DTU

Before we start:

If you feel ill, go home
Keep your distance to others
Wash or sanitize your hands
Disinfect table and chair
Respect guidelines and restrictions

02393 Programming in C++
Module 6: Classes and Objects

Lecturer:
Alceste Scalas

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

6 October 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	Summary & 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).

Recap

- Dynamic Allocation
- Containers: vectors, stacks, ...
- Strings
- File I/O

The "++" in C++

- So far: we have seen C with few elements of C++
  - ★ `string`, cin/cout, int &i, ...
- C++ extends C with two key features:
  - ★ **Object-Oriented Programming (OOP)** (today)
  - ★ **Templates** for generic programming (next lecture)

Motivation: Safe Bank Account

Live coding

OOP in C++ in a Nutshell

- A class is similar to a `struct`, but its members can be both variables and methods (a bit like functions)

OOP in C++ in a Nutshell

- A class is similar to a `struct`, but its members can be both variables and methods (a bit like functions)
- An object is an instance of a class

OOP in C++ in a Nutshell

- A class is similar to a `struct`, but its members can be both variables and methods (a bit like functions)
- An object is an instance of a class
- Class members can be public or private
  - ★ users of a class can only access public members (data encapsulation)

OOP in C++ in a Nutshell

- A class is similar to a `struct`, but its members can be both variables and methods (a bit like functions)
- An object is an instance of a class
- Class members can be public or private
  - ★ users of a class can only access public members (data encapsulation)
- Classes can have some **special methods**
  - ★ Constructor: called when an object is created (either statically, or dinamically using `new`)
  - ★ Destructor: called when an object is destroyed (either statically by exiting a scope, or dinamically using `delete`)
  - ★ Assignment: one can customise the behaviour of operator = (e.g., when the class internally uses dynamic allocation)

Abstract Data Types

- Abstract from implementation details
- Specify allowed operations on an ADT, by making them public
- Hide everything else, by making it private
- Instances of an ADT can only be constructed, accessed, and manipulated using public operations
- Programs that use the ADT do not need to be changed when the ADT's (private) implementation details are changed

Live Programming Examples

Let's implement our own `vector` class

More in the "File sharing" area of the course page (DTU Inside):

- Implementing our own `matrix` class
- Implementing our own `dictionary`/`map` class
- Implementing the "bag" (from previous exercises) in OO style

DTU

Before we leave:

Disinfect table and chair
Maintain your distance to others
Wash or sanitize your hands
Respect guidelines and restrictions outside