# Shell编程

计算机语言分类：机器语言，汇编语言，高级语言

编译型：代码—编译器—二进制—计算机运算

解释型：代码—解释器—编译器—二进制—计算机运算

编译型开发语言：1.c 2.java 3.c# 4.c++

解释型开发语言：1.python 2.php 3.rudy 4.shell 5.vb

Shell（壳程序）是linux系统内核与用户交互的一种媒介

分类：bash（默认），csh，dash，tsh，ksh

Shell可以将命令编译成计算机识别的语言

脚本：就是一个文件，将shell命令和控制语句写到一个文件中运行脚本方式：1.chmod a+x 文件名 ./文件名 执行脚本

2.source 路径文件名

#！/bin/bash：说明此脚本使用bash

Echo “hello” 文件名默认以.sh结尾

#作用：注释 不运行这行命令

变量名可以由字母、下划线、数字组成，但是不能以数字开头，又不能出现特殊字符

变量的分类：1.局部变量只能在当前shell中使用

2.全局变量：在当前shell以及子shell中使用

区分局部变量还是全局变量

Env：查看环境变量

Export：局部变量名=变量值 将局部变量转换成全部变量

Unset 变量名：取消变量

数组

数组：一组元素的集合，可以是中文，英文，数字，符号

数组名=（元素1 元素2 元素3…）例aa=（1 5 7）

下标0 1 2 ：0代表第一个元素

Echo ${数组名【下标号】}：提取数组中的某个元素

Echo ${数组名【@】} ：显示所有元素

Echo ${#数组名【@】} ：统计有多少个元素

Unset 数组名【下标号】 删除数组中某个元素

Echo ${数组名【@】：数字} ：这个数字位置本身和之前的元素不显示，显示后面的（切片）

控制语句分类：

判断语句if：单分支和多分支

循环语句for/while ：单循环和多循环

关系运算符

-gt（大于） -ge（大于等于） -eq（等于） -lt（小于） -le（小于等于） -ne（不等于）

算术运算符

+ - \* / %（求余运算）

1. a=`expr 1 + 1` 注：反引号的作用：将一个命令结果赋予一个变量

（b=`expr 2 \\* 3`） 双引号/单引号的作用将一个字符串赋予一个变量时，用双引号/单引号引用起来

1. echo $（（1+1）） c=$（（1+1）） echo $c
2. let e=1+5 echo $e

逻辑运算符：-a ：与 -0 ：或 ！：非

字符运算符：=：判断前后两个字符串是否相等

=！：判断前后两个字符串是否不相等

文件判断运算符

-f 文件名：判断是否为普通文件

-d 文件名：判断是否为目录

-r 文件名：判断文件是否可读

-w 文件名:判断文件是否可写

-x 文件名：判断文件是否可执行

命令----read 【参数】 变量名 ：从键盘上读取值赋予此变量

参数： -p “显示的内容” ：提示内容

-t 时间：在指定的时间内输入值

-s ：隐藏输入的内容

循环语句

For 语句

For 变量名 in 取值列表

Do

语句块

Done

Seq 起始位 步进 终止位

Useradd user$i

Echo user$i | passwd user$i –stdin

$? 的返回值如果是0，则表示前一个命令执行成功，如果返回值非0则表示前一个命令执行失败

Ping -c 2 192.168.0.$i >& /dev/null 将ping的过程放到黑洞文件里 >& 表示无论命令是否执行错误都将被导走

For ((变量初始值；条件表达式；变量变化式))

Do

语句块

Done

变量初始值：只在第一轮循环前执行一次

条件表达式：每轮循环前执行

变量变化式：每轮循环后执行

\n换行 echo 换行 echo -n 不换行 …e…\t：制表符

Break：用在循环语句中终止循环的作用遇到break就不往后执行

Continue：用在循环语句中终止本次循环，继续下一次循环

排序:1.冒泡法排序 2.选择法排序

While 语句

While 【条件】

Do

语句块

Done

参数传递：把脚本外的参数引用到脚本中使用

./脚本名称 参数1 参数2 …

#！/bin/bash

Echo $0 显示该文件名

Echo $1 第一个参数

Echo $2 第二个参数

Echo $# 参数个数

Echo $@ 显示参数

函数：可以重复使用的一段代码

作用：简化代码重复利用

函数的调用方法

1.函数名 2.函数名 参数1 参数2 参数3…

Shell脚本

# Python：

官网：[www.python.org](http://www.python.org)

Windows 下载 Windowsx86-64 executtable installer

Python：解释型的、面向对象的程序设计语言

解释型：需要用解释器来解释的

面向对象：编程思想

基于C语言开发出来的

Python：优点

1.简单、方便 2.开源免费 3.自带的库种类多

学完之后可以做什么：

1.人工智能 2.网站后台程序 3.爬虫 4.自动化测试

Python的解释器：

Cpython：基于c语言开发的解释器

Jpython：基于java开发的解释器

Pypypython：基于python开发的解释器

编辑器：写代码用的 pycharm

#!/usr/bin/python #定义一个解释器

#-\*- coding：utf-8 -\*- #定义编码方式

1.输出、打印 等同于shell的echo

print（‘hello’） print（‘hello’，‘123’，‘我’）

2.输入（从键盘上读取等同于shell的read-p 提示语 a）

a=input（‘请输入密码’） 只有字符是变量名时可以不用加引号，其余必须加引号

变量名=input（提示语）

变量（a=3）

定义变量的方式：变量名=值 例如：a=3，b=4

打印变量的时候：print（变量名）

多变量赋值：a,b,b,d,=1,2,3,4

变量的命名规则：

1.只能是字母、数字、下划线组成

2.不能以数字开头

3.不能命名成语言的内部命令（例：print=3）

3.数据类型：

1.str 字符串 2.int 整数 3.float 浮点数 4.none 空值 5.list 列表 6. tuple 元组 7.dict 字典 8.set 集合 9.bool 布尔值

Python的函数：type（）查看数据类型

例如：a=123 print（type（a））

布尔值：true（是）和false（否）

空值：none 代表的是一个特殊的值

字符串：一串字符的集合 例如：a=‘asbf1234de’

特点：1.不可变得（不能更改的）

2.支持索引（支持通过下标位置取字符串中的数据）

3.支持切片（从一串字符串取一段数据）

索引：print（a[1]）

S

反索引：print（a[-1]）

B

切片：print（a[4:8]）

1234

切片时只有一个数字和冒号，带有一个冒号：冒号在前默认为从开始到结束值；冒号在后默认为从开始值到最后

切片时有一个数字和两个冒号：冒号在后表示从起始值取到最后；冒号在前以数字的个数分组，每次取每一组的第一个（数字为负数时，则是倒着显示）

切片时有2个数字和两个冒号时，第一个数字代表每一组内的下标位置

属于字符串的函数：

1.将所有的小写变成大写 upper（）

b=a.upper（） #a是一个字符串

print（b）

ASBF1234DE

2.替换，将字符串中的某些字符替换为其他字符

b=a.replace（‘b’，‘g’） 将b替换成g

print（b） 某些字符一定是紧挨着的

asgf1234de

（‘b’，‘’） #表示把b变为空的

（‘b’，‘g’，‘1’） 后面的这个数字表示替换多少个b

3.分割，以某些字符为分割符将字符串分割成列表

a=‘asbf1234de’

b=a.split（‘bf’）

print（b）

[‘as’，‘1234de’]

4.去除左右的空格 strip（） 去除左边空格 lstrip（） 去除右边的空格 rstrip（）

a=‘ asbf1234de ’

b=a.strip（）

print（b）

asbf1234de

删除中间的空格可以 b=a.replace（‘ ’，‘’）

5.判断字符串是否以某些字符开头

a=‘asbf1234de’

b=a.startswitch（‘asb’）

print（b）

true或者 false

以某些字符结尾

a=‘asbf1234de’

b=a.endswitch（‘de’）

print（b）

true 或者false

6.连接，以某个字符串为连接符，将列表变为字符串 join（）

a=‘asbf1234de’

b=a.split（‘df’）

c=‘-’.join（b）

print（c）

as-1234de

a=‘asbf’，‘f1234’

b=‘’.join（a）

print（a）

asbff1234

7.格式化字符串

#填充字符串

b=2

a=‘user{}’.format（b） #占位符{}，填充位置

print（a）

user2

a=‘{}，{}，{}，{}’

b=a.format（‘我’，‘爱’，‘我’，‘家’）

print（b）

我，爱，我，家

a=‘user’

b=‘qwe’

print（a+b）

userqwe

a=‘user{sex}123{name}’

b=a.format（name=‘qwe’，sex=123）

print（b）

user123qwe

占位%字符，%填充

a=‘hello%s，我今年%d岁’

b=a%（90,100）

print（b）

hello 90，我今年100岁

%s填充任意数据 ，%d只能填充数字

8.index（）获取字符串中某个字符的下标位置

a=‘qewwre’

b=a.index（‘w’）

print（b）

2

9.count（）统计字符串中某个字符的个数

a=‘qewwre’

b=a.count（‘w’）

print（b）

2

Len （a）统计字符串中有多少个字符

Print（len（a））

1.列表: 一组元素的集合（相当于shell里的数组）

格式：中括号为标识，元素之间以逗号分割（元素数据类型不限）

变量名=[1，‘我’] 变量名=list（值）只能一个字 不能是整数

特点:支持索引（单个取值），支持切片（多个取值），可变的

.向列表添加一个元素 append（）只能添加一个元素，默认在最后一个

.添加指定位置 insert（索引，值）

.删除 remove（值）

.删除指定位置pop（索引）

.排序（升序）数据类型 sort（） 、sorted（列表名） 倒序需要先升序再反转

字符串排序是以首字母在ASCII吗中位置的排序

.反转 abc.reverse（） 更改：重新赋值a[1]=13

.统计列表中某个元素有多少个 bowen.count（）

.将两个列表合为一起 extend（） a.extend（b）

.清空列表 a.dear（）将a列表清空

.浅复制 copy（）只能复制第一层数据，第二层数据时共用的

.深复制 import copy ff=copy.deep copy（c）

2.元组：一组数据的集合，以小括号为标识，元素与元素之间用逗号隔开

特点：不可变的、支持索引、支持切片

index() 获取下标位置 count（）统计某元素的个数

3.字典dict：是一组数据的集合，以大括号为标识，数据都是以键值对的方式存储

格式：{键：值，键：值} 键必须是唯一的

特点：可变的 无序的 不支持索引和切片

A={‘name’：‘小明’}，‘age’：‘12’}

增加更改通过新的键值对去添加 a【‘sex’】=‘nan’

删除pop（）删除某个键值对， a.pop（‘name’）

默认删除最后一个键值对 a.popitem（）

获取所有键 b=a.keys（） 获取所有值b=a.values（）

获取所有键值对 b=a.items（）

将b中所有键值对更新到a中 a.update（b）

4.集合 set ：一组数据的集合，以大括号为标识，格式：{数据，数据}

特点：可变的； 不允许重复； 无序； 不支持索引和切片

增加 add（） a.add（‘er’） 每次只能添加一个数据，这个数据不可变的

删除 remove（） 删除指定数据 pop（）随机删除第一个

更新 update（） 将一个集合更新到另集合中

并集 | 交集& 差集-

a=str（） 重新定义数据类型

5.运算符：

算术运算符：+，-，\*，/，//，%，\*\*

比较运算符：>,<,>=,<=,==,!=

赋值运算符：+=，-=，\*=，/=，//=，%=，\*\*=

成员运算符：in not in

逻辑运算符：and or not

6.判断语句：

单分支判断语句格式：if 条件： 执行语句

跟shell区别：不需要then和fi 执行语句必须缩进 条件后面必须加：号

input（）函数，手动输入的任何数据都是字符串

双分支判断语句：

格式：if 条件： 执行语句 else：执行语句

多分支判断语句

格式：if 条件： 执行语句 elif 条件： 执行语句

嵌套判断 判断语句中在写入判断语句 满足跳跳1之后再满足条件2

循环语句 for while

格式：for变量名 in 范围： 执行语句

1.执行语句必须缩进 2.条件后必须加冒号

1.范围：可以写所有带有下标位置的数据类型

range()可以将数字编程一个范围

只有一个数字时默认从0开始到数字结束

只有两个数字时，第一个数字是开始值，第二个数字是结束值

有三个数字时第一个是起始值，第二个是结束值，第三个是递进值

嵌套循环 for语句中加入if语句或for语句

Break：终止循环 continue：结束本次循环 ，继续下一次

for …else…语句

只要循环没有被break掉，就执行else语句

不换行print（i，end=‘’）

取出列表中的值并取其下标值enumerate（a）只想取其下标位print（i【0】）

只要函数有多个结果，就可以有多个变量来接收

for j，I in enumerate(a): j,I =下标位置，值

While格式：while 条件：执行语句

While根据是否符合条件来决定是否循环

需要代码本身来控制循环的次数

适用的场景：for循环用于循环带有下标位置的循环

While循环用于根据某种条件进行的循环或者是无限循环

While…else… 语句

只要循环没有被break掉就执行else语句

列表推导式：将语句块直接写在列表中，让其产生的结果直接放到列表中b=【I for I in range（6） i>3】只添加有4,5如果只修改结果，只需要修改结果语句不要动，前面i为结果位置，后边是语句

对文件的操作：代码实现对文件的增删改查

Open（） open（‘文件名’，‘权限’，‘编码方式’）

F=open（‘a.txt’，‘w’，encoding=‘utf-8’）

打开了a.txt文件 当前路径下的文件 \t ,\n,\r转义字符

\n换行，write（）不换行

权限：w写入，a追加，r读取 w+，r+，a+可读可写 wb，rb，ab以二进制的读取和写入 读取出来的类型是字节wb用于图片，音频，视频的保存

写入f.write（‘字符串’） 关闭文件f.close（）

如果没有这个文件，先创建文件在写入，有这个文件直接操作

b=f.read（） print（b） 读取文件中所有的内容，结果是个字符串

b=f.readlines（）读取文件中所有内容，结果是个列表，文件的每一行，是列表中的每个元素

b=f.readline（）每次读取文件中的一行内容有多少个readline就读取多少行

with语句 上下文管理器 可以自动释放占用的资源，自动关闭文件

格式：with 要操作的对象 as 变量名：执行语句

With open（‘a.txt’，‘r’，encoding=‘utf-8’） as f：

不需要在写f.close（） 内部有-enter-进入 -exit-退出

报错的内容：第一行：报错的位置，第二行报错的语句，第三行：报错的类型和描述

1. 函数：具有可重复使用的某种功能的代码块

关于字符串，元组，type，len，sum等叫做内置函数

自定义函数：自己写得函数

函数的格式：def 函数名（）：语句块

调用函数：函数名（）

函数名的命名规则：只能以字母数字下划线组成 不能以数字开头 不能命名成语言内部命令 函数命名成跟功能相近的英文

1. 变量的作用域

分类：局部变量，全局变量

局部变量：只作用于函数的内部

全局变量：可以作用于整个文件

global 变量名 将局部变量变成全局变量 写在变量名上边

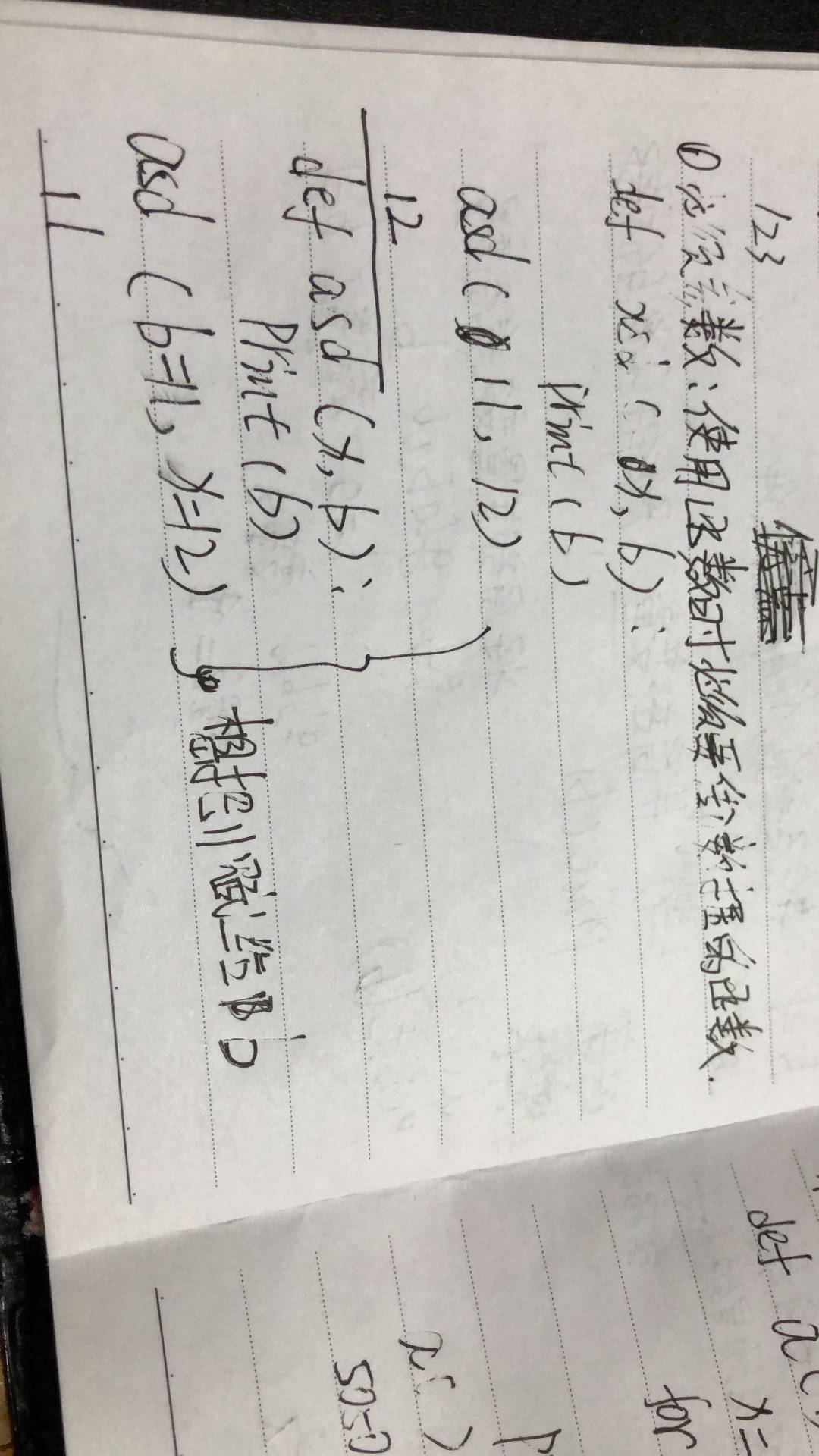
1. 传参

定义函数：def 函数名（参数名）：执行语句块 参数名等于变量名

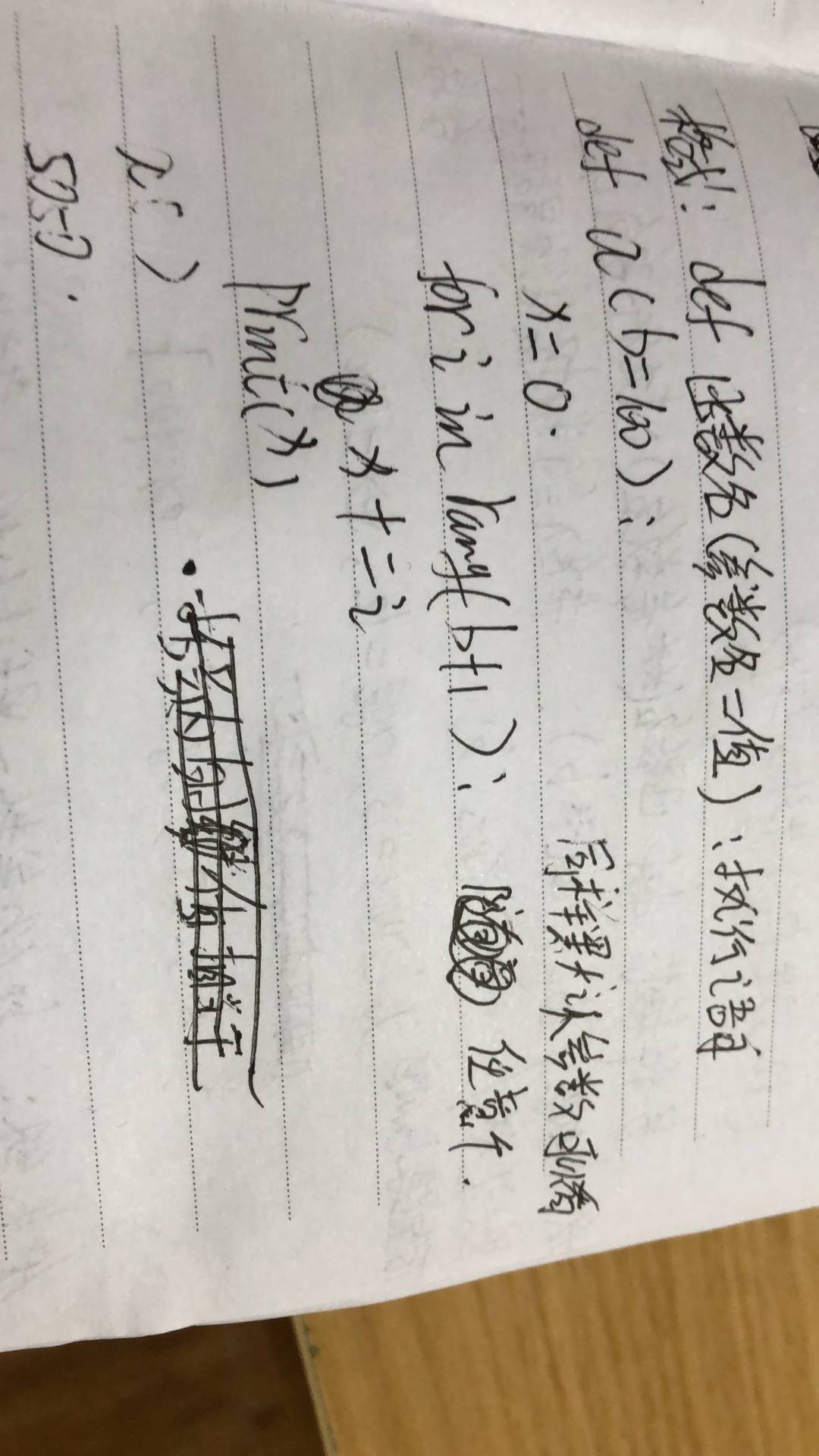
参数的个数是任意的

必须参数：使用函数是必须要传入数据的参数

参数不仅可以是数字，还可以是其他数据类型



默认参数：使用函数时可以传入数据也可以不传入数据



可变长参数:

特点：1.使用的时候可以传入数据也可以不传入数据

2.使用函数时可以传入多个数据

格式：def 函数名（\*参数名）：执行语句块

#将这些数据放在一个元组中

另一种格式：def 函数名（\*\*参数名）：执行语句块

#传入的数据必须是键值对的格式

4.return

作用：1.结束函数

2.赋值（可以赋任意个值，多变量接收）

5.lambda 匿名函数

def 都是用来写函数

lambda :只是一个简单的表达式，只能做一些逻辑结构简单

格式：函数名=lambda 参数 ：表达式

导入语句：

import 语句

1.import 文件名

import day1 #将day1文件中的代码导入到当前文件中

import random 产生随机数

day1.asd（） #调用day1中的函数 文件.函数名（）

2. from 文件名 import 函数名 #后面也可以跟文件名

from day1 import \* 导入day1的全部函数

from day1 import asd，qwe #导入多个函数的时候用逗号隔开

判断正在执行的文件是否为本文件

if \_\_name\_\_==’\_\_\_main\_\_’： #下划线是两个下划线

在python中类似于day1这样的文件称作模块

模块的分类：1.python