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

Alberto Lluch Lafuente

Sebastian Mödersheim (slides author)

March 7, 2016

Lecture Plan

#	Date	Topic	Chapter *
1	1.2	Introduction	1
2	8.2	Basic C++	1
3	15.2	Data Types	2
4	22.2		
5	29.2	Libraries and Interfaces	3
6	7.3	Classes and Objects I	4.1, 4.2 and 9.1, 9.2
7	14.3	Templates	4.1, 11.1
		Påskesferie	
8	4.4	Inheritance	14.3, 14.4, 14.5
9	11.4	Recursive Programming	5-7
10	18.4	Lists and Trees	10.5, 11, 13.1
11	25.4	Trees	13
12	2.5	Graphs	16
13	9.5	Summary	
	17.5	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

Motivation: Safe Programming with Arrays

Live programming...

The ++ in C++

- So far: basically C with few elements of C++
  - ★ `string, cout, int &i,...`
- C++ features for abstraction: ADTs and OOP
  - ★ Classes, Inheritance, Templates

OOP Basics—Summary

- A class is like a (struct) record with
  - ★ member variables and methods
- Object: instance of a class.
- Members can be public or private.
  - ★ Allows to realize ADTs: the user of a class cannot directly manipulate private members that implement the class, but only call public functions. Aka data encapsulation
  - ★ We can change the implementation without changing the calling program.
- Some special methods:
  - ★ Constructor: called when an object is created, e.g. a statically declared object or one dynamically allocated with `new`.
  - ★ Destructor: called when an object is deallocated, e.g. when the scope of a statically allocated object finishes or when a dynamically allocated object is deallocated with `new`.
  - ★ Assignment: there is an implicit assignment operator = but in some cases one needs to customize it (e.g. when the implementation uses dynamic allocation).

Abstract Data Types

- Abstract from implementation details (e.g. keyword-sorted array)
- Describe operations on ADT.
- ADTs can only be constructed, accessed, and manipulated using these operations.
- Programs that uses the ADT do not need to be changed when the ADT's implementation is changed.

Live Programming Examples

Implementing a `vector` class

Implementing a `matrix` class