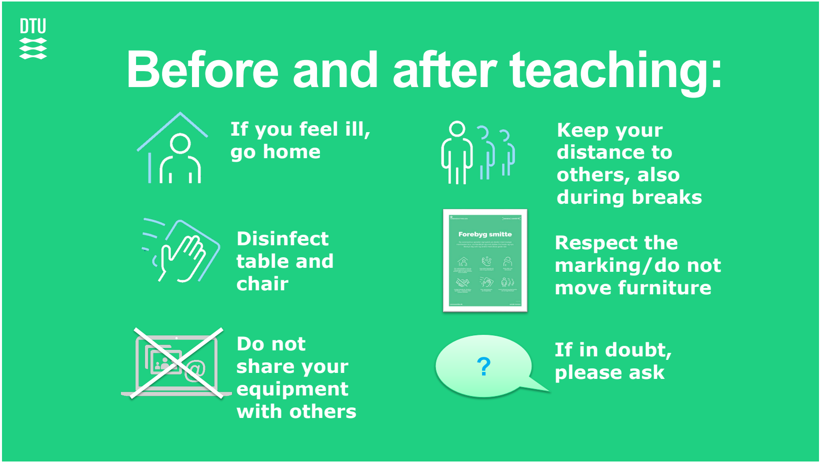
02393 Programming in C++



Before and after teaching:

If you feel ill, go home

Keep your distance to others, also during breaks

Disinfect table and chair

Respect the marking/do not move furniture

Do not share your equipment with others

If in doubt, please ask

DTU
Forebyg smitte
?

02393 Programming in C++
Module 13: Conclusion & Exam Preparation

Lecturer:

Alceste Scalas

(Slides based on previous versions by Andrea Vandin, Alberto Lluch Lafuente, Sebastian Mödersheim)

1 December 2020

Lecture Plan

#	Date	Topic	Book chapter *
1	01.09	Introduction	
2	08.09	Basic C++	1
3	15.09	Data Types	2
4	22.09	Libraries and Interfaces	3
5	29.09		
6	06.10	Classes and Objects	4.1, 4.2 and 9.1, 9.2
		Autumn break	
7	20.10	Templates	4.1, 11.1
8	27.10	LAB DAY	Old exams
9	03.11	Inheritance	14.3, 14.4, 14.5
10	10.11	Recursive Programming	5
11	17.11	Linked Lists	10.5
12	24.11	Trees	13
13	01.12	Conclusion & Exam Preparation	
	07.12	Exam	

* Recall that the book uses sometimes ad-hoc libraries that are slightly different with respect to the standard libraries (e.g., strings and vectors).

The exam

- Date & Time: **Monday 7 December 2020, 9:00**
- Place: **Online, from home**
  http://eksamensplan.dtu.dk/student
  http://eksamensplan.dtu.dk/Course
- Duration: **4 hours**
- **All aids allowed**
- Marking: **pass/fail**

The exam

A new sample exam (from 2019) is available

- **DTU Inside:** Course page → File sharing → Exam samples
- **CodeJudge:** see under “Exercises”

The solution will be available at 22:00

On DTU Inside, under: Course page → File sharing → Exam samples

Today you can use this (or other past exams) as **exam simulation**

We are available to answer your questions on any course topic

The exam

Structure of each exercise:

- A program `exZZ-main.cpp` and a header `exZZ-library.h`
- Multiple tasks (a), (b), . . .
  - ★ You are asked to implement code in `exZZ-library.cpp`
  - ★ You might be asked to also modify `exZZ-library.h`
  - ★ You get points for each task you complete, in any order
  - ★ Some tasks may depend on others
- You can use `exZZ-main.cpp` to check your solutions
- During the exam you can run tests on *CodeJudge*
  - ★ You can run tests as many times as you like
  - ★ This will not impact your grade
- More tests might be run after the exam

The exam

Some topics you can expect:

- Implement a **recursive** function
- **Iterate** over arrays/vectors/matrices/sets/. . .
- Basic use of **STL containers** (vectors, sets, maps,. . .)
- Implement (part of) a **parametric** data structure
- **Declare a class** and/or **implement its methods**
- Extend a class using **inheritance**
- Deal with **pointers**
- Some **list-like** or **tree-like** structure
- . . . (see previous exams)

The exam

Submission:

- Electronic submission through DTU Inside
  - ★ . . . or maybe DE Digital Eksamen? (you'll be notified)
- Testing through CodeJudge is **just for yourself**
  - ★ Does not count as official submission
  - ★ Does not impact your grade

Beyond this course

How to be a better (C++) programmer?

1. Practice, practice, practice!

2. Open your mind:
   - ★ Learn a new programming language/paradigm, e.g.: **Functional Programming (in F#)** (02157, 02257)
   - ★ Understand the foundations of programming languages, e.g.: **Computer Science Modelling** (02141)
   - ★ Understand how a program is compiled/interpreted/executed: **Compiler Construction** (02247)

3. Acquire programming and program analysis skills, e.g.:
   - ★ **Algorithms & Data Structures** (02105, 02110)
   - ★ **Program Analysis** (02242)
   - ★ **Model Checking** (02246)

Beyond this course

How to be a better (C++) programmer of **reliable software**?[1]

DTU's study line "**Reliable Software Systems**" has courses like:

- **Compiler Construction** (02247): covers the basics of analysis and optimisation techniques applied during compilation
- **Program Analysis** (02242) covers advanced analysis methods to spot errors and optimisations not found by compilers
- **Model Checking** (02246) focuses on errors of interacting software, e.g., to analyse that a program cannot get stuck

[1] The one that IT companies like Google, Microsoft and Intel deploy when they want to provide *rock-solid and performant* software-based products

Beyond this course

How to be a better (C++) programmer of **reliable software**?[1]

DTU's study line "**Reliable Software Systems**" has courses like:

- **Compiler Construction** (02247): covers the basics of analysis and optimisation techniques applied during compilation
- **Program Analysis** (02242) covers advanced analysis methods to spot errors and optimisations not found by compilers
- **Model Checking** (02246) focuses on errors of interacting software, e.g., to analyse that a program cannot get stuck

. . . and also **secure systems** (02244), **embedded systems** (02223), **distributed systems** (02220), **cryptographic systems** (02232), **high-performance** (02614), **data processing** (02632). . .

[1] The one that IT companies like Google, Microsoft and Intel deploy when they want to provide *rock-solid and performant* software-based products

Thanks a lot for your active participation!

Best of luck for the exam!

02393 Programming in C++

DTU

Before and after teaching:

If you feel ill, go home

Keep your distance to others, also during breaks

Disinfect table and chair

Respect the marking/do not move furniture

Do not share your equipment with others

If in doubt, please ask