程序初步-I

- Python 语言简介
- Python 程序设计入门
  - 数据，数值表达式及求值，数据对象和类型
  - 内置函数和数学函数包
  - 字符串及其基本操作
  - 标识符、变量和赋值
  - 简单脚本程序
  - 判断和条件控制，重复计算和循环
  - 计算的抽象：函数

Python 语言介绍

- 由荷兰国家数学和计算机中心程序员 **Guido van Rossum 1989** 年开始开发，并于 **1991** 年 **2** 月发布
  - **The name is a tribute to the comedy group "Monty Python", but the snake with the same name is also the symbol of the language.**
- 目前由非营利性国际组织 **Python Software Foundation (PSF)** 主导开发和管理
- 是一种面向对象 **(object-oriented programming)** 的、动态类型 **(dynamic types)** 的脚本语言 **(script language)**
  - **Python** 系统 **(**如 **PSF** 主导开发的 **CPython)** 的主要部分即是一个解释器
  - **(Note：Python** 程序并不是简单的解释执行**)**

Python 的基本特点

- 支持交互式执行以及模块开发，同时也能支持大规模软件开发
  - 是通用的高级编程语言
- 跨平台 **(cross-platform)**
- 易扩充 (胶水语言，**glue language)**
- 基本语言已提供了丰富的功能，并附带了大规模的标准库
- 具有非常丰富的、各个应用领域的工具包：
  - 科学计算与可视化：**NumPy, SciPy, pandas, matplotlib…**
  - 计算机视觉和图像处理：**PIL (Python Image Library), Pillow, OpenCV, VTK, ITK …**
  - **GUI** 图形用户界面：**WxPython, PyQt, PyGTK…**
  - **……**

集成开发环境 (IDE)

- 实际上，可以用任何文本编辑器 (比如记事本) 来编写 Python 程序；但使用某种 IDE 更方便
  - IDLE 是 PSF Python 系统自带的 IDE
- 集成开发环境 (Integrated Development Environment, IDE)
  - 一种提供程序开发环境的应用程序 (支持一种或者多种编程语言)
  - 其中一般包括代码编辑器、编译器、调试器和图形用户界面等工具
  - 集成了代码编写、分析、编译、调试、执行等多种功能
    - 一般支持语法高亮、自动缩进、代码跳转、智能提示、自动补全、编码转换、单元测试、本地执行、项目管理等

Python 的基本编程元素

- 数据
  - 数：整数、浮点数、复数
  - 字符串
  - 其他 (列表、元组、字典……)
- 表达式 (最简单的 Python 程序结构)：描述基本计算
  - 被 Python 解释器求值后得到计算结果
  - 在交互式执行中将自动显示结果
- 语句：描述操作
  - 不能被求值
- 控制和抽象

基本编程元素：数

- Python 有三类数
  - 整数，模拟数学里的整数
  - 浮点数，模拟数学里的实数
  - 复数，模拟数学里的复数
- 与对应的数学概念不同
  - 能表达的数值是有穷的
  - 浮点数和复数的精度有限
- 每一类数有规定的
  - 书写形式 (语法)
  - 表达的意思 (语义)

整数、浮点数

- 整数
  - 可以写多种进制的整数
  - 可以写任意大 (长) 的整数 (计算机的存储器有穷，只能表示有穷大小的整数)

进制	前缀	例子
十进制 decimal	无	367
二进制 binary	0b、0B	0b101101111
八进制 octal	0o、0O	0o557
十六进制 hexadecimal	0x、0X	0x16f

- 浮点数
  - 语法：连续数字序列，必须 — 或有小数点；或有表示指数的 e (或 E)， 后跟正负号 (可选) 和一个整数
  - 语义：表示相应的浮点数 (指数部分以 10 为底)
  - 性质：表示范围和精度均有限
- 语法错误 (SyntaxError)，例如： 3a 3e--3 3+

对象和类型

- Python 把程序运行中存在的任一实体 (即，使用和处理的各种元素) 统称为“对象” (object)
- 程序中，每一个对象都有确定的类型 (type) 和值 (value)
  - 类型：性质相同的一类对象所构成的集合
    - 具有相同的写法、属性和使用方法 (如行为、存储方式、计算规则等)
  - 每个类型都有名字 —— 类型名
    - 整数对象的类型名是 int，浮点数的类型名是 float
- type(...) 得到对象 … 的类型

  例：type(0.467)
- 类型(名) 的一个用途是做强制类型转换

  例：int(-2.3)　浮点数转换到整数的规则是 (趋零) 取整

算术表达式

- 算术运算符

+	-	*	/	//	%	**
加/正号	减/负号	乘	除	整除	求余数	乘幂

  - /：’真’除法，结果为浮点数
  - //： floor division operator (整除、地板除)
  - 余数：与除数同号
  - Python 系统保证：被除数 = 除数 * 整除商 + 余数

- 算术表达式
  - 语法：基于运算对象、运算符和括号 ()，符合运算符元数要求 (可以严格描述，参看语言手册)
  - 语义：按表达式描述的计算，算出的值

算术表达式描述的计算

- 合法的表达式
  - 描述了一个计算过程
  - 解释器计算 (求值) 表达式，将得到一个值
  - 注意：合法表达式在计算中也可能出错
- Python 严格规定运算符的优先级和结合方式等
  - 优先级：+ -，* / // %， 一元运算符，**
  - 结合方式
    - 一元运算符和 ** 从右到左，其余算术运算符从左到右
  - 括号 () 用于描述特定的计算顺序
  - 二元运算符，先计算左边的运算对象，再计算右边的

表达式计算和类型

- 表达式描述的计算与类型有关
  - 参与计算的对象有类型，计算结果也有类型；类型将影响计算的方式和结果
  - 整数计算是“精确的”，总得到精确的整数结果 (/ 除外)
  - 浮点数计算是近似计算，得到有限精度的近似结果
    - 溢出问题 (动态错误)
- 混合类型计算和类型转换
  - 不同类型混合计算时，先将两者 (自动) 转换到统一类型
  - 整数和浮点数运算时，先将整数转换到浮点数 (转换可能出错：整数可以任意的大，可能超出浮点数类型能表示的范围)
- Python 程序中的表达式用一维的字符序列表示
  - 原则上，要求一个表达式 (结构) 应该写在一行里 (程序行是 Python 的代码单位)；但存在续航规则，并允许反复使用

复数 (complex)

- 复数字面量 (语法)

  虚部用 (整数/浮点数) 加后缀 j 或 J 表示，不能空格

  如 2.0 + 3.2j，2 + 3j

- 或者，用 complex(m), complex(m, n) 构造 (complex 是复数类型名)

  m 表示实部，n 表示虚部

- 复数对象的实部和虚部都是浮点数

  2 + 3j 等同于 2.0 + 3.0j

- 混合类型计算

  如果有复数对象参与计算，整数和实数先转换到复数

```
>>> complex(2)
(2+0j)
>>> complex(2, -4)
(2-4j)
```

```
>>> 2.4 * (-3.1 + 4.2j)
(-7.439999999999995+10.08j)
```

```
>>> (3 + 5j) // 2
Traceback (most recent call last):
  File "<pyshell#104>", line 1, in <module>
    (3 + 5j) // 2
TypeError: can't take floor of complex number.
```

```
>>> (3 + 5j) % 2
Traceback (most recent call last):
  File "<pyshell#106>", line 1, in <module>
    (3 + 5j) % 2
TypeError: can't mod complex numbers.
```

```
>>> float(2.4 + 1.4j)
Traceback (most recent call last):
  File "<pyshell#111>", line 1, in <module>
    float(2.4 + 1.4j)
TypeError: can't convert complex to float
```

函数及其使用

- 函数：通过调用被执行的程序对象，实现特定的计算功能
  - 函数有名字 (即函数名)，可用在表达式或其他结构中
- 使用函数的基本结构：函数调用表达式
  - 语法形式

函数名(参数, ...)

其中参数可以是任意合适的表达式，称为实际参数 (argument, 简称实参)，多个实参用逗号分割

例：type(1.4)，int(2.4**1.2), complex(2.5, -3.2) 等

  - 语义：基于给定的实参 (也可能没有实参)，执行函数名所指定的函数对象，得到函数计算出的结果值 (函数返回值)

实参应满足函数需要 (如数目，类型等)

内置函数

- **Python 提供一组内置函数 (或称标准函数)，详见 The Python Standard Library — 2. Built-in Functions**

		Built-in Functions		
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

简单脚本程序的执行和内置输出函数 print

- 简单 Python 程序通常就是一系列语句：表达式 (语句)，import 语句，赋值语句 (之后介绍) ……
- Python 脚本程序：独立的、扩展名为 py 的文件
  - 脚本的内容是一系列语句
  - 执行一个脚本程序时，解释器将重新启动，重置执行环境
  - 执行过程中，表达式的值不会在执行窗口 (shell) 里显示；查看计算结果需要在脚本中调用输出函数
- 内置函数 print 被调用执行将产生输出，显示到执行窗口中

  print(表达式, 表达式, ……)
  - 语义：逐个求值多个实参表达式并输出；每两项输出之间用空格分隔，所有输出结束后换一行 (默认情况)
  - print(…) 是函数调用表达式，其值是一个特殊对象，用关键字 None 表示，不显示

内置函数 — 数值计算

- **abs**：求整数、浮点数的绝对值，复数的模
  - 实参为整数时，返回值为整数；其它情况，返回值为浮点数
- **max, min**：求实参中的最大值和最小值，允许任意多个实参 (至少两个)
  - 其它使用情况见手册
- **round**：求浮点数的近似值，第二个整数参数 (可缺) 说明保留的小数位数 (舍入规则详见语言手册)
- **pow**：计算乘幂结果
  - **pow(x, y) → x**y**
  - **pow(x, y, z) → x**y % z**

数学函数 (导入函数库)

- Python 标准库模块 math 提供了各种有用的数学函数，但不是默认/内置功能，使用前需要导入 math 模块/包
- 导入模块的三种方式 (import 语句)
  1. from math import *

     语义：导入 math 包的所有功能，并可直接可用
  2. import math

     语义：导入整个 math 包，以 math.sin, math.cos 形式使用
  3. from math import sin, cos, sqrt

     语义：导入 math 库函数 sin, cos 和 sqrt，并可直接可用
- 查阅 The Python Standard Library 9.2 节了解 math 库提供的数学函数 (语法和语义)
  - import math 后，执行 dir(math) 列出 math 模块内容

记录中间结果，变量

- 实际需求：计算中经常需要记录计算得到的中间结果
- 例：已知三角形三边长分别为 6, 8, 11，求面积

  面积公式：$$a = \sqrt{s(s-x)(s-y)(s-z)}$$
  $$\text{where } s = (x+y+z)/2$$

  - 其中使用 s 共 4 次，应该把首次算出的 s 值记录下来
- 变量：Python 里用于记录信息的机制
  - 语法：变量用名字 — 标识符 — 表示 (可以记录值，即约束于某个对象)
  - 语法：变量是一种基本表达式 (与整数字面量等类似)；把变量写在表达式里，就是要求使用它的值

标识符、关键字

- 合法标识符 (identifier) 的语法要求：

  常用形式：字母开头的字母数字序列，下划线当作字母

  区分大小写：如 OK, Ok, ok, oK 是不同的名字

  允许用一般的 (非英文) unicode 字符序列 (但并不建议)

- 关键字 (keywords / reserved words)
  - Python 中一组特殊的名字，由语言规定

False	await	else	import	pass
None	break	except	in	raise
True	class	finally	is	return
and	continue	for	lambda	try
as	def	from	nonlocal	while
assert	del	global	not	with
async	elif	if	or	yield

标识符、关键字

- 合法标识符 (identifier) 的语法要求：

  常用形式：字母开头的字母数字序列，下划线当作字母

  区分大小写；如 OK, Ok, ok, oK 是不同的名字

  允许用一般的 (非英文) unicode 字符序列 (但并不建议)

- 关键字 (keywords / reserved words)
  - Python 中一组特殊的名字，由语言规定
  - 每个关键字有其特殊意义，只能用在特定地方
  - 特别注意：关键字不能用作变量名
  - 类型名、函数名、程序包名等是普通标识符，不是关键字

  Python IDE 通常自动识别关键字，以特殊颜色/形式显示

变量

- 变量的命名
  - 取名即在定义/引入概念，是一种最基本的抽象技术
  - 建议：采用有意义的名字 (单词或词组)；变量名一般以 (小写) 字母开头
- 变量赋值：用变量记录计算的结果 (给计算出的对象“取名”/ 建立变量和对象之间的约束关系)
- 赋值是 Python 的基本操作，通过赋值语句实现
  - 语法：

    变量 = 表达式

    例：temperature = 98.6

变量

- 变量的命名
  - 取名即在定义/引入概念，是一种最基本的抽象技术
  - 建议：采用有意义的名字 (单词或词组)；变量名一般以 (小写) 字母开头
- 变量赋值：用变量记录计算的结果 (给计算出的对象“取名”/ 建立变量和对象之间的约束关系)
- 赋值是 Python 的基本操作，通过赋值语句实现
  - 语法：

    变量 = 表达式

    变量, ... = 表达式, ...　　(要求变量和表达式一一对应)
  - 语义：顺序求出各表达式的值，然后赋给对应的变量

变量和赋值

- 赋值是语句 (不是表达式)
  - 执行时，产生效果 (但不会得到值)
- Python 允许：在程序中随时引入新变量 (不需要声明)
  - 赋值即定义规则：给之前未定义的变量赋值，则该变量即被定义 (且建立约束/引用关系)
- 写在表达式里的变量，表示使用它关联的值 (即，变量求值)
  - 不能对没赋过值的变量求值，将报错 (NameError)
  - 如果变量已经有值，再次赋值将改变它的值 (即改变约束关系)
  - 变量求值总得到此前最近一次赋给它的值
- 例：求边长 5, 7, 11 的三角形的面积

字符串

- 字符串：字符的序列，是一种基本组合数据类型 (字符串类型)
  - 类型名：**str**
  - 字面量：一对单引号或一对双引号括起的字符序列，其中可以包括空格和一些特殊字符 (称为转义序列，见下表)

\'	\"	\n	\t	\\
单引号	双引号	换行符	制表符	反斜线符

  - 语义：相应的字符串对象 (空格等空白字符也是串的内容)
  - 该形式的字符串字面量只能写在一行里，不能物理换行
    - 如一个字符串结构不完整时换行，将报语法错
    - 可用续行符写长字符串

字符串基本操作 (1)

- 设 s, t 的值是字符串，i 是整数
  - **len(s)** 得到 s 的长度 (s 中字符的个数)
  - **s[i]** 取得 s 第 i 个字符，首字符 s[0]，最后字符 s[-1]
    - 合法下标范围：**0 ~ len(s)-1** 或 **–len(s) ~ -1**；超范围访问将报下标越界错误
    - 结果为包含单个字符的字符串
  - **s + t** 得到两个串的拼接串，如 "ab" + "12" 是 "ab12 "
  - **s * i** 或者 **i * s** 得到串 s 的 i 个拷贝的拼接
- 数值与字符串的转换：通过强制类型转换可以将各种数值类型的对象转换成字符串，也可以将内容合适的串转换成 (构造出) 一个数值对象

字符串基本操作 (2)

- 字符串切片：取出已有字符串的一部分，做成一个新串；假设 s 是一个字符串
  - **s[m:n]**：得到由字符 **s[m]** 到 **s[n-1]** 构成的字符串；如果 **m≥n**，得到一个空串，其中不包含任何字符
  - **s[m:n:d]**：得到由 **s[m]** 到 **s[n-1]**，下标的步进值为 **d**选出字符做成的字符串；如果 **d** 是正数且 **m≥n**，或者 **d**是负数且 **n≥m**，则得到空串

说明：下标范围采用左闭右开区间；**m**、**n**、**d**都是值为整数的表达式，**m** 或 **n** 的值为 **-1** 表示串 **s** 的末字符，其余负数类推；描述中冒号不能省略，但 **m**、**n**、**d**都可以省略： **d** 省略时表示 **1**；如果 **d > 0**，**m** 省略时表示 **0**，**n** 省略时表示 **len(s)**；如果 **d < 0**，**m** 省略时表示 **-1**，**n** 省略时表示 **-len(s)-1**

输入

- 内置 (行式输入) 函数 input
  - 调用形式：input(提示字符串)
  - 语义：
    1. 在执行环境中，输出“提示字符串” (也可缺)
    2. 等待用户输入 (直到用户输入信息并回车)
    3. 返回用户在 Enter 键之前的一行输入 (做成的一个字符串对象)
- 例：输入一个整数

```
n = int(input(“Input an integer: ”))  # 给出适当的提示串
```

输入一个浮点数

```
x = float(input(“…..:”))   # 使用输入数据前，一般需要做类型转换
```

- 实例：输入三角形三边长，计算并输出面积

组合数据 (初步)

- 程序中可定义多个变量保存多项数据，但无法满足某些实际需求
  - 需要保存的数据值数量不定
  - 需要写循环以统一方式逐个处理多项数据
  - 需要把一组不定数量的值送进函数或作为函数返回值
  - ……
- 程序中需要组合数据对象，用来包装一批 (更基本的) 元素，且
  - 能访问和使用其中的元素
  - 能个别地处理、或以统一的方式处理其成分元素
  - 能 (像简单对象一样) 整体赋给变量，传进传出函数

组合数据对象之表 list (初步)

- 表/列表：类型名 **list**，Python 内置的、最常用的组合数据类型
  - 表对象中的成分元素线性排列，每个元素有确定的排列位置 / 编号 — 下 标，首元素的下标为 0

  ➔ 一个表对象即是一系列表元素的有序组合

- 表的列举式 (display, 类似于字面量的表达式)

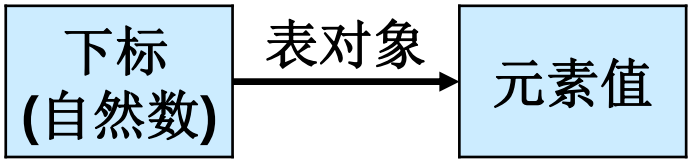
  [表达式, …]

  - 其中可以写任意多个元素表达式，之间用逗号分隔
  - 元素可以写任意表达式，建立表时以表达式的值作为元素
  - 没有元素的表称为空表，用 [ ] 表示

  (适用于表元素很少，或者各元素间没有统一的描述方式)

表的基本使用 (初步)

- 表对象的最基本操作：元素取值和赋值
  - 表元素用下标表达式描述
  - 表 lst 的元素下标从 0 ~ len(lst)–1 (或 -len(lst) ~ -1)

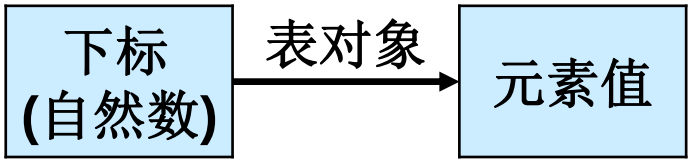


  - 描述下标的表达式中可以包含变量，实际所访问的表元素由该表达式的值确定
- 表对象支持切片、拼接等操作
  - **lst[m:n], lst[m:n:d]：下标范围为左闭右开区间；d 被省略时表示取值步长为 1；d > 0 为正向取值，m 省略时表示 0，n 省略时表示 len(s)；d < 0 为反向取值，m 省略时表示 -1，n 省略时表示 -len(s)-1**

表的基本使用 (初步)

表对象是可变对象，元素值及其结构都可以改变！

- 表对象的最基本操作：元素取值和赋值
  - 表元素用下标表达式描述
  - 表 lst 的元素下标从 0 ~ len(lst)–1 (或 -len(lst) ~ -1)



  - 描述下标的表达式中可以包含变量，实际所访问的表元素由该表达式的值确定
- 表对象支持切片、拼接等操作
- 一个表对象是一个整体
  - 可以作为变量的值，可以作为函数的参数和返回值
  - 程序中可以通过变量和函数实现对表对象的操作