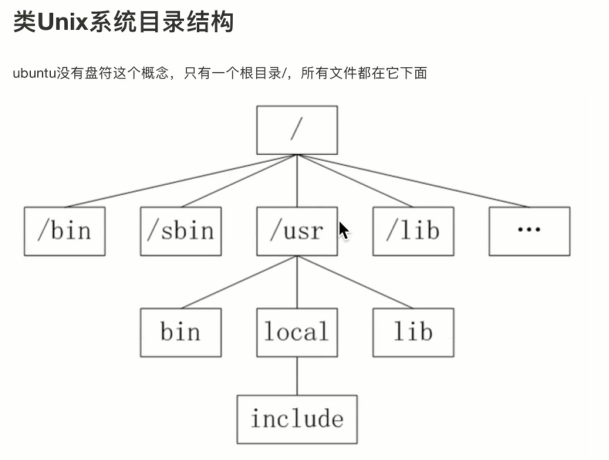
# Day01

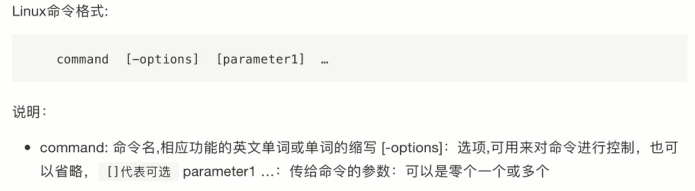
## Linux系统目录结构



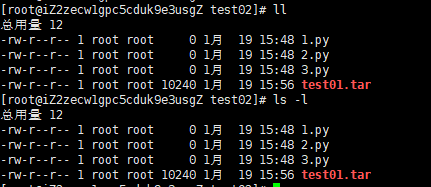
Window系统有盘符 ， linux系统有目录

## 命令的使用方法

### 基本格式



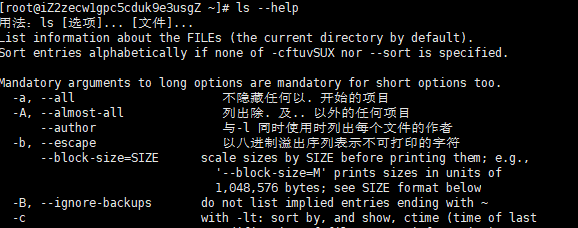
### ls 查看目录下所有文件和目录



### --help 查看帮助文档



例如:

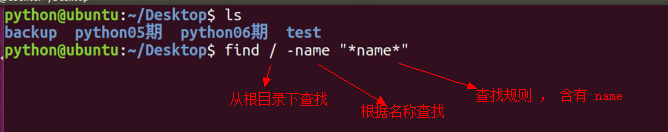


### Cat 查看文件内容



### find查找文件





### Tar打包和解压

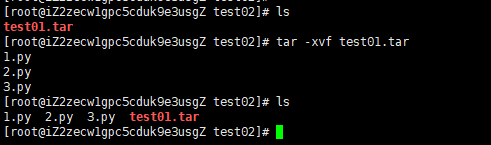


#### 压缩成 .tar

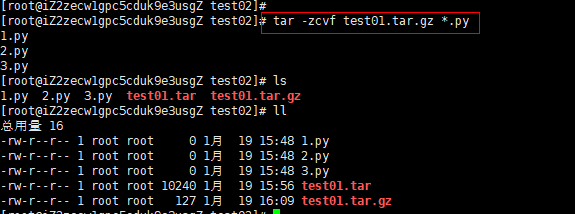


如同，touch命令创建三个文件,后打包成 test.tar

#### 解压 .tar

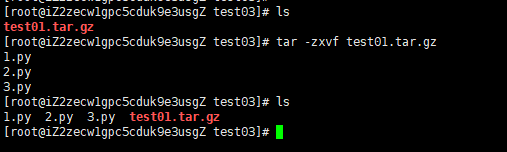


#### 压缩成 .tar.gz 格式



结果 .tar.gz 文件比 .tar格式文件小

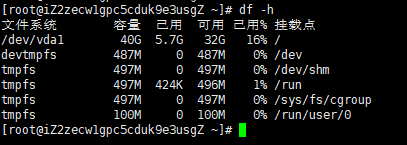
#### 解压 .tar.gz



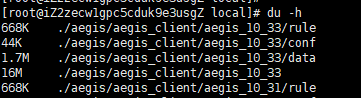
### ps 查看进程的信息



### df 查看硬盘占用空间情况

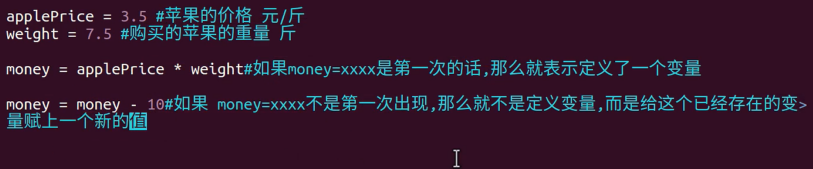


### du 查看某目录下文件占用空间情况

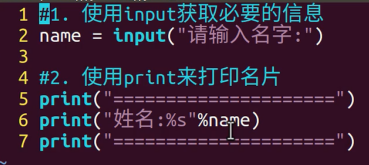


# Day03

## 定义变量，给变量赋值

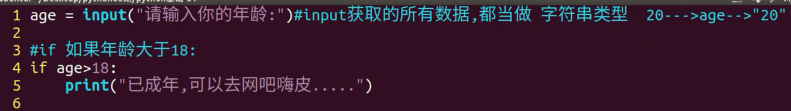


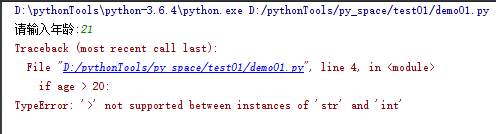
## 输入和输出



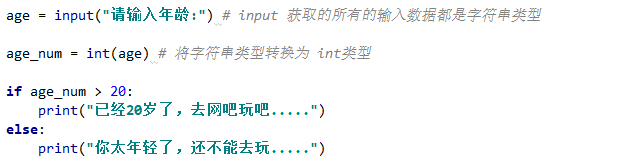
错误程序：

字符串类型与 数字类型比较出现问题

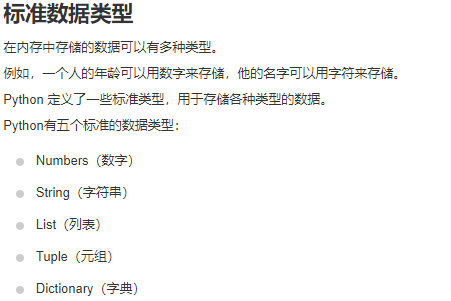




正确方式：进行类型的转换

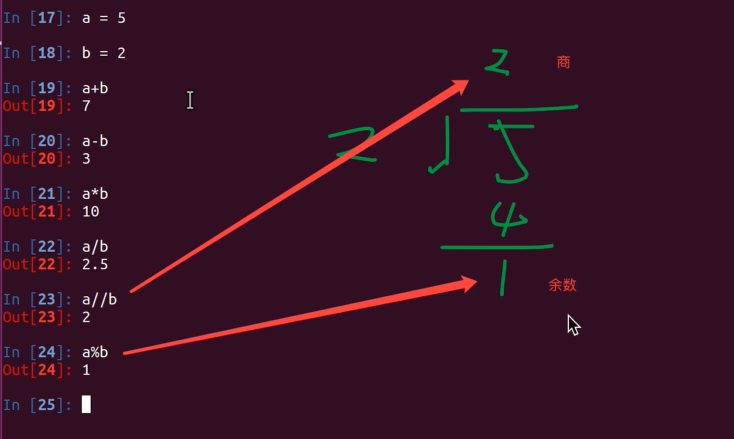


## 数据类型

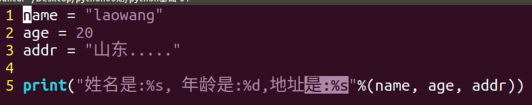


## 算术运算符





## Print一次输出多个变量

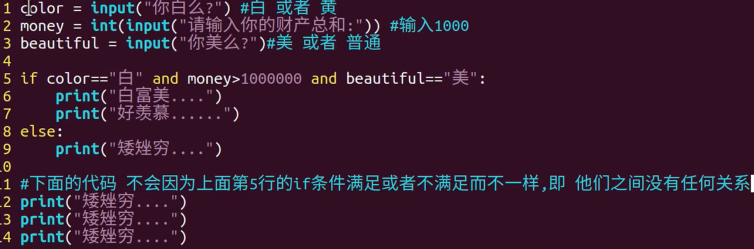


# Day04

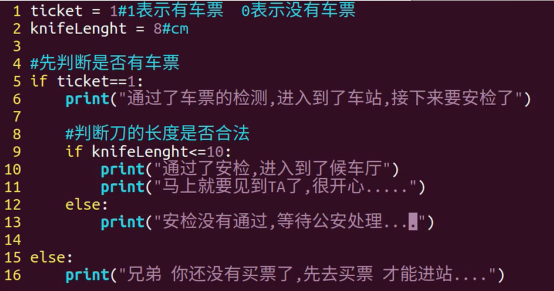
## 严格缩进



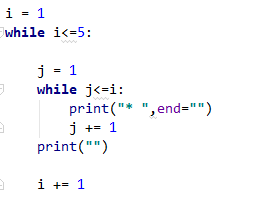
正常情况下前面是没有空格的



## If条件判断



## While循环

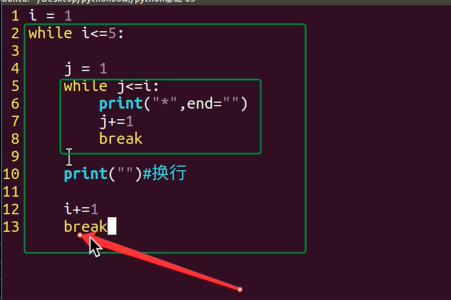


# Day05

## 结束循环

continue 语句跳出本次循环，而break跳出整个循环。

## 嵌套循环中break的作用范围



**break** , **continue 只在自己所在的循环中生效**

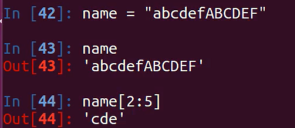
## 字符串和数字在内存中的存储方式

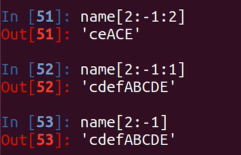


数字可以用较少的字节存储 ， 例如 数字100可以存储在一个字节中

字符串 “100“ 的存储需要三个字节，分别存储每个字符

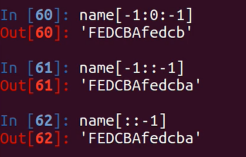
## 字符串切片，逆序



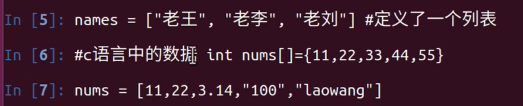


[ 2 : -1 : 2] 三个数字分别代表， 起点，终点，步长

逆序



## 列表



可以存储多种类型的数据

## 字典

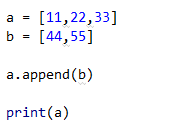
字典是另一种可变容器模型，且可存储任意类型对象。

字典的每个键值(key=>value)对用冒号(**:**)分割，每个对之间用逗号(**,**)分割，整个字典包括在花括号(**{})**中 ,格式如下所示：

d = {key1 : value1, key2 : value2 }

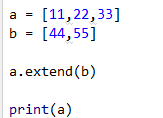
# day06

## 列表中append 和 extend 的 区别





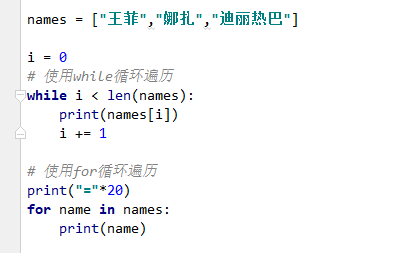
Append将整体作为一个元素添加





Extend 将后者的每个元素拆分后添加到前者中

## 遍历集合



## Python 元组

Python的元组与列表类似，不同之处在于元组的元素不能修改。

元组使用小括号，列表使用方括号。

元组创建很简单，只需要在括号中添加元素，并使用逗号隔开即可。

如下实例：

tup1 = ('physics', 'chemistry', 1997, 2000);

tup2 = (1, 2, 3, 4, 5 );

tup3 = "a", "b", "c", "d";

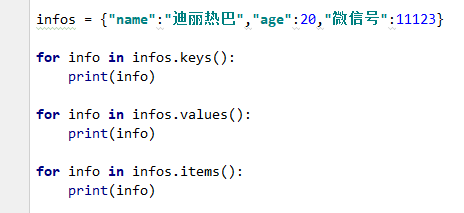
创建空元组

tup1 = ();

元组中只包含一个元素时，需要在元素后面添加逗号

tup1 = (50,);

## 遍历字典



## Python 函数

函数是组织好的，可重复使用的，用来实现单一，或相关联功能的代码段。

函数能提高应用的模块性，和代码的重复利用率。你已经知道Python提供了许多内建函数，比如print()。但你也可以自己创建函数，这被叫做用户自定义函数。

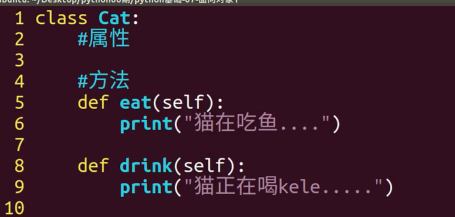
# Day07

# Day08

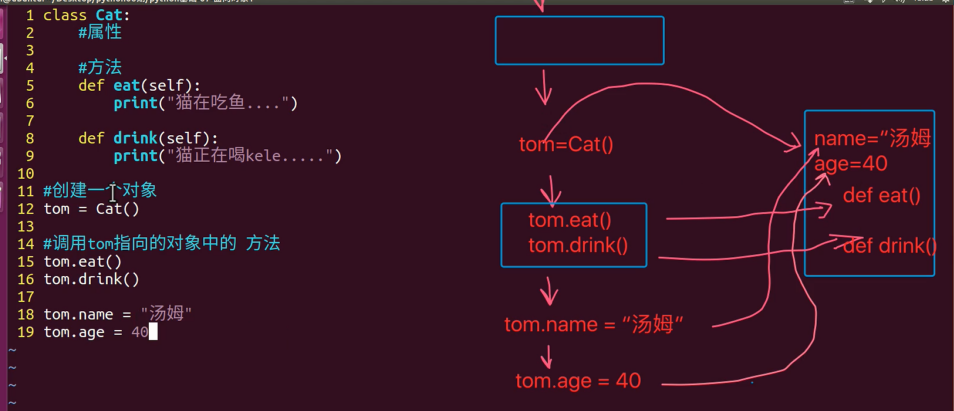
## 文件的读写

# Day09

## 定义类

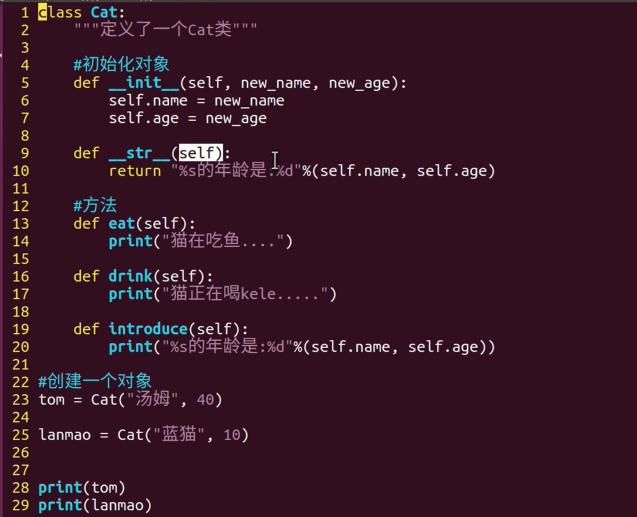


## 创建对象，向对象中添加属性

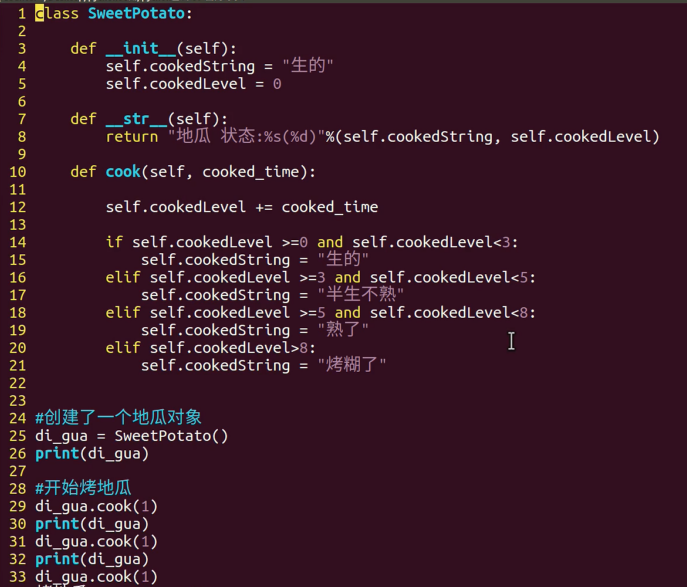


对象创建完成之后，还可以向对象中添加属性

## 构造方法，toString方法



## 案例：烤地瓜



# Day10

## 继承,多态,重写

# Day11

## 异常

# Day13 day14

## 游戏飞机大战