

Object-Oriented Programming

The task of composition of operations is often considered the heart of the art of programming. ... However, it will become evident that the appropriate composition of data is equally fundamental and appropriate.

-- N. Wirth, **Algorithms + Data Structures = Programs, 1976.**

Object-oriented design is based on the following concepts:

Abstraction: the decision to concentrate on properties which are shared by many objects or situations in the real world, and to ignore the differences between them.

Representation: the choice of a set of symbols to stand for the abstraction.

Manipulation: the rules for transformation of the symbolic representation as a means of predicting the effect of similar manipulation in the real world.

Axiomatization: the rigorous statement of those properties which have been abstracted from the real world and which are shared by manipulation of the real world and the symbols which represent it.

-- C. A. R. Hoare, **Notes on Data Structuring, 1972.**

Today, object-oriented programming is common practice among professional software engineers, and programming languages like C++ allow the programmer to express solutions to computational problems in a variety of different ways. However, the important thing is not the language, but being able to think in an object-oriented manner, and then use the language to implement a well-engineered solution that solves the problem.

In this course, we will focus on the object-oriented structure of data and algorithms for manipulating data, and on how to write programs using C++.

8/5/00

1

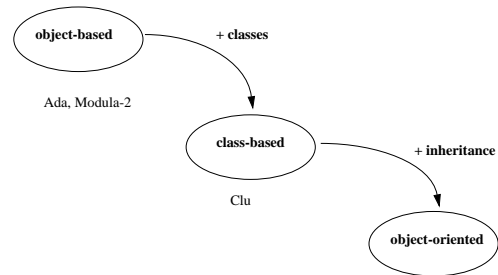
For Next Class

1. Read Appendix A and Chapter 1 in the course text.

8/5/00

3

Objects + Classes + Inheritance = OO Programs



Simula, Smalltalk, C++, Modula-3, Eiffel

Objects --- provide a conceptual framework for thinking about typed data and typed operations that correspond to a real world object that one wishes to model computationally.

The term 'object' will also be used to refer to the run-time state of a conceptual object.

Classes --- are used to declare a new type with compile-time (static) attributes of data and operations (methods) that a set of objects, or *instances* of that class, will have at run-time.

Templates -- allow compile-time (static) parameterization of a class by other types and constant values.

Inheritance --- is a composition mechanism that is used to define new classes by specializing existing classes. Inheritance is a compile-time (static) mechanism in a language like C++

8/5/00

2