EE2-12 Software Engineering 2: Object-Oriented Programming

Week 0 - Module presentation

Sahbi Ben Ismail

Imperial College London, Department of Electrical and Electronic Engineering

Imperial College London

Autumn Term 2018-19

Teaching Team, contact details

Sahbi Ben Ismail - Lectures & Labs

- Teaching Fellow in Software Engineering
- Room 603
- s.ben-ismail@imperial.ac.uk (subject starts with [EE2-12])
- Piazza: Q&A [more details during Lab 1]

Graduate Teaching Assistants (GTAs) - Labs

- Mark Zolotas, mark.zolotas12@imperial.ac.uk
- Jonathan Zheng, jonathan.zheng12@imperial.ac.uk

Module prerequisites

At the beginning of the module, you should be able to:

- Define and use basic data types in C++
- Declare and call functions in C++
- Declare and use pointers and structures in C++

EIE1 modules:

- EE1-07 Introduction to Computing
- EE1-08 Algorithms & Data Structures

Module Intended Learning Outcomes (ILOs)

By the end of the module you should be better able to:

1. Describe the key concepts and vocabulary of Object-Oriented Programming (OOP): classes, objects, **abstraction**, **encapsulation**, **inheritance**, and **polymorphism**[a]
2. Build new classes from other classes using **aggregation/composition** and **association**
3. Describe a software architecture using the Unified Modeling Language (**UML**) notations (**class diagrams**)
4. Design and write code with **polymorphic** behaviour using **inheritance**
5. Use **template** classes and apply **operator overloading**
6. Implement **exception** handling
7. Apply OOP concepts in other object-oriented programming languages (e.g. **Java**)

[a] called "the OOP Big Four"

Module Syllabus

Week 1	Classes and Objects I: Introduction
Week 2	Classes and Objects II: Constructors, and Operator Overloading
Week 3	Objects and Dynamic Memory
Week 4	Classes Relationships I: Association, Aggregation, and Composition
Week 5	Classes Relationships II: Generalisation/Inheritance
Week 6	Polymorphism and Virtual Functions
Week 7	Generic Programming: Templates, and the Standard Template Library (STL)
Week 8	Exceptions Handling
Week 9	C++ to Java
Week 10	Revision

Teaching method

Week N = {Lecture N, Lab N}

- **Lectures**: interactive, with in-class coding and quizzes, *Pomodoro* technique
- **Labs**: application of the lectures topics, with more challenging questions and optional problems
- **Team-Based Learning (TBL)** - like session to study an open-source software written in C++

Main Learning Technologies

- **BB**: module teaching materials + Questions bank
- **Mentimeter**, **Panopto** [autochaptering not automatically generated, to be done collaboratively]
- **Piazza**: collaborative Q&A
- Short **recorded videos** for pre/post sessions

Blended learning for a better learning journey, hopefully!

Assessment & Feedback

Formative Assessments

For feedback and development purposes; do not count towards the module grade

- Labs
- In-class MCQs
- BB questions bank (self assessment)
- TBL-like session

Summative Assessments

- **1h Lab assessment**[a]: **Wednesday 12/12/2018**, two groups 09:00-10:00 & 10:00-11:00.
- **2h final written Exam**[b]: **Monday 03/06/2019** 10:00-12:00.

---

[a] **counts for the Second Year Computing Lab module**

[b] **A revision session should be scheduled in May 2019. date TBC.**

Have your say

For you and the next year(s) EIE2 students

Your feedback is vital!

Short-term level Make adjustments during the ongoing module [the teaching team is approachable]

Mid-term level How should EE2-12 look like in 2019-20?

Long-term level Curriculum Review

Surveys: write comments!

- SOLE
- NSS

Books (future readings) I

[There is no Textbook for the module.]

Module [Savitch(2016)]

More C++ [Stroustrup(2018), Meyers(2014)]

Java [Savitch(2015)]

Design Patterns [Gamma(1995)]

UML [Rumbaugh et al.(2004)Rumbaugh, Jacobson, and Booch, Fowler(2004)]

Soft. Engineering [Sommerville(2015)]

Martin Fowler.

UML distilled: a brief guide to the standard object modeling language.

Addison-Wesley Professional, 2004.

Books (future readings) II

Erich Gamma.

Design patterns: elements of reusable object-oriented software.

Pearson Education India, 1995.

Scott Meyers.

Effective modern C++: 42 specific ways to improve your use of C++ 11 and C++ 14.

O'Reilly Media, 2014.

James Rumbaugh, Ivar Jacobson, and Grady Booch.

Unified modeling language reference manual, the.

Pearson Higher Education, 2004.

Walter Savitch.

Absolute Java.

Pearson, 2015.

Books (future readings) III

Walter J Savitch.

Absolute C++, Global Edition.

Pearson Education, 2016.

Ian Sommerville.

Software Engineering (10th Edition).

Pearson, 2015.

Bjarne Stroustrup.

A Tour of C++.

Addison-Wesley Professional, 2018.

Additional resources

ECOOP - European Conference on Object-Oriented Programming

- Open Access
- https://2018.ecoop.org/

cppCon - The C++ Conference

- https://cppcon.org/
- https://www.youtube.com/user/CppCon
- https://github.com/CppCon/
- 2014-15-16-17, demos, keynotes, presentations, tutorials

- The Evolution of C++ - Past, Present, and Future by **Bjarne Stroustrup**, https://github.com/CppCon/CppCon2016
- http://cppcast.com/2017/05/bjarne-stroustrup/