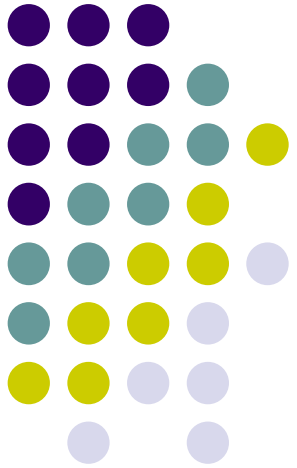
面向科学问题求解的编程实践



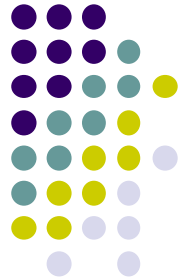
课程信息

- 面向科学问题求解的编程实践
  - 学时：30/20；周五（6，7）5201
  - 15次课
  - 上机时间：待定
  - 主讲：孙广中 gzsun@ustc.edu.cn
  - 助教：何钰、徐中天、王海林

课程信息

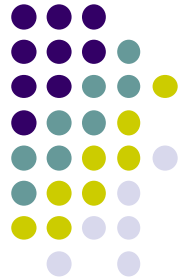
- 这是一门面向全校本科生的计算机编程实践课程。该课程面向有一定编程知识的本科生（已学过一门程序设计语言），通过有针对性的几个专题编程训练，进行问题求解的实践，加深学生对问题求解与程序设计的理解与认识，使得学生具有通过编写一定规模的程序求解问题的实践能力。作为一门以实践为主的课程，采用以课堂内讲授与课堂外实践相结合的授课形式。

课程信息



- 2020 年春季课程内容，40/40
  - 程序设计语言 (3/15)
    - Python（面向对象、软件工程）
  - 问题求解方法(4/15)
    - 枚举、模拟、分治；字符串、高精度数
  - 编程实践示例(5/15)
    - 物理随机模拟、统计数据分析、机器学习应用
  - 编程工程环境(3/15)
    - 高性能计算/并行计算/超级计算

课程信息

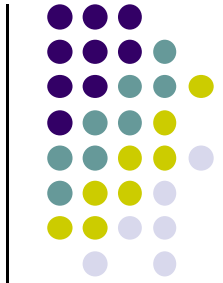


- 参考教材有两本，分别是MIT和Princeton的编程课程教材。有中文翻译版、电子版。
  - Introduction to Computation and Programming Using Python, 2nd Edition，by John V. Guttag，MIT Press，2013
  - Introduction to Programming in Python: An Interdisciplinary Approach, by Robert Sedgewick, Kevin Wayne and Robert Dondero, Addison-Wesley Professional，2015

课程信息

- 成绩评定
  - 论文或报告（个人完成）
  - 最终截止时间：预计在2021年7月
  - 过程管理
    - 2021年5月提交第一版（选题）
    - 2021年7月提交第二版（求解）

提纲



- 计算思维：原理、方法、工具
  - 平方根
  - 一道数学题
- 程序设计语言
  - 低级-高级
  - 通用-专用
  - 解释-编译
- 面向对象编程

面向对象编程

- Object Oriented Programming（OOP）
  - Smalltalk、C++、Java、Python ...

- OOP philosophy: Software is a simulation of the real world.
  - We know (approximately) how the real world works.
  - Design software to model the real world.

面向对象编程

- Procedural programming [verb-oriented]
  - Tell the computer to do this / do that.
- OOP [noun-oriented]
  - Programming paradigm based on data types.
  - Identify objects that are part of the problem domain or solution.
  - Identity: objects are distinguished from other objects (references).
  - State: objects know things (instance variables).
  - Behavior: objects do things (methods).

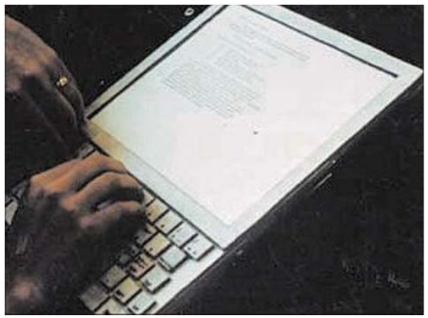
面向对象编程

"Object-oriented programming is an exceptionally bad idea which could only have originated in California."

"Computer science is no more about computers than astronomy is about telescopes."

—— Edsger Dijstra(1930—2002)

面向对象编程



"The computer revolution hasn't started yet."

"The best way to predict the future is to invent it."

"If you don't fail at least 90 percent of the time, you're not aiming high enough. "

—— Alan Kay(1940 —)

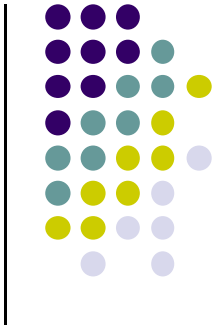
Encapsulation（封装）

- Data type

- Set of values and operations on those values.
  - Ex. int, float, char, String, Complex, Vector, Document, …

- Encapsulated data type.
  - Hide internal representation of data type.

Encapsulation（封装）



- Separate implementation from design specification.
  - Class provides data representation and code for operations.
  - Client uses data type as black box.
  - API specifies contract between client and class.

- Bottom line. You don't need to know how a data type is implemented in order to use it.

Encapsulation（封装）

Client
(needs to know how to use API)

API

Implementation
(needs to know what API to implement)

Implementation and client need to agree on API ahead of time.

Encapsulation（封装）

Client
(needs to know how to use API)

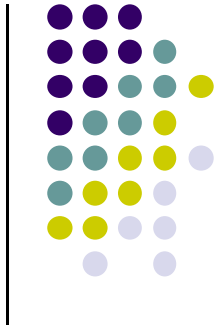
API

Implementation
(needs to know what API to implement)

Substitute better implementation

without changing the client.

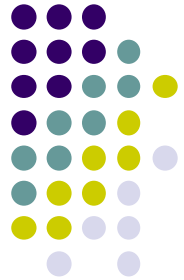
Time Bombs



- Internal representation changes.
  - [Y2K] Two digit years: January 1, 2000.
  - [Y2038] 32-bit seconds since 1970: January 19, 2038.

- Lesson.
  - By exposing data representation to client, might need to sift through millions of lines of code in client to update.

"Ask, don't touch."



- Encapsulated data types.
  - Don't touch data and do whatever you want.
  - Instead, ask object to manipulate its data.

- Lesson.
  - Limiting scope makes programs easier to maintain and understand.
  - "principle of least privilege"