①hello

#coding:utf-8 #编码声明，表示可以支持中文

#这是一段注释 快捷键是ctrl+?/ 单行注释

'''

这是一个代码块的注释，也称为多行注释

'''

'''

print "hello world!"

print "hello python!"

②变量

#coding:utf-8

'''

变量，可变的量，区别于常量，常量为固定不变的量。

变量也可以理解为容器，可以存储值，也可以被其他对象引用。

变量的定义方式：变量名=value（值） 例：a=1

多个变量的命名方式： 变量名1,变量名2....=value1，value2.... 例：a,b=1,2

删除变量名格式： del 变量名(删除后的变量名再也无法访问)

'''

'''

定义变量名尽量做到见名知意

例如：你定义一个变量为count,我大概就知道它要表示一个总数

'''

'''

变量名定义时要注意的事项：

1. 变量名不允许以数字开头

2. 变量名不支持除了\_（下划线）以外的特殊字符

3. 变量名是区分大小写的 例a=1 A=2

4. 不支持中文作为变量名，必须使用英文

5. 不能使用python关键字作为变量名 例：print就是python的关键字，不能用来做变量名

'''

\_a=1

a = 1

A = 2

print a

print A

b,c=3,4

print b

print c

# del a

# print a

③变量类型-数值类型

#coding:utf-8

'''

变量类型（4种）：1.数值类型 2.字符串类型 3.序列类型 4.映射类型

'''

'''

数值类型（4种）：1.整型 2.长整型 3.浮点型 4.复数类型

'''

#整型（int）：整型用于代表数字中所有的整数，带符号（包括负整数）

#查看类型函数使用：type(变量名)

a = 2

b = -2

print a

print type(a)

print type(b)

print "==========华丽的分割线1==========="

#长整型(long)：用来代表数字中位数较大的整数

e = 123L #固定格式，需要在数字后加L，L也可以用小写，但不建议，避免与数字1混淆

print e,type(e)

print "==========华丽的分割线2==========="

#浮点型(float)：用于表示数字中所有的小数（包括负数小数）

c = 3.14

d = -3.3

print c,d,type(c),type(d)

print "==========华丽的分割线3==========="

#复数类型（complex）:分为实部和虚部

h = 3+2j #固定格式 虚部后面要加一个j，也可以是大写

print h,type(h)

④变量类型-字符串类型

#coding:utf-8

'''

字符串（string）：当我们的值不是纯数字的时，必须使用字符串定义，字符串可以定义内容包括

数字、汉字、符号、字母等。

'''

#定义方式： ①变量名='xxxx' ②变量名="xxxx" ③变量名='''xxxx'''

g = '中文'

print g,type(g)

h = "English"

print h,type(h)

j = '''@'''

print j,type(j)

k = "123"

print k,type(k)

i = "I'm jack"

k = 'I\'m jack' # 用单引号定义时，要是里面内容有单引号，必须使用 \ （转移符）

print i

⑤变量类型-序列类型

#coding:utf-8

'''

序列类型（两种） 1.列表 2.元组

'''

'''

1.列表(list)：有序数组，列表可以添加多个数据，添加字符串，数字，列表等。

列表定义方式： 变量名 = [xxx,'xxx',[xxx,"xxxx"]] #每个元素使用英文逗号隔开

'''

list = [123,"jack",456,[277,"Tom"]]

print list

print type(list)

list1 = ["你好",'Jan']

#转换中文输出格式： str(变量名).decode('string-escape')

print str(list1).decode('string-escape')

print "================分割线============"

'''

2.元组（tuple）:有序数组，代表多个元素的组合，特点是只读类型变量，意味一旦生成，内容不可修改。例：身份证号码

元组的定义方式：变量名 = (xxx,'xxx',[xxx,"xxxx"])

'''

tuple = (123,"jack",456,[277,"Tom"])

print tuple

print type(tuple)

tuple1 = ("你好",'Jan')

#转换中文输出格式： str(变量名).decode('string-escape')

print str(tuple1).decode('string-escape'),type(tuple1)

⑥变量类型-映射类型

#coding:utf-8

'''

字典（dictionary）：字典由两部分构成 key name(键名)与key value（键值）

字典的定义方式： 变量名 = {'键名1':键值1,'键名2':键值2.....}

'''

dict = {"name":'chen','age':18,'sex':"男",'kg':50}

print dict

print type(dict)

print str(dict).decode('string-escape')

⑦python打印集合

#coding:utf-8

'''

python打印必须使用print 打印结果（函数运行结果）

'''

#1.打印纯数字

print 123

#2.打印字符串

print 'jack',;print "tom",;print '''jan'''

#3.打印变量名 print 变量名

a = 1

print a

#4.打印多个内容，格式： print 变量名1,内容1,type(变量名)

print a,123,"chen",type(a)

#5.为打印添加说明

age = 18

print "my age is",age

#6.打印格式字符（格式化输出）

# ①打印整型（%d） decimal 十进制的整数

age = 19

print "my age is %d" %age #注：当变量名为整型的时候，可以使用%d来代替变量，%age是代入变量

#② 打印字符串（%s） string

name = "张三"

print "我的名字是：%s" %name

#同时打印整型和字符串时使用 %（变量名1，变量名2）

print "my age is %d,my name is %s." %(age,name)

#③打印浮点型（%f）

shengao = 172.345

print "my shengao is :%f" %shengao #浮点型打印时默认保留6位小数

print "my shengao is :%2.2f" %shengao

print "my shengao is :%8.2f" %shengao

#第二和第三条格式： %n.nf(第一个n代表打印宽度，总数=整数+小数点+小数位数；第二个n代表小数后保留位数)

#补充： 当第二个n生效保留完小数点后总位数依然大于第一个n时，第一个n不生效

# 当第二个n生效保留完小数点后总位数依然小于第一个n时，第一个n的总数才会生效，并用空格补齐不足位数

⑧python运算符

#coding:utf-8

'''

运算符（5种）：1.算数运算符 2.关系运算符 3.赋值运算符 4.成员运算符 5.逻辑运算符

'''

'''

1.算数运算符（7个）

+ - \* / % \*\* //

加 减 乘 除 取余 幂运算 取整

'''

print '1+1=',1+1

print '2-2=',2-2

print '2\*2=',2\*2

print '1/2=',1.0/2 #若被除数与除数都为int类型时，结果也会为int类型，如果需要打印小数，那么被除数与除数其中一个必须为float类型

print '10%3=',10%3

print '2\*\*2=',2\*\*2

print '10//3',10//3

print "==================分割线===================="

'''

关系运算符（7个） 运算结果返回布尔值（True或False）

< > <= >= != <> ==

小于 大于 小于等于 大于等于 不等于 不等于 相等

'''

print '1<2',1<2

print 1>2

print 1<=2

print 1>=2

print 1!=2

print 1<>2

print 1==2

print "==================分割线===================="

'''

赋值运算符(8个)

= += -= \*= /= %= //= \*\*=

等号 加等 减等 乘等 除等 取余等 取整等 幂运算等

'''

a = 1

a+=1 #同 a=a+1

print a

b = 2

b-=1 #同 b=b-1

print b

print "==================分割线===================="

'''

成员运算符（2个） 返回结果为布尔值

in 在该有序数组内

not in 不在该有序数组内

'''

c = [1,2,[3,4]]

print 1 in c #True

print 3 in c #

print 4 in c

print [3,4] in c

print 2 not in c

print "==================分割线===================="

'''

逻辑运算符（3个）

and 与 例：a and b若整体要为True,则a与b都应为True，否则结果为False

or 或 例：a or b若整体要为True,则a与b至少一个为True，否则结果为False

not 非 例： not a若要为True，那么a本身为False

'''

a = 1<2 #True

b = 1>2 #False

print a and b

print a or b

print not a

⑨输入函数

#coding:utf-8

'''

1.input()函数

格式： 变量名 = input('请输入文字说明')

input()函数会自动识别输入内容的能力，常用于Number（数字）类型，若要进行字符串输入不建议使用

'''

'''

age = input("请输入你的年龄：")

print "你的年龄是%d岁" %age

print type(age)

name = input("请输入你的名字：") #键盘输入字符串时，需要加上引号

print "你的名字是%s" %name

print type(name)

print "==================分割线==================="

'''

'''

2. raw\_input()函数输入

格式： raw\_input('请输入文字说明')

raw\_input函数不论输入的内容为数字还是字符串都将被视为字符串类型

'''

# name1 = raw\_input("请输入你的名字:")

# print "你的名字是：%s" %name1

# print type(name1)

# input() 和 raw\_input()函数的区别

a = input("请输入一个表达式：") #表达式输入2+2

print a,type(a)

b = raw\_input("请输入一个表达式：") #表达式输入2+2

print b,type(b)

⑩程序流程结构--分支结构if

#coding:utf-8

'''

程序结构：3种

1.顺序结构：即语句从上到下按顺序执行

2.分支结构：一条大路，会有很多分支路口。在Python中常用if elif else 判断语句

3.循环结构：如歌曲/歌单循环播放。在Python中常用while循环和for循环

'''

'''

2.分支结构：if条件语句：若满足对应条件，则执行对应语句

'''

#①未分支前

#格式：

#if 条件语句: （注意条件语句后有冒号）

# 对应语句 （注意对应语句需要缩进）

# if 1<2: #当1>2为假的时候，下面对应的语句则不会执行

# print "当if条件为真的时候，我才会打印" #使用缩进表示代码块，属于if代码块下的内容

# print "这个print是一个新的代码块，什么时候都会打印，不受上面的条件影响"

#②if else分支

'''格式:

if 条件语句:

对应语句1

else:

对应语句2

'''

# if 1>2:

# print "如果if条件为真的话，我就会打印"

# else:

# print "如果if条件为假的话，我就会被打印"

#③if elif else 分支结构（其中elif可以分支很多条路）

'''

格式：

if 条件语句1:

对应语句1

elif 条件语句2:

对应语句2

elif 条件语句3:

对应语句3

.......

else:

对应语句n

'''

sex = raw\_input("请输入你的性别：")

if sex=="male":

print "你的性别是男"

elif sex=="female":

print "你的性别是女"

else:

print "性别输入错误"

①①程序流程结构--循环结构

#coding:utf-8

'''

循环结构（2种）1.while循环 2.for循环

'''

'''

1.while循环：多次循环，当条件为真（True）是，则会运行循环体语句，直到条件为假（False）时跳出循环

格式：

while 条件语句：

循环体语句

'''

'''

a=1

while a<10:

#a+=1 #同a=a+1 为了防止死循环

print a

a+=1

'''

'''

for循环：遍历循环

作用：将一个有序数组中所有数据按顺序依次进行输出的过程（包括不仅限于字符串，列表，元组，字典等）

格式:

for 变量名1 in 变量名2:

for循环相当于依次把变量名2有序数组中的数据赋值给变量名1

'''

# ①遍历字符串

# a = "python"

# for str in a:

# print str

# ②遍历列表

# list = [1,2,3,'1994',[2,3]]

# for l in list:

# print l

#③遍历元组 略

#④遍历字典

dict = {"name":"tom","age":18,"sex":"男"}

for d in dict: #遍历字典时，默认是遍历键名，不会输出键值

print d,dict[d] #dict[d]是输出键值的方法

①②循环结构中的关键字

#coding:utf-8

'''

break:中断循环，当条件语句为真时，直接中终止跳出循环

continue:跳出本次循环，执行下一轮循环

pass:占位符

'''

#while举例：

'''

a = 0

while a<10:

a+=1

if a==7:

# break

# continue

pass

print a,

else: #当上述循环非强制情况下完成循环，则执行else下的语句

print "=====over======"

'''

#for 举例：

str = "python"

for obj in str:

if obj=='t':

continue

print obj

else:

print "=====over===="

①③字符串的切片和方法

#coding:utf-8

'''

切片，截取，切片与截取基于index下标索引进行截取字符串的一部分或者全部，python的index从0开始计算位数

格式： 变量名[index]

'''

#1.正序切片:

a = "Hello"

print a[1] #切取单个内容

print a[0:3] #切取下标索引为0~3之间不包括3的内容

print a[:3] #从左边0开始切片直到切取到有边界值以内的值，不包括边界值

print a[0:] #从下标索引为0数据开始，切片直到最后一位为止

#2.倒序切片

#就是把下标索引倒过来即最后一位为-1，倒数第二位为-2，依次类推，下标索引可以倒序，但不能反切

#（必须按照原有变量中的顺序进行切片）

print a[0:-3] #下标所有右到左算，倒数第三位为边界值，切取后剩He

print a[0:-2]

#3. 切完整的Hello方法：

print a[0:]

print a[:]

print a[0:5]

#字符串支持的方法:

#1.加法运算

a = "Hello"

b = "python"

c = a+b #将a,b两个字符串进行合并，编程一个新的字符串

# print c,type(c)

print a+' '+b,type(a+b)

#2.乘法运算

print c\*2

①④字符串支持的函数

#coding:utf-8

#1.upper() 2.lower() 3.capitalize()

#格式：变量名.upper()

s = "helLoPyThoN"

print s.upper() #全部字母大写格式

print s.lower() #全部字母小写格式

print s.capitalize() #首字母大写格式

#1.find()

t = "I love python"

#find查找子字符串，返回子字符串的首字符下标索引

print t.find("love")

print t.find('py')

print t.find("heno") #当不包含子字符串是，返回-1

#1.split() #字符串分割，以列表返回分割后的部分

c = "Hello world"

print c.split(),type(c.split()) #默认以空格作为分隔符

d = "He:llo: wor:ld"

print d.split(":") #指定以：作为分隔符

#1.startswith(),endswith()

#判断字符串以xxx开头/结尾，返回布尔值

print "helloworld".startswith("a")

print "helloworld".startswith("hel")

print "helloworld".endswith("ld")

①⑤列表的切片和方法

#coding:utf-8

#1.列表的切片方法同字符串

# ①正序切法

'''

a = [1,2,3,'Jack',[3,4]]

print a[4]

print a[0:3]

#②倒序切法

print a[1:-1]

# 2.列表支持加法和乘法运算

b = [1,2,3,'Hi']

c = [4,5,6,'Jack']

print b+c #生成一个新的列表

print b[3]+' '+c[3] #不同列表的字符串可以相加

print b[1]\*3 #若元素为纯数字，可以进行算法运算

print b[1]\*c[2] #不同列表都为纯数字时，可以进行算法运算

print b[3]\*c[0]

# print b[3]\*c[3] #不支持字符串相乘

# print b[2]+c[3] #不支持整型和字符串相加

'''

#3.修改删除列表的元素

#修改格式： 变量名[index]=value（修改的值）

#删除格式： del 变量名[index]

d = [1,2,3,4,5,6]

print d #原列表数据

d[2] = 7

print d #修改后的列表数据

del d[5]

print d #删除后的列表数据

①⑥列表支持的函数

#coding:utf-8

#1.append()函数：向列表追加元素（把元素加到列表末尾）

# 格式：变量名.append(新元素) 注：新元素只能传一个

a = [1,2,3,3,4,5]

a.append(6)

print a

#2. insert()函数：向列表插入元素（通过索引从指定位置插入）

#格式：变量名.insert(index,新元素)

a.insert(1,'tom')

print a

#3. count()函数:返回某个值在列表中出现的次数

#格式：变量名.count(元素)

print a.count(3)

#4.sorted()函数：内容排序

#格式：print sorted(变量名)

#先排列列表中的数字类型（从小到大），然后排列列表类型，再排列符号与字符串，当字符串

#为英文时则按照字母排序，先排序大写开头（A~Z），再排序小写（a~z），首字母相同则对比第二个字母，依次类推

#5.max()函数：列表中的最大值 max(变量名)

# min()函数：列表中的最小值

# len()函数：列表中数据个数

list =[1,1990,27,175.2,'Jack','JAckson','male',' ',[60,120],"中文"]

print list #原列表输出是根据顺序打印

print str(sorted(list)).decode('string-escape')

print max(list)

print min(list)

print len(list)

①⑦元组的内容

#coding:utf-8

'''

元组（tuple）:属于只读变量，所以不支持增加、删除和修改元组内的元素

'''

#1.元组的切片，同列表。（举例：参考列表，把中括号改为小括号即可）

#2.元组的加法和乘法运算，同列表。

# b = (1,2,3,'Hi')

# c = (4,5,6,'Jack')

# d = b+c

# print d

#3. 无法修改和删除单个元素，可删除整组元素

# 格式：del 变量名

a = (1,3,3,4,2,5)

print a

# del a

# print a

#4.元组中支持的函数

#count()函数

print a.count(3)

#sorted()函数

print sorted(a),type(sorted(a))

#max(),min(),len() 参考列表

①⑧字典的内容

#coding:utf-8

'''

字典（dict）:一个字典由多个键构成，每个键又由键名和键值构成（使用：分隔）

键与键之间使用,隔开

'''

#1.字典在生成后会进行默认的键重排序（只要有新的值产生后都会重新排序）

dict = {"name":"Jack",'age':27,"sex":"male","kg":90}

print dict

#2.字典中若出现相同键名，则后者键会替换前者，只保留一个键，所以键名是唯一的。

dict1 = {"name":"Jack","name":'Tom','age':27,"sex":"male","kg":90}

print dict1

#3.字典切片不能使用下标索引，根据键名获取对应键值

#格式 变量名[键名]

print dict["name"]

#4.修改字典内的数据，根据键名修改对应键值

#格式：变量名[键名]=新键值

dict['age']=29

print dict

#5.排序使用sorted()函数，结果是对键名排序，排序后返回列表形式

dict3 = {"name":"Jackson",'age':23,"sex":"female","kg":80}

dict4 = sorted(dict3) #排序后的字典

print dict4,type(dict4)

#6.删除键（根据键名删除整个键）

# 格式：del 变量名[键名]

dict5 = {"name":"Jackson",'age':23,"sex":"female","kg":80}

del dict5["name"]

print dict5

#7.添加键

# 格式：变量名[新键名]=新键值

dict5["height"]=172.2

print dict5

①⑨python函数的作用

#coding:utf-8

#函数就是一种运算法则，小程序

#函数的作用是为了降低代码的复杂程序。

#例如：max()、min()、len()、sorted()....这些为python安装时自带的函数，这些函数为内置函数，可以

#直接进行调用来运行其中的计算法则。

# a = [1,2,3,4,5]

# print max(a)

'''

abs()函数：绝对值函数，我们使用时直接print abs(-9)

'''

# print abs(-9)

num = input("请输入一个数字：")

if num<0:

print -num

else:

print num

②〇 python中的range()函数

#coding:utf-8

#range()函数：常用于Number

'''

格式：

range(x,y,step),只传一个参数时，传入的是y（边界值）

传入两个参数时，传入的是x,y,分别代表数字与右边界值

传入三个参数时，传入的分别是数字，右边界值，步长（从起始数字开始每隔多少提取一个数字放入到生成的列表中）

'''

print range(20) #生成一组纯数字从0~19的列表一共20个数字，不包括20本身

print range(1,20) #生成1~19的纯数字列表，不包括边界值

print range(1,20,2) #1~19中每隔2步去一个数字放入新列表中

#通过for遍历函数查看输出的值：

for i in range(20):

print i,

②①构建函数

#coding:utf-8

'''

构建函数使用def关键字

构建函数有四种形式：

1.不传递任何参数

2.传递固定个参数

3.缺省参数，默认参数，当不传缺省参数的对应参数时按照默认进行输出

4.不定个数参数，就是在对应参数前加\*

'''

# 1.不传递任何参数

def info(): #构建函数以def开头，info()为函数名

print "My name is Tom" #构建函数的对应语句（方法）

info() #调用函数的方法

# 2.传递固定个参数

def info1(name,age):

print "My name is %s,My age is %d" %(name,age)

info1("白居易",27)

# 3.缺省参数，默认参数，当不传缺省参数的对应参数时按照默认进行输出

def info2(name,age=28,kg=90): #注：默认参数只能放在非默认参数之后

print "My name is %s,My age is %d,My kg is %d" %(name,age,kg)

info2('Jack',30,80)

# 4.不定个数参数，就是在对应参数前加\*

def info3(\*num):

for i in num:

print "num is %d" %i

info3(1,2,3,4,5)

②②构建函数中的关键字

#coding:utf-8

# return :返回值

'''

def my\_abs(num):

if num<0:

print -num

else:

print num

print my\_abs(8)

'''

#其中my\_abs(8)已经调用了def里的方法，def里的方法就有print的操作，即print 8，而

#在调用方法前面加入print即是输出def返回的默认值，默认值为None

#注：若在编写函数时未添加返回值，则默认return一个None值

#理解print和return的区别：

def a():

print 1

def b():

return 3

d=a() #d是a()函数下的方法，print 1 即打印1

e=b() #e是b()函数下的方法，return 3 返回一个3的值

# print d+1 #打印出来的1是不能存储后期使用，所以不能执行1+1=2的操作

print e+1 #return返回的值可以存储着后续使用，因此可以打印3+1=4

# 注：print打印的值只是作为显示，但return返回的值可以存储继续使用

②③模块中的调用

#coding:utf-8

'''

模块在python中被视为自定义函数或是运算法则等内容，可被其他文件调用的.py文件

调用的方式有两种：

1.import 模块/包名

2.from 模块/包名 import 模块名/\*（代表所有模块）

注：被调用的.py文件，调用后项目会生成一个同名.pyc后缀的文件

'''

import iPhone #当只调用模块名时，需要使用模块名下面的函数，使用格式：模块名.函数名()

iPhone.huawei()

iPhone.xiaomi()

from iPhone import huawei #当只调用模块名下一个方法时，那只能使用这个方法下的对应语句

huawei()

from iPhone import \* #当调用了模块下所有的方法时，可以直接使用所有的方法

xiaomi()

huawei()

②④构建类

#coding:utf-8

'''

类是面向对象语言中独有的。

类：可以定义框架（类属性）和对应使用的方法，根据类可以构建对象

构建出的对象将满足可以使用该类所有的方法并且拥有所有的属性

类使用关键字 class

'''

'''

格式：

class 类名:

类变量=0 构建计数 （不是必须）

def \_\_init\_\_(self,属性1，属性2，属性3....):

self.属性1 = 属性1

self.属性2 = 属性2

self.属性3 = 属性3

....

类名.类变量+=1

def 方法1(self):

内置语句1

def 方法2(self):

内置语句2

.....

'''

class people: #people是类名

count=0 #count是类变量

def \_\_init\_\_(self,name,age,sex): #注意init前后有两个下划线，self是一个固定格式，后面的name,age,sex都是属性

self.name = name #封装属性

self.age = age

self.sex = sex

people.count+=1 #构建计数+1

print "构建完成"

def drink(self): #定义一个drink方法

print "I like coffee" #drink方法下的内置语句

def sleep(self):#定义一个sleep方法

print "I want to sleep all day" #sleep方法下的内置语句

#根据类构建对象实例 格式：变量名 = 类名(对应属性给予的值)

people1 = people('Jack',27,'male')

#根据该构建的对象使用所有方法 格式：变量名.函数名()

people1.drink()

people1.sleep()

#查看类变量的方法

print "我是第%d人" %people1.count #通过对象访问类变量

print "我是第%d人" %people.count # #修改对象实例的属性

print "长大两岁了"

people1.age = 29

print "我Jack现在%d岁了" %people1.age

#构建第二个对象

people2 = people('Tom',20,"male")

print "我%s是第%d个人" %(people2.name,people2.count)

②⑤类继承和重写

#coding:utf-8

'''

类的继承和重写

类的继承：可以理解为孩子（子类）继承父亲（父类）的财产、基因等。

类继承和重写，子类继承父类，若不修改重写内容，则默认父类所有的方法及属性框架

子类继承父类写法：

class 子类名(父类名):

'''

class father: #父类

def \_\_init\_\_(self,name,age):

self.name = name

self.age = age

def sleep(self):

print "我%s要去睡觉了" %self.name

def work(self):

print "我今年%d岁，我%s要去工作了" %(self.age,self.name)

#构建实例化对象

if \_\_name\_\_=="\_\_main\_\_": #其中name和main前后的是两个下划线，这句加在所有执行语句前面，这样可以避免其他包调用时

#执行到该方法的语句（注：这个方法加入后，下面的执行语句需要缩进）

father1 = father('Jack',30)

father1.sleep()

father1.work()

print father1.name

print father1.age

class son(father):

def \_\_init\_\_(self,name,age,sex): #添加了一个新的属性

self.name = name

self.age = age

self.sex = sex

def drink(self): #添加一个新的方法

print "我%s要喝奶奶" %self.name

def work(self): #重写的方法

print "我才%d岁，还不会工作" %self.age

if \_\_name\_\_=="\_\_main\_\_":

son1 = son("Jackson",2,'male') #构建实例化对象，需要对应属性个数

son1.sleep() #继承了父类的方法

son1.drink() #调用了自己定义的新方法

son1.work() #调用重写的方法

②⑥调用多重继承

#coding:utf-8

'''

类的调用和多重继承。

类的调用也是遵循模块调用的两种方式。

多重继承：可以理解为还有个孙子，他不单单可以继承父亲，还可以继承爷爷。

多重继承：多个类间的互相继承

多重继承的写法：

class 类名(父类名,子类名):

注：相同方法名的方法以及框架都优先继承父类，若不同方法则可以完全同时继承。

'''

from leijichenghechongxie import \*

class sun(father,son):

def play(self): #添加新方法

print "我%s还在妈妈的肚子里" %self.name

def drink(self): #重写了drink的方法

print "我%s还不会喝东西" %self.name

sun1 = sun('Jacksun',0) #优先继承父类名的框架（属性）

son1.sleep() #继承了父类的方法

son1.drink() #调用了自己定义的新方法

son1.work() #调用重写的方法