Python
基本的数据类型和数据结构

刘理
厦门大学2016级数学科学学院计算数学系

首先，我希望你已经安装好并且运行好了Python。如果没有安装好，请咨询上一个授课的成员。

另外，本节开始我们需要多次的在命令行和解释器中运行代码。如果需要自己操作，最好不要直接复制粘贴，采用手打的方法录入。请不要漏掉任何一个空格或者标点符号

01 基本的数据类型

02 基本的数据结构

03 如何规范输出

04 *一些有趣而有用的骚操作

Chapter 1
基本的数据类型

为什么需要数据类型？

这是一个很长的故事...

数字

- 整型用来表示一个**整数**（默认为十进制数）。
- 浮点型用来表示一个**分数**（小数）。
- 整型与整型之间可以进行运算且Python**在足够大的位数情况下是精确的**，浮点型与浮点型之间虽然可以进行运算，但是因为**系统有效数字位数的限制，经常会产生误差。**
- 整数与浮点数之间运算时，整数会被转换为浮点数。

补充：常量，变量，变量名

$$a = 2$$

Python运行示例:数字运算

```
a = 2
b = 3
print(a+b)

a = 5.5
b = 7.8
print(a+b)
```

- 如果你是在命令行运行，那么连print都可以省了

某种意义上来说，Python就是一个计算器

字符与字符串

- 一个字母(A-Z,a-z)，一个数字(0-9)，一个标点符号，一个空格，这些都是一个字符，**使用符号' '组合起来**。比如说a, 3等。

- 多个字符连在一起就是一个字符串，**使用符号”　”组合起来**。比如”I'am”表示的字符串是I'am。”　”不会被算入字符串中（**怎么知道呢？**）

Python运行示例:字符与字符串输出

```
a = 'f'
print(a)

b = 'g'
print(b)

c = “I'am a student from XMU”
print(c)
print(len(c))
```

如何使用print输出”I'am a student”?

- 如果你是在命令行下，也可以不使用print，看看结果有什么区别

*转义字符\

- 用于防止字符与规范符之间的冲突。
- 用于输出一些特殊的字符（比如换行）

Python运行示例:转义字符

```
a = “\”I'am a student\””
print(a)

b = “Hello \n world”
print(b)
```

到现在为止，汉子们已经可以使用Hello world和妹子装逼了\滑稽

Chapter 2
基本的数据结构

为什么又要有数据结构？

其实，一切来源于生活...

列表——list

- **可变有序**列表，用[]括起来，比如[1, 2, 3]就是一个列表，里面有三个元素1, 2, 3。

- 可以随意的**添加，删除**和**改变**元素，也可以将元素的元素取出来（**索引**）。

*Python运行示例:list列表操作

```
a = [1, 2, 3]
a.append(4)
a.append("XMU")
print(a)

a = [1, 2, 3]
a.insert(1, "XMU")
print(a)
a.pop()
print(a)
a.pop(1)
print(a)

a = [1, 2, 3]
print(a[1])
print(a[-1])
a[1] = "XMU"
print(a)
```

我承认刚开始说这么多确实有点难了，但是很遗憾的告诉你们，这里所有的操作都是常规操作。如果你以后会跟Python经常打交道，熟记这些操作是必须的。

列表——tuple

- **不可变有序**列表，用()括起来，比如(1, 2, 3)就是一个列表，里面有三个元素1, 2, 3。
- **不可以**随意的添加，删除和改变元素，但可以进行索引
- 索引的操作和list完全相同。

tuple和list相比有什么优势呢？

Python运行示例:tuple列表操作

```
a = (1, 2, 3)
print(a)
print(a[1])
print(a[-1])
a[1] = 4 # Error line

a = (1,)
print(a)
len(a)
```

字符串(str)

- 如果把字符串的每个字符当作一个列表中的元素，那么字符串就是一种特殊的列表。但是请注意它是**不可变的**。

补充：Python运行示例:函数len()

- 计算列表和字符串的**元素个数。**

```
a = “XMU”
print(len(a))
a = [1, 2, 3]
print(len(a))
a = (1,)
print(len(a))
a = “Hello world”
print(len(a))
```

字典(dict)

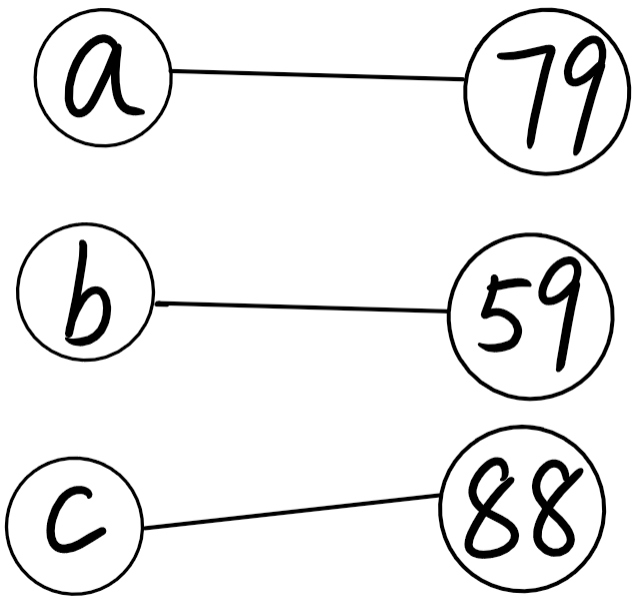
它与列表存储数据的区别在哪里？优劣势呢？

- 用来表示元素与元素之间的对应关系，用{}括起来，可变。

- 举例：

```
a = {'a':79, 'b':59, 'c':88}
```

- 使用key-value的表示方法存储数据。**不可以**使用列表的常规索引方法。



Python运行示例:字典(dict)

```
a = {“xmu”:40, 23:30, 'd':50}
print(a)
print(a[“xmu”])

a[40] = 234
print(a)
a[“xmu”] = 60
print(a)

print(len(a))
print(list(c.keys()))
print(list(c.values()))
```

Chapter 3
*如何规范输出

print格式化输出

- 在之前，我们已经接触了很多print的例子。可以看出，print只可以输出变量的内容，也可以直接指定想输出的字符串内容。

- 如果要求字符串里“嵌入”变量的内容，就需要使用格式化输出。

Python运行示例:格式化输出

```
a = 2
print(“I want %d apples”) % a
b = 3
print(“He wants %d apples”) % b
print(“I want %d apples but he
wants %d apples.”) % (a,b)

a = 'a'
print(“%c”) % a

b = “XMU”
print(“%s”) % b

c = 23.857
print(“%10.3f”) % c
```

在自己操作的时候，请永远不要忘记对照，任何一个字符的不同都有可能导致代码报错。

Chapter 4
*一些有趣而有用的骚操作

Python运行示例:拼接字符串

```
a = 'I love '
b = 'XMU'
print(a+b)
print('{}{}'.format(a,b))
print(a, b, sep='')

data = ['ACME', 50, 91.1]
print(','.join(str(d) for d in
data))
```

对于任何一个新手来说，这些操作都有些难以理解。但是如果你需要运用Python，那么这只是基本功。

Python运行示例:替换字符串

```
a = 'XMU'
print(a.strip('X'))
print(a.strip('XU'))

b = '====XMU----'
print(b.replace('=', '-'))
```

更加高级的操作需要涉及到正则表达式。为了不让大家感到难受，我们这里就不介绍了。

补充：新手编程的注意事项

官方文档！	注意，你一定会不时的遇到你没学过的新命令，请多查查OD
注意规范	多检查你的代码，比如Python缩进错误是不会通过编译的，不要随便动它
多练	坑是需要踩的，不犯错误怎么熟悉诡异的报错
多问	可以对报错多多询问，但是请一定要提供完整代码和报错信息
多找	没有人的时候，报错信息Google一下？
多看书	现在Python的书和网络教程都一大堆了，你还一本没看过？

THANK YOU