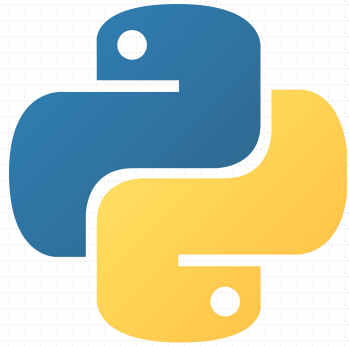
人工智能实践教程
从Python入门到机器学习



第一部分Python核心编程技术

- **所有代码及ppt均可以由以下链接下载**
- **https://github.com/shao1chuan/pythonbook**
- **https://gitee.com/shao1chuan/pythonbook**

目录

CONTENTS

01 Python简介

•Python 是一门优雅而健壮的编程语言,它继承了传统编译语言的强大性和通用性,同时也借鉴了简单脚本和解释语言的易用性。

•Python的作者是著名的“龟叔”Guido van Rossum，1989年，龟叔为了打发无聊的圣诞节，开始编写Python语言。1991年，第一个Python编译器诞生。它是用C语言实现的，并能够调用C语言的库文件。

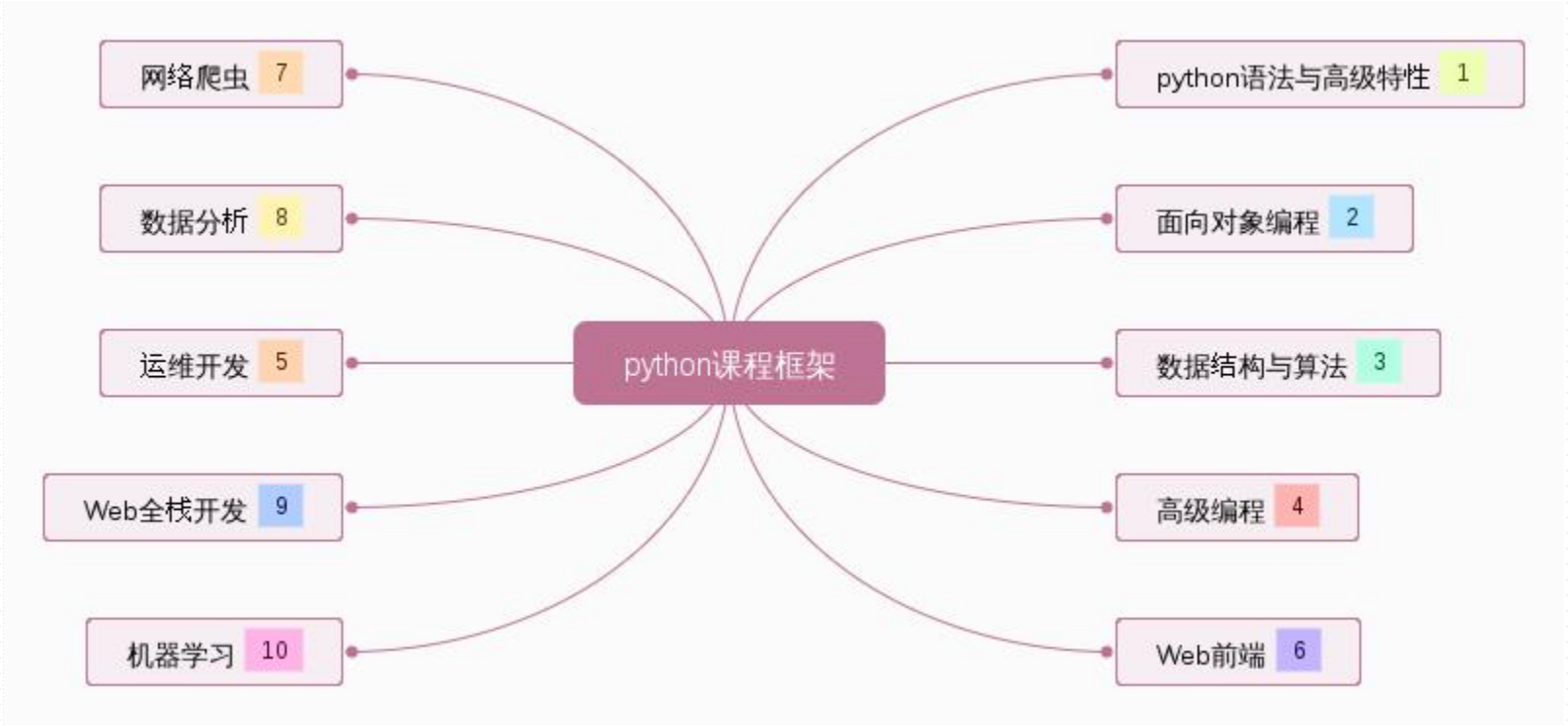
为什么是蟒蛇?

01 Python简介

Guido

1）官 网：
https://www.python.org/ 2）
中文社区：
http://www.pythontab.com/

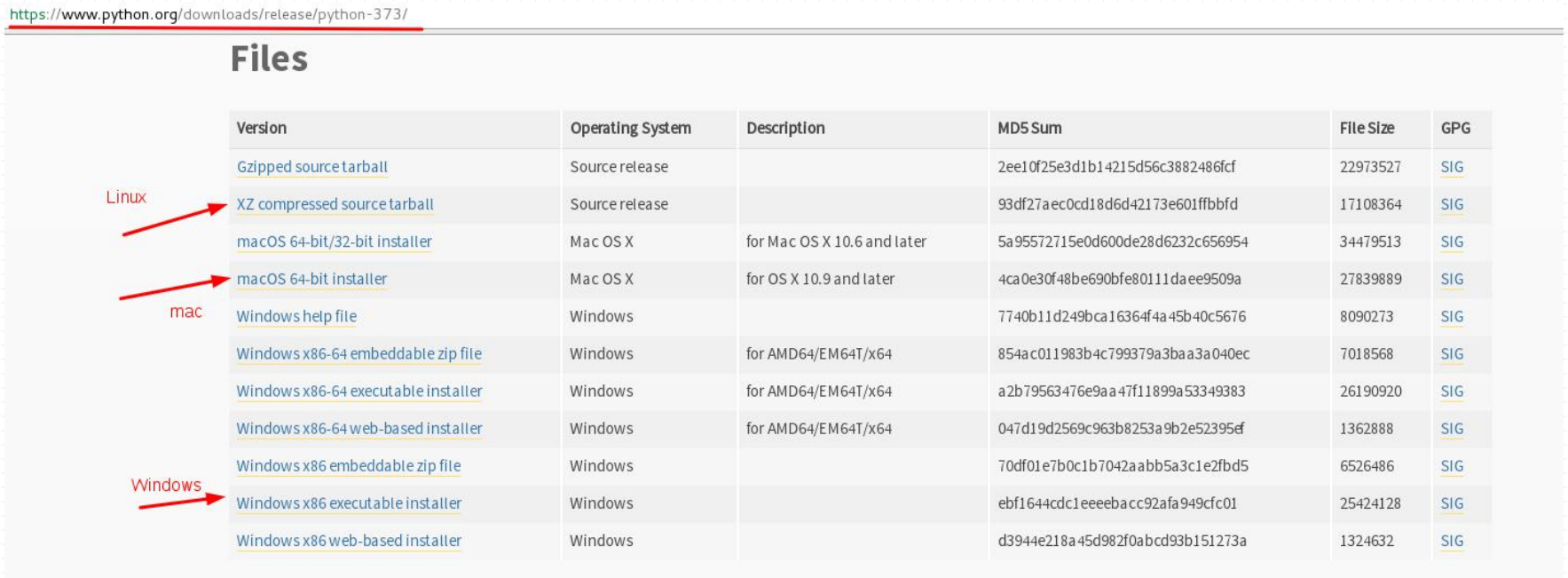
01 Python简介


python课程框架
python语法与高级特性 1
面向对象编程 2
数据结构与算法 3
高级编程 4
Web前端 6
网络爬虫 7
数据分析 8
Web全栈开发 9
机器学习 10
运维开发 5

02 Python开发环境搭建

Downloads

官方网站: http://python.org



02 Python开发环境搭建 Linux源码安装python3

1). 解压安装包到指定目录

2). 安装编译过程中需要的依赖包(gcc， zlib, zlib-devel, openssl-devel)

3). 进入解压的安装包进行编译

```
./configure --prefix=/usr/local/python --with-ssl
```

4). 安装

```
make && make install
```

5). 添加py3命令到环境变量PATH

临时添加: export PATH="py3命令所在的路径:$PATH"

永久添加:

```
echo export PATH="py3命令所在的路径:$PATH" >> ~/.bashrc
source ~/.bashrc
```

02 Python开发环境搭建 python代码编写

Anaconda是一个开源的包、环境管理器，可以用于在同一个机器上安装不同版本的软件包及其依赖，并能够在不同的环境之间切换.

1). 交互环境:

```
[kiosk@foundation0 ~]$ python
Python 2.7.5 (default, Aug  2 2016, 04:20:16)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-4)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print "hello python"
hello python
>>> exit()
```

2). 文本环境:

```
[kiosk@foundation0 ~]$ vim hello.py
[kiosk@foundation0 ~]$ chmod +x hello.py
[kiosk@foundation0 ~]$ ./hello.py
hello world
[kiosk@foundation0 ~]$ cat hello.py
#!/usr/bin/python
#encoding:utf-8
print "hello world"
```

02 Python开发环境搭建 python代码编写

1). 指定python解释器

#!/usr/bin/python

#!/usr/bin/env python

2). 字符编码:

#encoding:utf-8

#coding:utf-8

ASCII只能识别英文

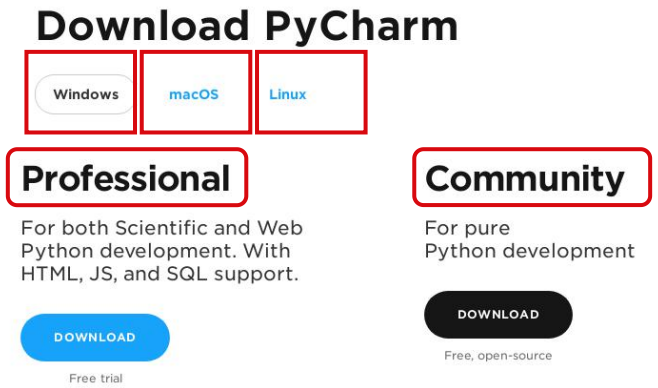
UTF-8 是「编码规则」, 可以识别中文和英文.

02 Python开发环境搭建PyCharm 集成化管理工具

PyCharm是一种Python IDE，带有一整套可以帮助用户在使用Python语言开发时提高其效率的工具，比如*调试、语法高亮、Project管理、代码跳转、智能提示、自动完成、单元测试、版本控制。*此外，该IDE提供了一些高级功能，以用于支持Django框架下的专业Web开发。

下载地址: https://www.jetbrains.com/pycharm/download/

02 Python开发环境搭建PyCharm 集成化管理工具



键位	功能
ctrl + shift + f10	运行脚本
ctrl + /	注释行
ctrl + p	查看函数参数
双击 shift	全局查找
Ctrl + Space	基本的代码完成
Shift + F9	调试
Shift + Enter	另起一行

1). 免费激活码: http://github.com/shao1chuan/soft

2). 下载安装

3). 新建项目

4). 配置 Pycharm

5). 快捷键设置(更多快捷键)

02 Python开发环境搭建动手练习

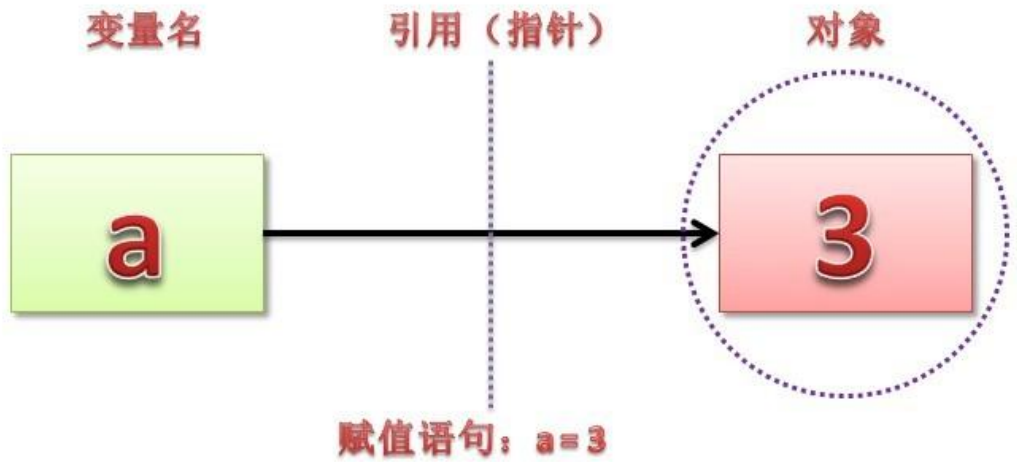
1. 安装 Python3.8解释器
2. Pycharm 专业版IDE工具的安装与自定义配置
3. 编写脚本。录入系统, 录入用户的名字、年龄、最喜欢的颜色和与你相关的一些事情(背景、兴趣、爱好等等)。

03 Python 运算符

变量

- 变量是内存中的一块区域。对象赋值实际上是对象的引用。a=10
- 变量的命名: 变量名由字母，数字，下划线组成， 不能以数字开头. (a,b, c)

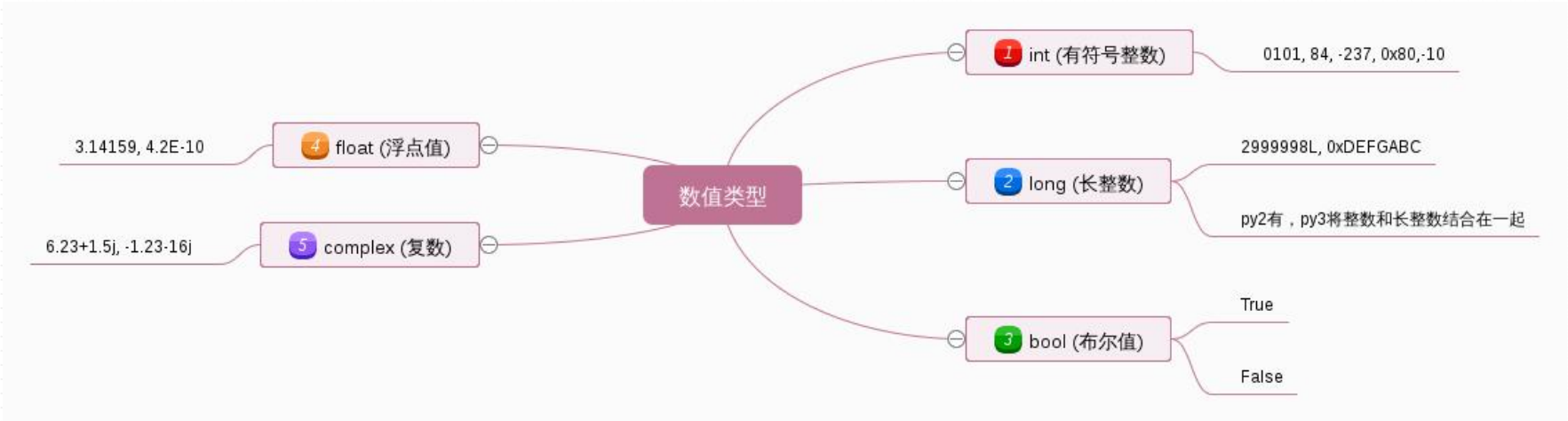
  hello = 100, hello_world = 100, count2 =100, 1count=10
- Python中，变量定义时不需要指定类型的，当用变量的时候，必须要给这个变量赋值;



03 Python 运算符

基本数据类型-数值类型

Python 支持五种基本数字类型:



注意:

1). 整数一般以十进制表示,但是 Python也支持八进制(**“0”**开始)或十六进制(**“0x” 或 “0X”** 开始)来表示整数。

2). 整数的范围取决于机器是32位还是64位? 但长整数不是， 取决于虚拟内存的大小.

03 Python 运算符

运算符

1). 算术运算符：+，-，*，**，/，%，//

2). 赋值运算符：=, +=, -=, /=, *=, %=

3). 关系运算符: >, >=, <, <=, !=, ==

4). 逻辑运算符:逻辑与and， 逻辑或or， 逻辑非not

```
>>> print 5/2
2
>>> print 5.0/2
2.5
>>> print 5//2
2
>>> print 5.0//2
2.0
>>> print 2**3
8
```

```
>>> a = 1; a += 3
>>> print a
4
>>> a =1; a -= 1
>>> print a
0
>>> a=1; a *= 8
>>> print a
8
>>> a = 10; a /= 2
>>> print a
5
```

注意: =和==的区别?

```
>>> 1 > 2
False
>>> 1 < 2
True
>>> 1 != 2
True
>>> 1 == 2
False
>>> 1 >= 2
False
>>> 1 <= 2
True
```

```
>>> 1 > 2
False
>>> 1 < 2
True
>>> 1 != 2
True
>>> 1 == 2
False
>>> 1 >= 2
False
>>> 1 <= 2
True
```

表 3-14 运算符的优先级

优先级	运算符	类	结合性
1	()	括号运算符	由左至右
1	[]	方括号运算符	由左至右
2	!、+（正号）、-（负号）	一元运算符	由右至左
2	~	位逻辑运算符	由右至左
2	++、--	递增与递减运算符	由右至左
3	*、/、%	算术运算符	由左至右
4	+、-	算术运算符	由左至右
5	<<、>>	位左移、右移运算符	由左至右
6	>、>=、<、<=	关系运算符	由左至右
7	==、!=	关系运算符	由左至右
8	&（位运算符 AND）	位逻辑运算符	由左至右
9	^（位运算符号 XOR）	位逻辑运算符	由左至右
10	|（位运算符号 OR）	位逻辑运算符	由左至右
11	&&	逻辑运算符	由左至右
12	||	逻辑运算符	由左至右
13	?:	条件运算符	由右至左
14	=	赋值运算符	由右至左

03 Python 运算符

以下的代码的输出将是什么? 说出你的答案并解释?

```
def div1(x,y):
    print("%s/%s = %s" % (x, y, x/y))

def div2(x,y):
    print("%s//%s = %s" % (x, y, x//y))

div1(5,2)
div1(5.,2)
div2(5,2)
div2(5.,2.)
```

03 Python 运算符

以下的代码的输出将是什么? 说出你的答案并解释?

```
def div1(x,y):
    print("%s/%s = %s" % (x, y, x/y))

def div2(x,y):
    print("%s//%s = %s" % (x, y, x//y))

div1(5,2)
div1(5.,2)
div2(5,2)
div2(5.,2.)
```

python3:

```
5/2 = 2.5
5.0/2 = 2.5
5//2 = 2
5.0//2.0 = 2.0
```

python2:

```
5/2 = 2
5.0/2 = 2.5
5//2 = 2
5.0//2.0 = 2.0
```

注:

在 Python 3 中, / 操作符是做浮点除法,而 // 是做整除.

而在 Python 2 中, / 就是整除,即和 Python3 中的 // 操作符一样。

03 Python 运算符

数值类型标准类型函数

1). 标准类型函数(cmp, str和 type): 可以用于所有的标准类型。

2). 转换工厂函数(int, long, float, bool和 complex)

3). 功能函数(abs, divmod, pow和 round)

4). 进制转换函数(hex, oct)

5). ASCII转换函数(chr, ord)

```
>>> divmod(10, 3)
(3, 1)
>>> pow(3, 3)
27
>>> round(3.141514, 3)
3.142
>>> import math
>>> for i in range(5):
...     print(round(math.pi, i))
...
3.0
3.1
3.14
3.142
3.1416
```

```
>>> hex(16)
'0x10'
>>> hex(15)
'0xf'
>>> hex(10)
'0xa'
>>> oct(7)
'0o7'
>>> oct(8)
'0o10'
>>> oct(9)
'0o11'
>>> ord('a')
97
>>> chr(97)
'a'
>>>
```

相关拓展模块:

decimal, array, math, random

03 Python 运算符

I/O操作

输入Input：

1). input与raw_input

2). input与getpass

input: python2中接收整数，

python3中接收字符串;

raw_input: python2中接收字符串，

python3中删除;

getpass: 接收密码， 输入的信息不可见;

转换类型	含义
d,i	带符号的十进制整数
o	不带符号的八进制
u	不带符号的十进制
x	不带符号的十六进制（小写）
X	不带符号的十六进制（大写）
e	科学计数法表示的浮点数（小写）
E	科学计数法表示的浮点数（大写）
f,F	十进制浮点数
g	如果指数大于-4或者小于精度值则和e相同，其他情况和f相同
G	如果指数大于-4或者小于精度值则和E相同，其他情况和F相同
C	单字符（接受整数或者单字符字符串）
r	字符串（使用repr转换任意python对象）
s	字符串（使用str转换任意python对象）

输出Output：

1). print

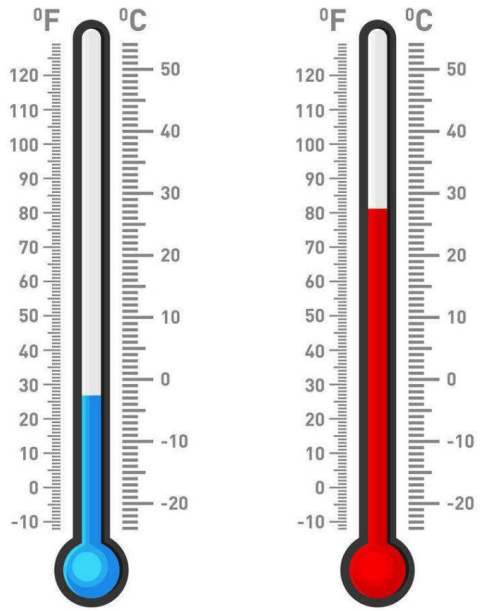
需求: 输入某学生的三门课程成绩，计算出该学生的总成绩sumScore和平均成绩avg_score(保留两位小数点)。

提示: (course1+course2+course3)/3

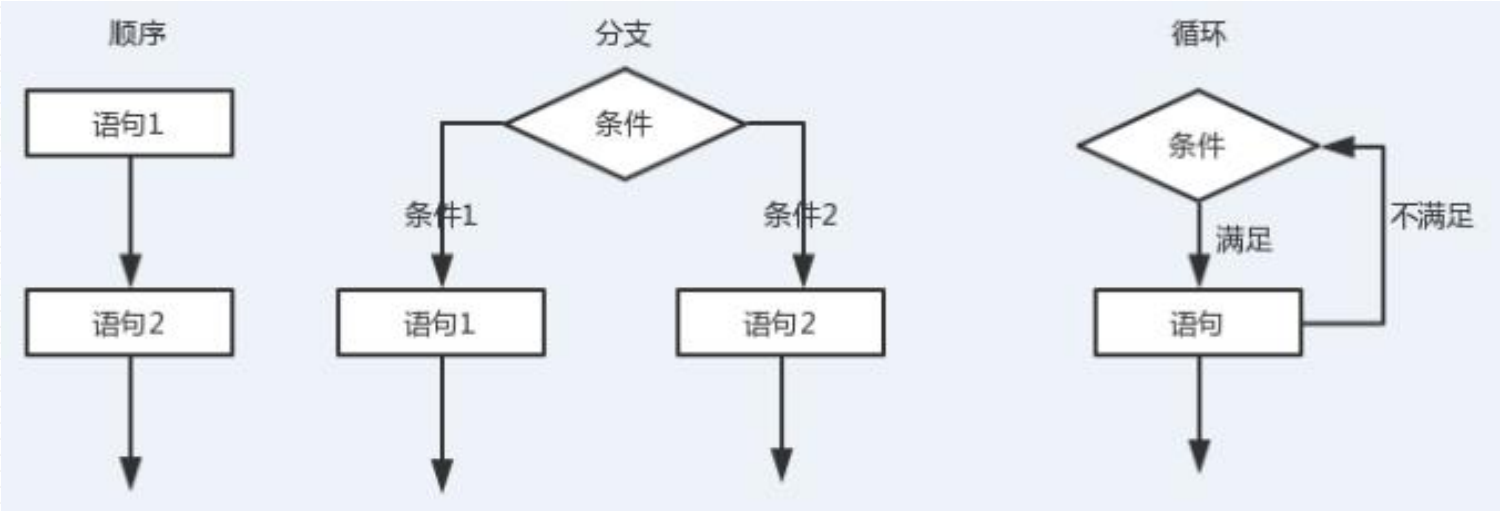
03 Python 运算符

I/O操作练习

编写程序将温度从华氏温度转换为摄氏温度。转换公式为celsius * 1.8 = fahrenheit - 32



04 Python 分支与循环



04 Python 分支与循环

代码块及缩进对齐

python 有两大特性, 一是简洁,二是可读性好。Python代码块缩进完全能够清楚地表达一个语句属于哪个代码块。

```
1
2 age = int(input())
3 if age >= 18:
4     print("已成年")
5 else:
6     print("未成年")
7
8
9
```

04 Python 分支与循环

if 语句四种语法结构

```
if expression:
    if_suite
```

1.标准 if 条件语句

```
if expression:
    if_suite
else:
    else_suite
```

2.if-else 语句

```
if expression1:
    if_suite
elif expression2:
    elif_suite
else:
    else_suite
```

3.if-elif-else语句

```
if_suite if expression1 else else_suite
```

4.三元运算符

```
a>b?a:b
a if a>b else b
```

if 语句练习题目

需求: 判断给定年份是否是闰年？

规则：

一个闰年就是指它可以被 4 整除,但不能被 100 整除, 或者它既可以被 4 又可以被 100 整除。

解释: year能被4整除但是不能被100整除 或者 year能被400整除, 那么就是闰年;

测试用例：

1992,1996 和 2000 年是闰年,但 1967 和 1900 则不是闰年。（and, or, not）

年份	二月份天数	年份	二月份天数	年份	二月份天数
1981	28	1989	28	1997	28
1982	28	1990	28	1998	28
1983	28	1991	28	1999	28
1984	29	1992	29	2000	29
1985	28	1993	28	2001	28
1986	28	1994	28	2002	28
1987	28	1995	28	2003	28
1988	29	1996	29	2004	29

04 Python 分支与循环

while循环语句语法结构

while循环原理： while 循环的 suite_to_repeat 子句会一直循环执行, 直到 expression 值为布尔假.

```
while expression:
    suite_to_repeat
```

while循环标准格式

```
count = 0
while (count < 9):
    print('the index is:', count)
    count += 1
```

1.计数循环

```
while True:
    cmd = input()
     if cmd:
        os.system(cmd)
```

2.无限(死)循环

04 Python 分支与循环

for循环语句语法结构

与传统语言(e.g.C/C++,Java)中的 for 语句不同, Python 的 for语句更加简洁.

for循环原理(有概念即可, 讲生成器时详细说):

可以遍历序列成员, 可以用在 列表解析 和 生成器表达式中, 它会自动地调用迭代器的 next() 方法, 捕获 StopIteration 异常并结束循环(所有这一切都是在内部发生的).

for循环一般语法

04 Python 分支与循环

range() 内建函数

range语法总结：

range(start, end, step =1)返回一个包含所有 k 的列表, start <= k < end , k每次递增 step

试一试:

range(6)

range(1, 6)

range(0, 10, 2)

range(1, 10, 2)

range(-4, 4)

range(6, 1, -1)

04 Python 分支与循环

for循环语句语法结构

试一试：

```
for item in 'hello':
    print("字符显示:", item)
```

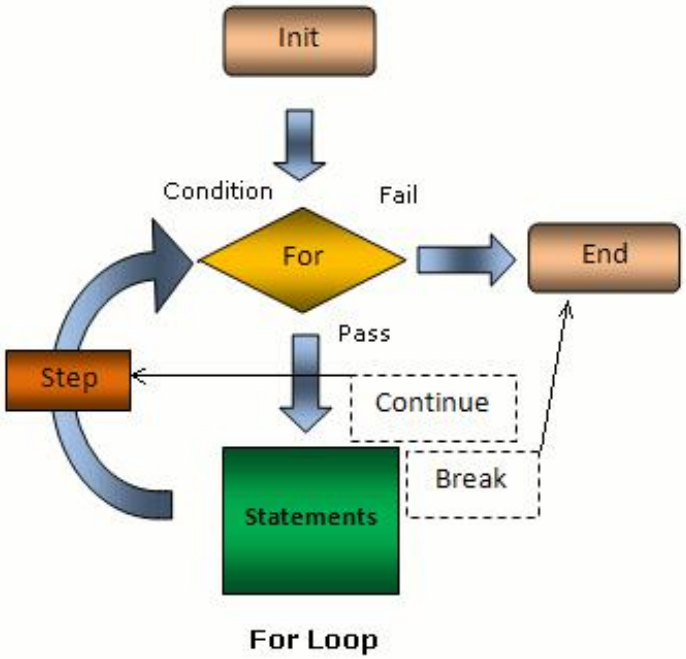
1.序列类型for循环

```
for item in range(10):
    print("变量显示:", item)
```

2. range() 内建函数

04 Python 分支与循环

跳出循环语句break 语句和continue 语句



break语句用来终止循环语句，即循环条件没有False条件或者序列还没被完全递归完，也会停止执行循环语句。

continue 跳过当前循环的剩余语句，然后继续进行下一轮循环。

04 Python 分支与循环

循环语句练习

需求1： 求1～100之间所有偶数的和;

需求2： 求1～100之间所有奇数的和;

需求3： 用户输入一个整形数， 求该数的阶乘; 3!=3*2*1=6

04 Python 分支与循环

用户登陆程序

用户登陆程序需求:

1. 输入用户名和密码;
2. 判断用户名和密码是否正确?

   name='root'

   passwd='westos'
3. 为了防止暴力破解， 登陆仅有三次机会， 如果超过三次机会， 报错提示;

提升: 密码能不能明文存储? 为什么?如何解决?

用户登录

黑客帝国

04 Python 分支与循环

密码加密算法Base64

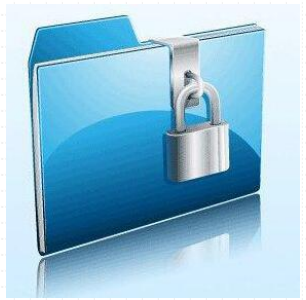
Base64编码，64指A-Z、a-z、0-9、+和/这64个字符，还有“=”号不属于编码字符，而是填充字符。

优点：方法简单

缺点：不保险，别人拿到密文可以自己解密出明文

编码原理(了解即可)：

将3个字节转换成4个字节((3 X 8)=24=(4X6))，先读入3个字节，每读一个字节，左移8位， 再右移四次，每次6位，这样就有4个字节了。



代码实现:

```
s1 = base64.encodestring('hello world')
s2 = base64.decodestring(s1)
```

04 Python 分支与循环

猜数字游戏作业题目

有猜数字游戏，其游戏规则为：

1). 程序内置一个 1 到 100 之间的数字作为猜测的结果，由用户猜测此数字(仅5次机会)。

2). 用户每猜测一次，由系统提示猜测结果：大了、小了或者猜对了；

3). 直到用户猜对结果，则提示游戏胜利。用户可以提前退出游戏，即，游戏过程中，如果用户录入数字0，或者超过5次机会，则游戏失败。

**

本节练习一

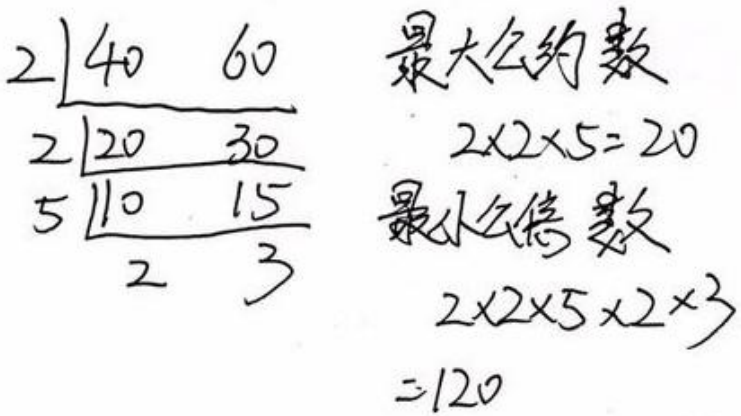
最大公约数和最小公倍数.

输入两个数值, 求两个数的最大公约数和最小公倍数.

1.两个或多个整数公有的倍数叫做它们的公倍数，其中除0以外最小的一个公倍数就叫做这几个整数的最小公倍数。

2.求最小公倍数(lcm)的算法: 最小公倍数 = 两个整数的乘积 / 最大公约数

3. [40, 60] = 120



**

本节练习一

最大公约数和最小公倍数.

输入两个数值, 求两个数的最大公约数和最小公倍数.

1. 最大公约数就是A和B能整除的最大的数;

2. 求最大公约数(gys)算法:

方法1： 欧几里得法(辗转相除法)

1). 整数A对整数B进行取整, 余数用整数C来表示　举例: C = A % B

2). 如果C等于0,则C就是整数A和整数B的最大公约数

3). 如果C不等于0, 将B赋值给A, 将C赋值给B ,然后进行 1, 2 两步,直到余数为0, 则可以得知最大公约数。

3. (40, 60) = 20

本节练习一

最大公约数和最小公倍数.

输入两个数值, 求两个数的最大公约数和最小公倍数.

1. 最大公约数就是A和B能整除的最大的数;

2. 求最大公约数(gys)算法:

方法2： 穷举法(一个一个除)

1). A,B两个数的最大公因数肯定小于等于相对更小的那个数;

2). 从两个数中较小数开始由大到小列举;

3). 直到找到公约数立即中断列举，得到的公约数便是最大公约数;

3. (40, 60) = 20

**

本节练习二

x 的平方根

实现 `int sqrt(int x)` 函数。

计算并返回 *x* 的平方根，其中 *x* 是非负整数。

由于返回类型是整数，结果只保留整数的部分，小数部分将被舍去。

示例 1:

```
输入: 4
输出: 2
```

示例 2:

```
输入: 8
输出: 2
说明: 8 的平方根是 2.82842...,
     由于返回类型是整数，小数部分将被舍去。
```

本节练习三

求解一元二次方程

解题要点：

利用math的sqrt()方法取平方根;

$$Let\ ax^2+bx+c=0 \quad (a\neq 0)$$

$$x1=\frac{-b+\sqrt{b^2-4ac}}{2a}$$

$$x2=\frac{-b-\sqrt{b^2-4ac}}{2a}$$

when $\triangle = b^2-4ac=0$ *only one repeated root*

when $\triangle > 0$ *two real roots*

when $\triangle < 0$ *no real root*

总结

1. python: 解释型语言;
   Version： python2.7 -->python2.8(x, 没有), python3.x
   Anaconda
   Pycharm(IDE集成化管理工具)

2. python数值类型: int, float, long, bool, complex
   运算符: /, **, %, a+=1, //

3. Python 分支与循环:顺序， 分支， 循环

分支： if, if-else, if-elif-else, 三元运算符("成年" if age>18 else "未成年")

循环: while, for,(range(5), range(0,101, 2)) , while....else...., for.....else....

4. 卸载Pycharm

```
rm -fr pycharm-2018.3.2/        # 删除安装包
rm -fr ~/.PyCharm2018.3/        #删除配置信息
```

总结

5. pycharm常用设置总结(Ctrl+Alt+s)

1). Appearance: 菜单栏的

感谢聆听！

THANK YOU!