Note from Introduction to Computer Science and Programming

numpy网站是：http://numpy.scipy.org

matplotlib网站是：http://matplotlib.sourceforge.net

python2.x与python3.x区别：

http://my.oschina.net/chihz/blog/123437

Lecture 1 Introduction

1The students learn best when they take notes, so this course the professor do not give handouts.

2 Think like a computer scientist

What is computation?

Declaritive (声明式) Imperative（命令式）

3 earlier computer----fixed program computer (calculator, Turing Bombe)

stored program computer

4 program is a recipe

Syntax

Static semantic: which program are meaningful

Semantics: what does the program mean; what will happen if I run it.

Style

Lecture 2 Operators and operands; statements; branching; conditionals, and iteration;

Basic Syntax

**Python**

Values:

Numbers

Strings

Operations: 2\*\*3=8

Variables

type：变量的类型会变，如果x=3之后，x=‘abc’则x的类型有整形变为字符串。

注释：#

Statement: legal commands python interpret

-----print, assignment,

（1）straight line program

（2）branching program-can change order of instructions based on a few values.

“:”冒号很重要，冒号是执行下面分支内容的提示符；

if (some test):

Block of instructions

else:

Block of instructions

(3)iteration:loop

Lecture3: Common code patterns: iterative programs

Some important type: Tuple, range

**Tuple**: ordered sequence of elements. (immutable)

ex: foo=(1,2,3,4)

Slicing: foo[1:3]=(2,3,4)

range(1,10)

=[1,2,3,4,5,6,7,8,9,10]

Lecture4: Decomposition and abstraction through functions; introduction to recursion.

Functions

Assignment

Conditionals

Looping

I/O

We don’t have decomposition and abstraction.

So, we should have Functions!

Function

Lecture 5: Floating point numbers, successive refinement, finding roots

Some care about Float

arbitrary precision:

ex: a=2\*\*10000

Long

由于float类型的特殊存法，在float类型中，一定记得“==”判断是很危险的，取而代之，我们应该用：“abs(a\*a-2.0)<epsilon”

Lecture 6: Bisection methods, Newton/Raphson, introduction to lists.

Tuples(), List[]

Non-scalar: Tuples(); Strings these two things immutable.

Mutable: Lists[](一个List里的内容可以是不同类型)

input()

append

remove

Lecture7: Lists and mutability, dictionaries, pseudocode, introduction to efficiency.

Dictionaries{}(典型应用见lecture13\_fastfib()中的应用)

Dictionaries: mutable (like list), can be heterogeneous (like list), not ordered, generalized indexing. <key value>

采用了Hashing的存法

get used to write pseudocode;

efficiency: space an time

(1)choice of algorithm

(2)map a problem into class of algorithm

Lecture8: Complexity; log, linear, quadratic, exponential algorithms

Algorithm

1. Iteration: exp, Hanoi Towers
2. big Oh
3. Divide and Conquer: Bisearch
4. link lists

Lecture9: Binary Search, bubble and selection sorts

Sorting

1. Bisearch
2. We cannot sort in a sub-linear time, if we use comperative sorting, we at least nlgn.
3. select sort
4. bubble sort

Lecture10: Divide and Conquer methods, merge sort, exception

Sorting (hashing, Exception)

1. Divide and Conquer Algorithm
2. Merge sort
3. hashing ?????
4. Exception: unhandled, handled

try-except block

ex: polymorphic

1. difference between Exception and assert: assert: pre-condition, post-condition; handled by users.

Exception: The exception handle the error himself or other levels.

**Lecture11: Testing, Debugging (Important) (38:00分钟的debug演示很好)**

1. validation: process to uncover problems and increase confidence

Testing && reasoning

1. Debugging: process to ascertaining why progress failing.

Function debugging &performance debugging (not fast enough)

1. Defensive programming: About validation &debugging(ex: using assert)
2. Test and debug is not the same thing. Test: examine input/output pairs.

Two testing: Unit testing (Functions, Classes)

Integration testing (Overall program)

Test suite?: small enough; large enough ( to boost confidence)

**A large part of being a good programmer is learning how to debug**.

Myths about bugs:

Goal: Not to eliminate 1 bug, but bug-free.

Two most famous debugging tools: prim-statement; reading.

By systematic.

Reduce search space.

Localize source of problem.

Step: (1) Study program text and ask: how it have produced this result; is it part of family; How to fix it?

(2)Scientific Method:

Studying available data.

Test Results

Program text.

Form hypothesis.

Design& run repeatable experiment

Refute hypothesis

Useful intermediate results

Expected result.

Find simplest input to show a bug.

Binary Search.

When you look for bugs: examine the following things:

1. Reversed order of arguments
2. Spelling
3. Initialization
4. Object vs Value equality
5. Aliasing-Deep and Shallow copy
6. side effects
7. Keep record of what you tried
8. Reconsider assumption
9. Debug code, not comments
10. Get help: explain somebody else what the program is doing.
11. Walk away: with a fresh eye.
12. Make sure that you can revert. Save the old versions.
13. The key point: Keep focusing on the bugs that you often make.

(**有关dynamic programming和greedy algorithm详见算法导论**)

Lecture12: More abot debugging, knapsack problem, introduction to dynamic programming

Dynamic Programing: Optimizing Probelms

(1)Two parts:

A function to maximize or minimize

A set of constraints

ex: shortest path, Traveling sales person(TSP), Bin Packing, Sequence alignment, Knapsack, Problem Reduction,

Here, we start with the Knapsack Problem.

Greedy Algorithm（贪心算法）：Wikipedia

0/1 Knapsack problems( 0-1背包问题)

**Dynamic Programming(动态规划):见Wikipedia**

（1）Overlapping sub-problems: 例如：斐波那契数列f(5)= f(4)+f(3),f(4)=f(3)+f(2),那么求f(5)和求f(4)都用到f(3),overlapping.

解决方案：Memorization. Record a value 1st time, then look it up subsequently. 见Lecture13:fastfib()函数 \_table lookup

（2）Optimal Substructure:01背包问题

Lecture 13: Dynamic Programming: overlapping subproblems, optimal substructure.

Greedy Algorithm&&Dynamic Algorithm(Code is pretty good!)

（2）Optimal Substructure

Global optimal solutions can be constructed from optimal solutions to sub-problems, this may not be true to all problems, but to a lot of them.

**Decision Trees**:

两段经典的代码：fastFib, fastMaxVal

贪心法和动态规划：

因为贪心法和动态规划都利用最优子结构性质，故人们往往会在贪心解足以解决问题的场合下，给出一个动态规划解，或者在需要动态规划方法的场合下使用贪心方法。

Lecture14: Analysis of knapsack problem, introduction to object-oriented programming

Dynamic Programming&&Python: Class

Knapsick:

Time: O(ns) n:items, s:size

Space: O(ns)

pseudo polynomial time

Dynamic Programming:

1. trading time for space
2. Don’t be intimidated by exp. problems
3. Dynamic Programming is broadly useful.
4. Problem reduction.

**Now, let we return to python.**

**Module** (modularity)

Collection of related functions

Refer to funs using **.** notation

ex:

import math

math.sqrt()

import Set

import Table

Table.member(e,t)

A particular kind of module: **Classes**.

**Object-Oriented Programming**: Java, C++, Python;

Data abstractions.

Abstract data types

Object= collection of data and functions

User defined types

**Encapsulation**.

Message passing metaphor:

Class: collection of objects with characteristics in common.

Python 语法：当请求字典对象里面没有的key时,python会抛出异常KeyError.

Lecture15: Classes: Abstract data types, classes and methods

看了这里的程序，发现python的return返回值的形式可以是任意的。

Shallow equality---‘is’(Object)

Deep equality---we can define(Value)

class—template

Python运算符重载: <http://blog.csdn.net/wklken/article/details/8126381>

<http://blog.csdn.net/adupt/article/details/4551910>

\_\_init\_\_: create an instance

\_\_str\_\_: printed representation

\_\_cmp\_\_: comparison （返回值-1，0，1）

\_\_eq\_\_: equal.

operate overloading.

use”self” to refer to that instance

dir(p): 可以使用dir(对象)来查看我可以对该对象做什么; 还可以查看模块内的可以使用的函数;

Data hiding: one can only access instance values through defined methods.

Python doesn’t do the data hiding.

Lecture16: Class: Encapsulation, inheritance, shadowing.

Inherit(继承)

supperclass –Person

subclass- MITPerson

Shadowing——Overriding(覆盖)

MITPerson的cmp覆盖父类Person的cmp

python语法：

\_\_iter\_\_(): http://luozhaoyu.iteye.com/blog/1513198

next():

StopIteration:

Lecture 17: Computational models: random walk随机游走 simulation

Dealing with and exploiting randomness

Making sense of data

Evaluating quality of answers

Start with simple model.

Simulate random walk.

Python 语法：类中也允许default values

Lecture 18: Presenting simulation results, Pylab, plotting

Simulation:

(1)Inner loop that simulates 1 trial.

(2) “Enclose” inner loop in a loop that conducts appropriate number of trials.

(3)Calculate & present statistics.

Lecture 19: Biased random walks, distributions

Lecture 20: Monte Carlo simulations, estimating pi

Lecture 21: Validating simulation results, curve fitting, linear regression

Lecture 22: Normal, uniform, and exponential distributions; misuse of statistics

Lecture 23: Stock market simulation

Lecture 24: Course overview; what do computer scientists do?