - Exposes the low-level details of the machine to the developer
  - Efficient but difficult to master
    (memory corruption, poor support from the environment)
  - => used to develop operating systems, video game engine, HPC applications, large-scale web applications, AI engines…

- High-level programming language
  - Hide most of the low-level details of the machine
  - Easy to master but inefficient
    (runtime checks, garbage collectors, sometimes interpreted)
  - => used to develop demonstrators or as a front-end to use low-level libraries
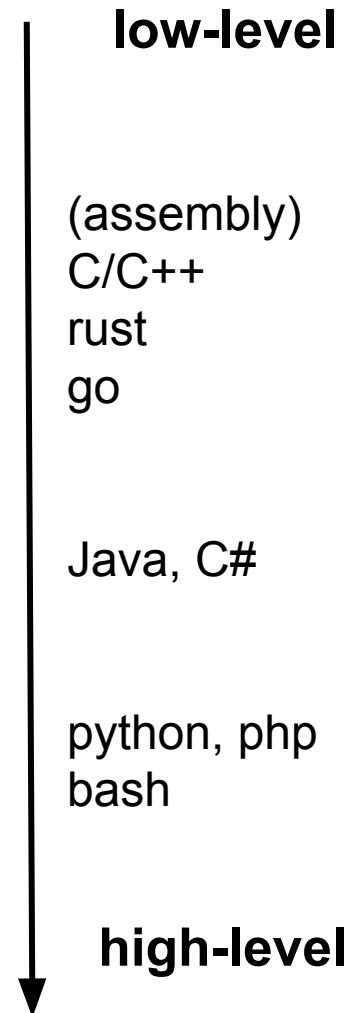
Low-level vs high-level languages

- ■ Low-level programming language
  - Exposes the low-level details of the n
  - Efficiency but difficult to master
    (memory corruption, poor support fro
  - => used to develop operating system
    applications, large-scale web applica

- ■ High-level programming language
  - Hide most of the low-level details of t
  - Easy to master but inefficient
    (runtime checks, garbage collectors,
  - => used to develop demonstrators or
    low-level libraries

low-level

(assembly)
C/C++
rust
go

Java, C#

python, php
bash

high-level

Object-oriented programming in C++   Introduction

Low-level vs high-level languages

■ Illustration : find the max in an array
(10^6 elements, without using a library)
  • C or C++: 4.7ms
  • python: 692.1ms => 147x slower!

low-level

(assembly)
C/C++
rust
go

Java, C#

python, php
bash

high-level

Object-oriented programming in C++   Introduction

Organization

- 14 weeks in two waves
  - Weak 1 to 7: the C and C++ programming language
    - Weak 1 to 3: the C programming language
    - Weak 4 to 7: the C++ programming language
  - Weak 8 to 14: long-term project in C++

- Each week:
  - Lecture: 1h30
  - Lab: 2h
  - Homework to finish the lab: 2h

Object-oriented programming in C++ Introduction

Prerequisites

- None!

- If you are not good at programming, this course is the perfect opportunity to become a master!

- If you are already a guru, you will also learn a lot :)

- Note: do all the exercises even if some look hard, it's by coding yourself that you will learn how to code

Object-oriented programming in C++   Introduction

Final grade

- The final grade is composed of three grades:
  - Graded labs: 10%
  - Final exam: 40%
  - Project: 50%

Please, don't cheat :)

chatgpt or your best friend can help you, but what you will learn here will be very useful for the remainder of your study

(even if you don't want to become a computer scientist! programming is today essential for any science)

Before the first lab

■ In the labs, we will use a Unix system

■ Before the first lab you have to install it
  - For the Linux user, it's a Unix system
  - For the Windows user, install WSL (Windows Subsystem for Linux)
  - For the MacOS, it's a Unix system
    – But you need macport or brew

■ Before the first lab, you have to install a Unix system

https://cs.ip-paris.fr/courses/cse201/Supports/?page=annexe-software