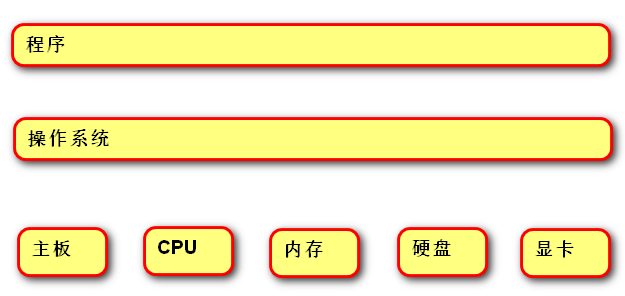
# Python 简介



计算机大致构成

1. 编程：编写程序。
2. 程序(软件)：一组由计算机执行的指令(代码)
3. 操作系统：隔离程序与硬件的差异，使程序开发简单化。

Windows/Linux/Unix

## 定义

是一个免费，开源，跨平台，动态，面向对象的编程语言。

## Python程序执行方式

### 交互式

定义：在命令行输入指令，回车即可得到结果。

步骤：

1. 打开终端
2. 输入：python3
3. 输入: print(“Hello World”)
4. 退出：exit()

### 文件式

定义：将指令编写到.py文件中，可以重复运行程序。

步骤：

1. 创建文件：

/home/tarena/1902/month01/day01/hello.py

1. 打开终端进入指定目录

cd /1902/month01/day01

1. 运行python程序

python3 hello.py

## Linux常用命令

pwd: 查看当前工作路径

cd: 进入目录

ls:查看当前目录中的文件

…

练习1：创建python程序，输出你好，世界！

通过文件式执行。

## 程序执行过程

由源代码转换为机器码的过程，分为编译和解释。

编译：在程序执行之前，通过编译器将源代码转换为机器码。

-- 特点：运行时，计算机可以直接执行。例如：c语言

-- 优势：运行速度快。

-- 缺点：不能跨平台，开发效率低。

解释：在程序执行中，通过解释器将源代码转换为机器码。

-- 特点：运行时逐语句解释执行。

-- 优点：可以跨平台，开发效率高。例如：JavaScript

-- 缺点：运行效率低。

python执行过程：

源代码 –编译--> 字节码(特定python的表现形式.pyc) –解释--> 机器码

|—————————1次—————————————|

## Python 版本

python 2.7（2020年结束维护）

python 3.5（教学版本）

python 3.8（最新）

## Python 解释器类型

CPython(c语言开发)

Jython(Java开发)

IronPython(.Net开发)

# 数据基本运算

## pycharm 快捷键

# 格式化代码 ctrl+alt+L

# 运行 shift+ctrl+f10

# 复制行 ctrl+d

# 注释 ctrl+/

## 注释

定义：给人看的，描述代码的文本。

-- 单行注释： # 内容

-- 多行注释：

“””

内容

“””

## 函数

表示一个功能，制作函数的人叫做函数定义者，使用函数的人叫做函数调用者。

print(数据):将括号里面的数据，显示到控制台中。

变量 = input("提示信息"):将用户在控制台中输入的内容赋值给变量

day01作业：

练习1：在控制台中录入学生信息(姓名,年龄,性别,成绩。。。)

在控制中显示学生信息。

格式：xx的年龄是xx，性别是xx，成绩是xx。

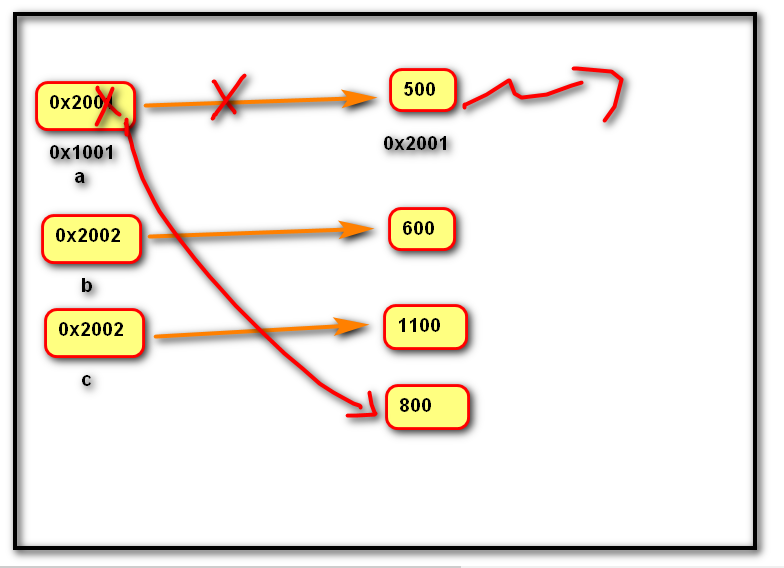
代码：day02/day01\_exercise/exercise01.py

练习2：看看教程 [www.runoob.com](http://www.runoob.com)

练习3：逛逛社区 [www.pythontab.com](http://www.pythontab.com)

## 变量

代码：day01/code01



1. 定义：关联一个对象的标识。

2. 命名：

-- 必须为字母数或下划线，后面跟着字母，数字，下划线。

-- 不能使用保留关键字

3. 建议命名：见名知意

字母小写，多个单词用下划线隔开。

例如： class\_name student\_name

1. 语法：

变量名 = 表达式

变量名1 = 变量名2 = 表达式

变量名1，变量名2 =表达式1，表达式2

练习1:画出下列代码内存图

a =”我”

b = “爱”

a =”python”

d = a + b

练习2：在控制台中依次获取两个变量，然后交换变量，

输出结果。

例如：请输入第一个变量：100

请输入第二个变量：200

….

第一个变量是：200

第二个变量是：100

代码：day02/ exercise01

## 数据类型

代码：day01/code02

变量没有类型，关联的对象有类型。

可以通过type方法，判断对象的类型。

语法：变量 = type(数据)

### None

表示不存在的特殊对象

作用:用来占位 a = None

变量解除绑定

### 整形int

整数。包含正数，负数，0.

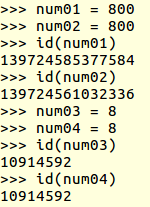
字面值：

--- 十进制：100，-5, 0

--- 二进制(0,1)：0b0,0b1,0b10,0b11, 0b100

--- 八进制(0-7)：0o0,0o7,0o10

--- 十六进制(0-9,a-f)：0x0,0xa



小整数对象池：CPython中整数-5 到 256，永远存在小整数对象池中，不会释放。

对象池：缓存的机制。

### 浮点型float

小数，包含正数，负数，0.0

字面值：

--- 小数：1.5

--- 科学计数法：1.5e1 0.5e-1

### 字符串str

用于记录文本信息。

字面值：单引号 双引号 三引号

### 布尔bool

True 表示真(成立)

False 表示假(假的)

### 复数complex

由实部和虚部组成的数字。

虚部以j或J结尾

字面值：

5 + 3j 5j

## 数据类型转换

转换为整形：int(数据)

转换为浮点型：float(数据)

转换为字符串：str(数据)

转换为布尔：bool(数据)

False : 0 ，0.0 ， “” ,None

True: 有值

## 运算符

### 算数运算符

代码：day02/code03

+

-

\*

/ 小数除(结果为float类型)

// 底板除(向下取整)

% 取余数

\*\* 幂运算

### 增强运算符

代码：day02/code04.py

x += 1 等同于 x = x + 1

-=

\*=

/=

//=

%=

\*\*=

在变量基础上进行操作

### 比较运算符

>

<

== 等于

>=

<=

!= 不等于

结果是bool类型

练习1：在控制台中依次获取小时，分钟，秒，

计算总秒数。

代码：day02/exercise02.py

练习2：在控制台中获取一个4位整数，

计算每位相加和。

例如：1234

1 + 2+3+4

代码：day02/exercise03.py

### 逻辑运算符

#### 与 and

表达并且，一假俱假。

re = True and True # True

re = False and True # False

re = True and False # False

re = False and False # False

#### 或 or

表达或者，一真则真

re = True or True # True

re = False or True # True

re = True or False # True

re = False or False # False

#### 取反 not

re = True

re = not re

print(re) # False

#### 作用

比较两个bool值关系。

例如：

限制飞机水平范围

如果 到了最左边 and 还想想左移动 or 到了最右边 and 还想向右移动

停

练习：在控制台中输入一个年份

判断是否为闰年

条件1：年份能被4整除，但是不能被100整除

条件2：年份能被400整除

代码：day02/exercise04.py

#### 短路逻辑

如果ａｎｄ前面的表达式结果为false，后面的表达式不再执行

建议：将耗时的判断尽量放在后面

re= input("请输入a:") == "a" and input("请输入b:") == "b"

如果or前面的表达式结果为True，后面的表达式不再执行

建议：将耗时的判断尽量放在后面

re = input("请输入a:") == "a" or input("请输入b:") == "b"

### 身份运算符

代码：day02/code06.py

is ： 判断两个对象是否为一个对象

is not：与is相反

语法：

变量 = 变量1 is 变量2

结果是bool类型

### 优先级

从高到低：

算数运算符

比较运算符

增强运算符

逻辑运算符

小括号最高

（1 + 2） \* 5

## 调试步骤

目的：

1. 理解程序执行过程
2. 排除逻辑错误(不是Error)

步骤：

1. 加断点(程序运行到本行停止，没有执行本行)
2. 开始调试shift+alt+f9
3. 逐行执行 F7
4. 停止调试ctrl + f2

错误信息

File "/home/tarena/1902/month01/day02/exercise03.py", line 10, in <module>

unit04 = number // 1000 + "a"

TypeError: unsupported operand type(s) for +: 'int' and 'str'

File "文件路径", line 行号, in <module>

错误代码

XXXError: 错误描述

day02 作业：

1. 使用二进制输出1 3 5

八进制 1 8 10

十六进制 1 13 20

代码：day03/day02\_exercise/ exercise01

1. 在控制台中获取圆形的半径，

输出面积(3.14 \* r 的平方)与周长(2 \* 3.14 \* r)

代码：day03/day02\_exercise/ exercise02

1. 在控制台中输入总秒数，计算几小时零几分钟零几秒钟。

代码：day03/day02\_exercise/ exercise03

1. 读书：Python编程：从入门到实践 第二章

## 内置函数

round(数据，精度)：四舍五入，精度如果省略，则默认为0.

## del 语句

1. 作用：del 语句可以删除变量
2. 语法：del 变量名1，变量名2
3. 对象引用计数：当对象绑定给变量时，计数增加1，当变量解除绑定时，计数减少1.待计数为0时，对象自动释放。
4. 实例：

a = 800

del a

此时变量a被删除，对象800被释放

a = 800

b = a

del a

此时变量a被删除, 对象800不会释放。

del b

此时变量b被删除, 对象800被释放。

练习：画出下列代码内存图

num01 = 500

num02 = 500

num03 = num01

num01 = None

del num03

# 语句

## 行

1. 物理行：程序员编写代码时的行。
2. 逻辑行：python解释器需要执行的指令。

建议一个物理行对应一个逻辑行。

1. 示例：

print(“ok”); print(“no”)

一个物理行，两个逻辑行。

1. 隐式换行：所有括号的中的内容都可以换行。

x = 1 + (2+ 3+

4)

1. 显示换行：通过折行符 \ 换行。

x = 1 + 2+ 3 \

+ 4

1. 缩进：4个空格

## pass 语句

通常用来填充语法空白。

if xxxx：

pass

else:

yyy

## 选择语句

### if else

1. 作用：让程序根据条件有选择性的执行语句。
2. 语法：

if 真值表达式:

满足条件执行的语句块

else:

不满足条件执行的语句块

if 真值表达式1:

满足条件1执行的语句块

elif 真值表达式2:

满足条件2执行的语句块

else:

以上条件都不满足执行的语句块

1. 说明

elif 字句可以有0个或者多个。

else 字句最多有1个，只能放在最后。

练习1：在控制台中获取一个整数，

如果是整数则打印”正数”

如果是负数则打印”负数”

如果是零则打印”零”

代码：day03/exercise01.py

练习2：在控制台中获取一个月份，

　　打印春(１--3)　夏(4--6)　秋(7--9)　冬(10--12)

代码：day03/exercise02.py

练习3：输入月份，返回天数

2 月 28天

1　３　５　７　８　１０　１２　月 ３１天

4 6 9 11 月　３０天

代码：day03/exercise03.py

### if语句的真值表达式

if 变量: # 相当于 if bool(变量)

xxx

else:

yyy

练习1：在可控制台中获取一个整数

打印奇数或者偶数

代码：day03/exercise04.py

练习2： 判断闰年，如果是闰年输出29天，

否则输出28天。

代码：day03/exercise05.py

### 条件表达式

代码：day03/exercise05.py

1. 作用：可以选择性的为变量进行赋值
2. 语法：

变量 = 满足条件的结果 if 条件 else 不满足条件的结果

1. 示例：

y = "yes" if False else "no"

print(y) # no

## 循环语句

### while 语句

1. 作用：满足条件，重复执行语句。
2. 语法：

while 真值表达式:

满足条件执行的循环体

else:

条件不满足执行的语句

1. 说明：

-- 先判断条件，如果满足则执行循环体，如果不满足则执行一次else子句。

-- else 最多只有1个。

练习１：在控制台中获取两个整数，作为循环开始和结束的点。

代码：day03/exercise06.py

练习2：一张纸的厚度是0.01毫米

请问对折多次，可以超过珠穆朗玛峰8844.43米。

代码：day03/exercise07.py

练习３：猜数字游戏1.0 版本

让用户在控制台中重复猜测。如果没有猜对，提示：大了，小了。

如果猜对了，提示正确，并显示猜了多少次。

猜数字游戏２.0 版本

最多只能猜６次。

代码：day03/exercise08.py

### for 语句

代码：day04/code01

1. 作用：用来遍历可迭代对象的元素。

-- 可迭代对象：能够依次获取数据元素的对象。

例如：字符串 ”abcded”

1. 语法：

for 变量列表 in 可迭代对象:

循环体

else:

循环过后执行的语句

1. 说明：

- - else子句可以省略

-- 循环体中有break语句，退出时else不执行。

#### range函数

1. 作用：可以生成一系列整数的可迭代对象。
2. 语法：range(开始点，结束点，间隔)
3. 说明：

-- 返回的数字不包含结束点。

-- 间隔默认值为1

-- 开始点默认为0

4. 示例：

range(3,6) # 3 4 5

range(4) #0 1 2 3

range(1,10,2) #1 3 5 7 9

range(5,0,-2) # 5 3 1

range(4,0,1) # 空

代码：day04/code01

## 跳转语句

### break语句

1. 跳出循环体，后面的代码不再执行。
2. 可以让while语句的else部分不执行。

作业1：输入一个季度，打印该季度的月份。

作业2(扩展)：

一个球从100m的高度落下，

每次弹回原高度的一半。

计算：1. 总共经过？次最终落地(可以弹起的最小高度0.01m)。

1. 记录总共经过？米。

作业3：看书：python从入门到实践

作业4：添加微信公众号

Python爱好者社区

Python 知识圈

Python编程

Python 开发者

CSDN

### continue 语句

跳过本次循环，继续下次循环。

练习：在控制台中输入一个整数，判断是否为素数。

规则：只能被1和自身整除。

2 3 5 7 11 ….

提示：排除法

判断x 与该范围（2 -- x – 1）之间的数字

满足条件的不是素数：x % 数字 == 0

例如：9

3 4 5 6 7 8

代码：day04/exercise01

# 容器

## 通用操作

字符串：day04/code04

### 数学运算符

+：拼接两个容器

\*：重复生成容器元素

+=：用原容器与右侧容器拼接，并重新绑定变量

s = “abc”

s+=”D”

\*=：用原容器生成重复元素，并重新绑定变量

< > >= <= == !=：依次比较两个容器中元素，一但不同则返回结果。

“abc”

“aaf”

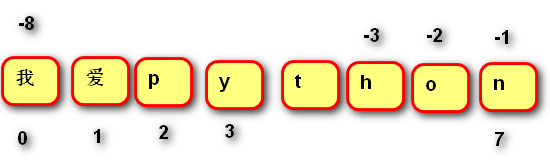
### 成员运算

语法：数据 in 容器

数据 not in 容器

作用：判断容器中是否包含指定数据，返回bool类型。

### 索引



1. 作用：访问容器元素
2. 语法：容器[整数]
3. 说明：

-- 正向索引从0开始，最后一个是len(字符串) -1

-- 反向索引从-1开始，-1代表最后一个，第一个是 - len(字符串)

### 切片

1. 作用：从容器中取出相应的元素重新组成一个容器。
2. 语法：容器[开始索引:结束索引:步长]
3. 说明：

-- 结束索引不包含该位置元素

-- 步长是切片每次获取完当前元素后移动的偏移量

-- 省略开始索引，默认为0.

-- 省略结束索引，默认最后。

### 内建函数

len(容器) 返回容器长度

sum(容器) 返回容器所有元素的累加和（元素必须是数值）

max(容器) 返回容器中最大的元素

min(容器) 返回容器中最小的元素

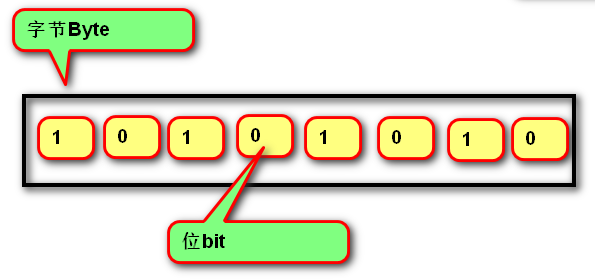
## 字符串str

### 定义

有一系列字符组成的不可变序列容器，存储的是字符编码值。

### 编码

代码：day04/code02



字节Byte：计算机存储的最小单位,等于8位bit。

字符：单个的数字，文字，符号。

字符集(码表)：存储字符与二进制序列的对应关系。

编码：将字符转换为对应的二进制序列的过程。

解码：将二进制序列转换为对应的字符的过程。

编码方式：

1. ASCII编码：包含英文、数字等字符。每个字符1个字节。
2. GBK编码：兼容ASCII，包含21003个中文。每个英文1个字节，汉字2个字节。
3. Unicode字符集：国际统一编码，旧字符集每个字符2个字节，新字符集每个字符4字节。
4. UTF – 8编码：Unicode的存储和传输方式。英文1个字节，中文3个字节。

### 相关函数

1. ord(字符串):返回该字符串的Unicode码。
2. chr(Unicode码)：返回对应的字符串。

练习1：在控制台中输入一个字符串，打印该字符串的每个字符编码。

练习2：循环输入编码值，显示字符。待输入负数时退出。

代码：day04/exercise02

### 字面值

代码：day04/code03

#### 单引号和双引号

单引号内的双引号不算结束符

双引号内的单引号不算结束符

#### 三引号

可以包含单双引号

可见即所得，常常作为文档注释

#### 转义符

1. 定义：可以改变原有字符含义的特殊字符
2. 常用的：\’ \” \””” \n

\t 水平制表格 \0 空字符 \\

1. 原始字符：

str02 =r"a\b\c\d.txt"

输出的是 a\b\c\d.txt

#### 字符串格式化

1. 定义：生成一定格式的字符串。
2. 语法：” …%格式….” %(变量)
3. 格式：%[- + 0 宽度.精度]类型码

- ： 左对齐(默认就是右对齐)

+ ：显示正号

0：左侧空白位置补零

宽度：整个数据输出的宽度(字符大小)

精度：保留小数点后多少位

1. 类型码：s 字符串 d 整数 f小数

练习：在控制台中打印时间

120秒显示为：02:00

119 01:59

…..

0 00:00

代码：day04/exercise03

练习1：在控制台中输入一个很长的字符串，

如果该字符串中包含”qtx”，则提示”老师好”.

练习2：输入一个字符串，打印如下内容。

1. 打印字符串第一个字符
2. 打印字符串最后一个字符串
3. 如果长度是奇数，则打印字符串中间的字符串

len(字符串) 返回长度

练习3：在控制台中输入一个整数，打印一个矩形。

例如：4

\*\*\*\*

\* \*

\* \*

\*\*\*\*

提示：\* 乘以 4

中间输出4 - 2空格

练习4：写一个程序，输入一个字符串，把字符串的第一个字符串和最后一个字符去掉。

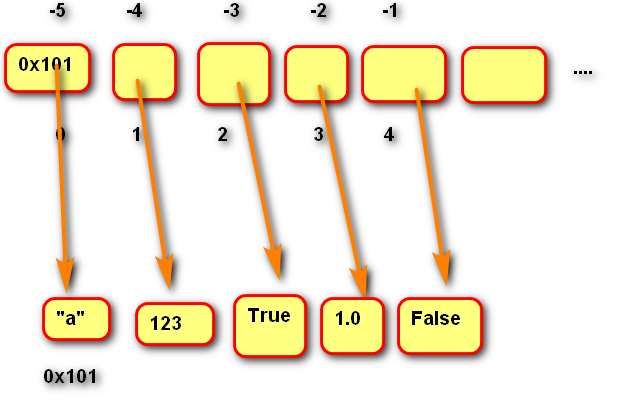
扩展：输入一个字符串，判断是否为回文。

回文是中心对称的文字。

例如：上海自来水来自海上

山西运煤车煤运西山

## 列表list



### 定义

由一系列变量组成的可变序列容器。

### 基础操作

代码：day05/code01

创建：[元素] list(可迭代对象)

增加：insert append

删除：remove del

修改：列表名[索引] = 数值

获取：索引 切片

练习1：体会列表的增删改查。

练习2：在控制台中录入学生成绩

先输入：学生总数

再依次录入成绩。

计算总分

代码：day05/exercise01

练习3：在控制台中循环输入整数，存入列表，当输入-1时退出。

要求：整数不能相同。

计算：最大值、最小值。第二个最大值

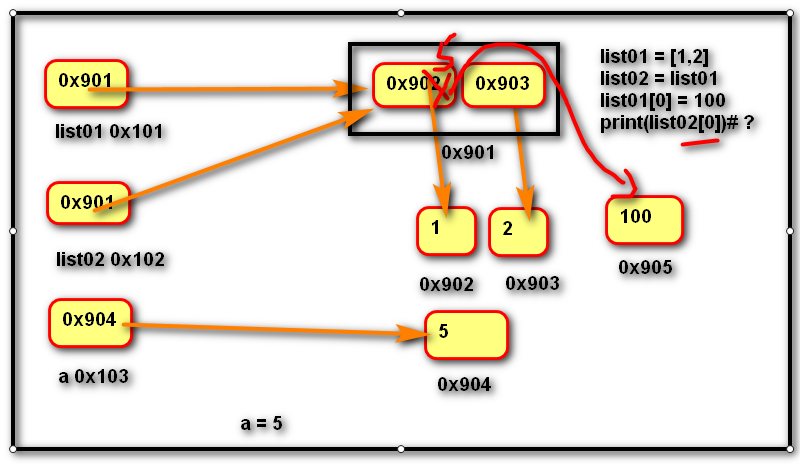
8 3 2 9 5

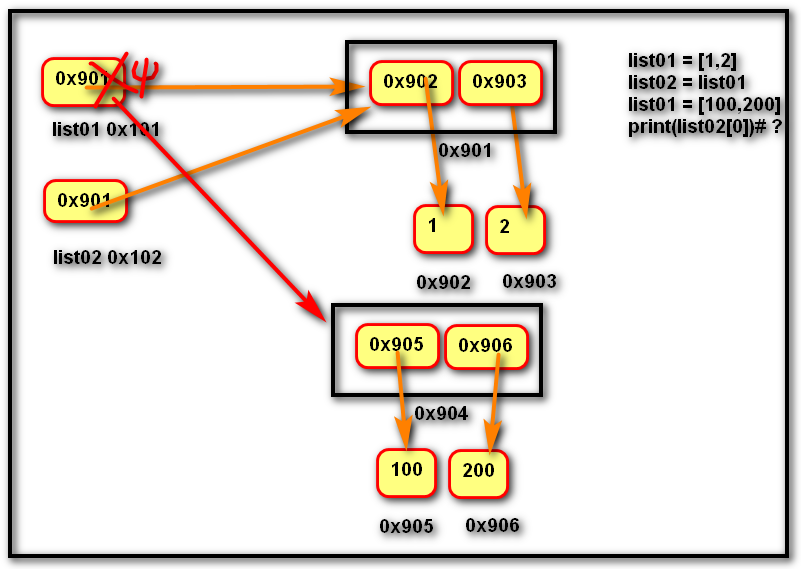
2 3 5 8 9

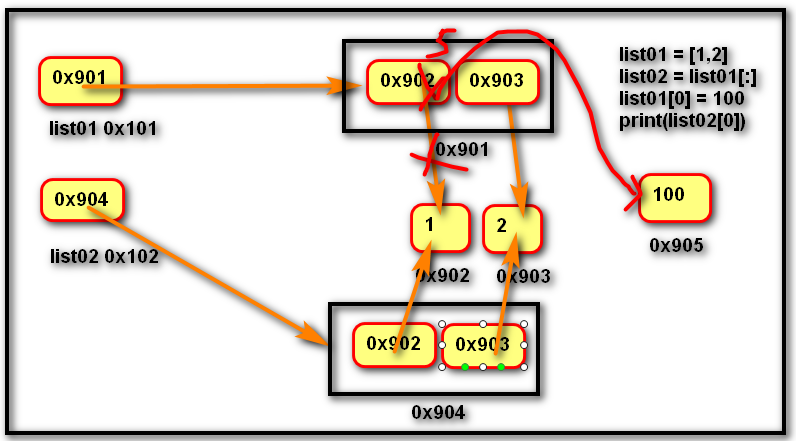
代码：day05/exercise02

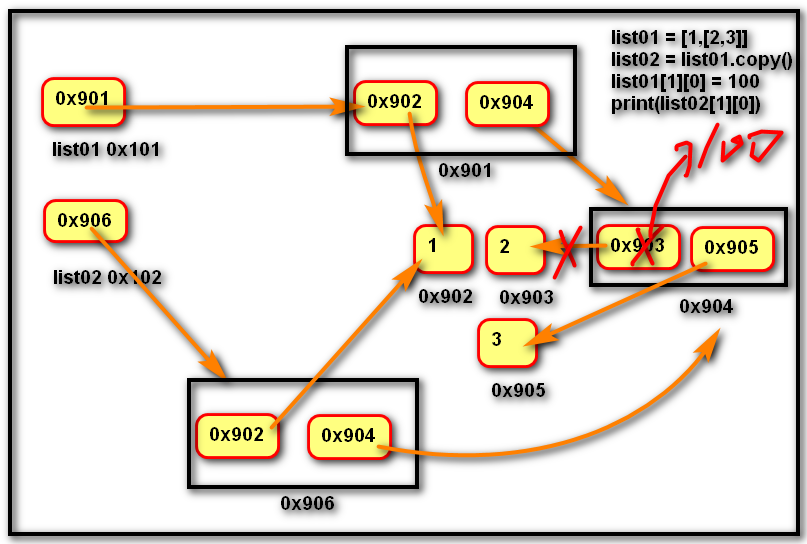
### 内存图

代码：day05/code02







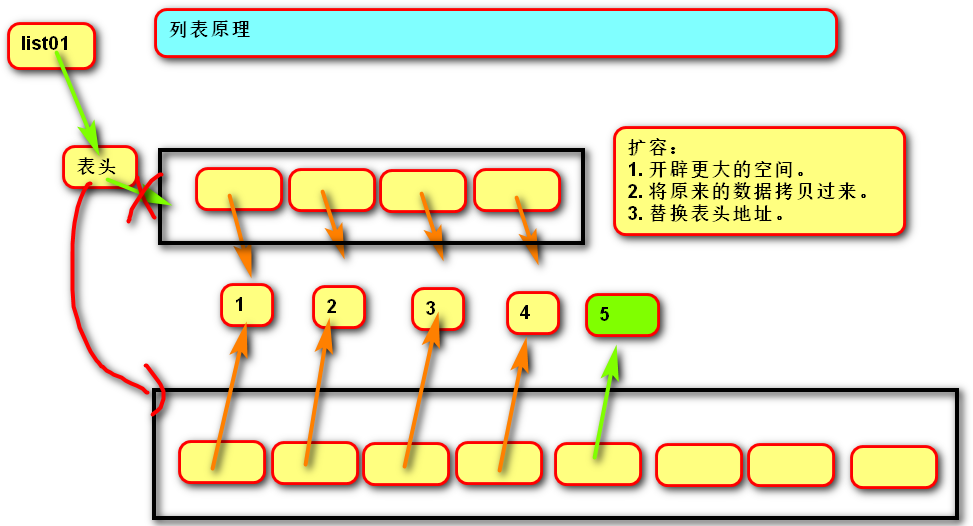


浅拷贝：在复制过程中，只复制一层变量。不会复制深层变量绑定的对象。

深拷贝：复制整个依赖的变量。

(略….)

### 扩容原理图



### 字符串VS列表

代码：day05/code03

1. 列表和字符串都是序列，元素之间有先后顺序。
2. 字符串时不可变的序列，列表可变。
3. 字符串中每个元素存储字符，而列表可以存储任意类型对象。
4. 列表和字符串都属于可迭代对象。
5. 函数：

将多个字符串拼接为一个。

result = "连接符".join(列表)

将一个字符串拆分为多个。

列表 = “a-b-c-d”.split(“分隔符”)

练习1：在控制台中循环输入字符串，待输入q时退出。

显示一个新的字符串。

代码：day05/ exercise03

作业1：单词反转

”How are you” --> “you are How”

代码：day06/ day05\_exercise/ exercise01

作业2：参照字符串与列表文档，自学常用方法。

作业3【扩展】：在控制台中购买彩票

一注彩票：7个球(整数)

6个红球：1 --- 33 【不能重复】

1个蓝球：1 – 16

例如：请输入第1个红球号码：

请输入第2个红球号码：

号码超过范围

已经存在

最后输出彩票：红球号码升序排列

代码：day06/ day05\_exercise/ exercise02

### 列表推导式

代码：day06/code01

1. 语法：

变量 = [表达式 for 变量 in 可迭代对象]

变量 = [表达式 for 变量 in 可迭代对象 if 条件]

1. 作用：使用简易方法，将可迭代对象转生成为列表。
2. 示例：

# 传统生成列表写法

list\_reuslt = []

for item in list01:

list\_reuslt.append(item \*\* 2)

# 使用列表推导式生成列表

list\_reuslt = [item \*\* 2 for item in list01]

#### 推导式嵌套

1. 传统写法：

result = []

for r in ["a", "b", "c"]:

for c in ["A", "B", "C"]:

result.append(r + c)

1. 推导式写法：

result = [r + c for r in list01 for c in list02]

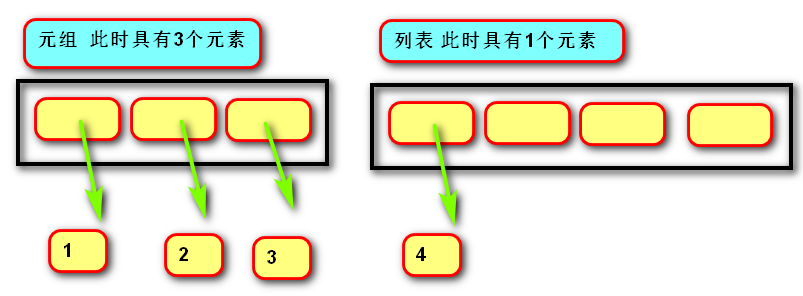
1. 语法：

[表达式 for 变量1 in 可迭代对象1 for 变量2 in可迭代对象2]

## 元组tuple

代码：day06/code02

### 内存图



### 定义

由一系列变量组成的不可变序列容器。

### 作用

元组与列表都可以存储一系列变量。

列表会预留内存空间，所以可以增加。

元组会按需分配内存，建议变量数据固定时使用元组，因为通常占用空间小。

### 基础操作

创建空元组：（） tuple()

创建有元素的元组：（元素） tuple(可迭代对象)

获取元素：索引 切片

练习1：在控制台中输入月，日，计算是这一年的第几天。

代码：day06/exercise02

练习2：输入月份，显示天数。

代码：day06/exercise03

## 字典dict

代码：day06/code03

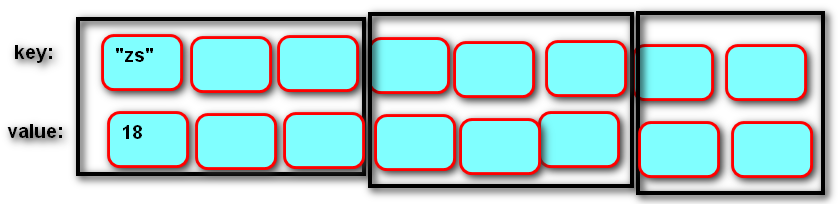
### 定义

由一系列键值对组成的可变映射容器。

映射(哈希算法)：通过键对应值，每条记录无序。

键必须唯一且不可变对象(字符串、数字、元组)，值没有限制。

### 内存图



### 基础操作

创建字典：{键1：值1，键2：值2}

dict(可迭代对象)

添加/修改：

字典[键] = 表达式

说明：键不存在则创建，键存在则修改。

获取记录：

字典[键]

说明：如果不存在键，则错误。

删除：

del 字典[键]

作业1：在控制台获取季度，返回月份。

key value

“春” “1月2月3月”

作业2：在控制台中获取一段文字，

打印这个文字中出现的字符以及次数。

“abcaabcd”

a 3

b 2

c 2

d 1

扩展作业：

游戏： 石头 剪刀 布

在控制台中获取：0 1 2，代表石头剪刀布。

根据游戏规则，显示：平局、胜利、失败。

提示：import random

random.randint(0,2)

将胜利策略存入元组

(

(“石头”,”剪刀”),

(“剪刀,”布”),

(“布”,”石头”)

)

将用户输入的与系统生成的结果

（" ” ，” ”）in 胜利策略

代码：day07/day06\_exercise/ exercise 01

### 字典推导式

代码：day07/code01

语法：{键:值 for 变量 in 可迭代对象 if 条件}

{键:值 for 变量 in 可迭代对象}

## 集合

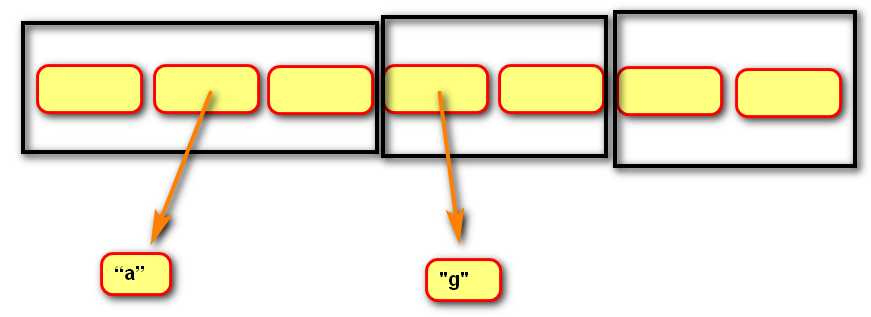
代码：day07/code02

### 定义

由一系列不重复的变量组成的可变映射容器。

相当于只有键没有值的字典。

### 内存图



### 基础操作

创建空集合：set()

创建具有默认值的集合：{元素1,元素2}

set(可迭代对象)

添加：add

删除：discard

### 运算

交集、并集、补集、对称补集、子集、超集

## 固定集合

定义

不可变的集合。

基础操作

创建空集合：frozenset()

创建具有默认值的集合：frozenset (可迭代对象)

运算

与集合相同

练习1：

在控制台中循环输入内容.如果录入空字符串，则退出。

统计输入次数，打印不重复的内容。

“a" "b" "a"

3次 a b a

代码：day07/exercise01

练习2：

经理：[曹操,刘备,孙权]。 技术员：[曹操,刘备,张飞,关羽]。

使用两个列表分别存储经理与技术员。

使用集合计算：

1. 即是经理也是技术员的有谁？

2. 是经理，但不是技术员都有谁？

3. 是技术员，但不是经理都有谁？

4. 张飞是经理吗？

5. 身兼一职的都有谁？

6. 经理和技术员共有几个人？

代码：day07/exercise02

作业1：实现两个列表元素的全排列

[“香蕉”,”苹果”,”哈密瓜”, ”草莓”]

[“牛奶”,”咖啡”,”雪碧”]

作业2：在控制台中录入学生信息name,age,score

将每个学生输出到控制台(一个学生一行)

数据结构：

[

{

“name”:”zs”,

”age”:25,

”score”:100,

},

{

“name”:”ls”,

”age”:35,

”score”:80,

}

]

扩展作业:设计一个算法，判断列表中是否具有相同元素。

[1,4,7,5,1,9,8]

# 函数

代码：day08/code01

## 定义

用于封装一个特定的功能，表示一个行为。

可以重复执行的语句块。

## 作用

提供程序的复用性， 可维护性（代码层次结构更清晰）

## 定义函数

1. 语法：

def 函数名(形参列表):

函数体

1. 说明：

-- def关键字，全称define，意为”定义”。

-- 函数名：建议使用动词，对函数体中语句的描述。

-- 形参列表：形式参数

方法定义者要求调用者提供的信息

-- 函数体：完成该功能的语句。

建议语句不要太多。

1. 函数的第一行语句可以选择性的使用文档字符串存储函数与参数的说明。

## 调用函数

1. 语法：函数名(实参列表)
2. 说明：根据形参传递内容。

练习：定义函数，在控制台打印矩形 。

代码：day08/exercise01

## 返回值

1. 定义：方法定义者告诉调用的结果。
2. 语法：在方法体中，通过return关键字指明结果。

return 结果

1. 说明：

-- 不写return关键字，相当于返回None

-- return 还意味着方法结束。

练习1：定义判断列表是否具有相同元素的函数

代码：day08/exercise02

练习2：定义方法，根据月份返回季度 。

代码：day08/exercise03

练习3：定义函数，判断字符串中，中文的个数。

代码：day08/exercise04

练习4：定义方法，根据月份，返回天数

代码：day08/exercise05

练习5：定义函数，将列表中0元素，移动到末尾。

代码：ProjectMonth01/2048game\_core01

作业1：看电子书函数部分。

作业2：所有课堂练习，做到独立完成。

作业3：玩游戏2048

<http://2048game.com/>

作业4：定义方法,根据成绩判断等级。

0---100 优秀 良好 及格 不及格

扩展作业：定义方法，计算指定范围内的素数。

1. -- 100

## 函数参数

### 实参传递方式argument

代码：day09/code01

#### 位置传参

实参与形参的位置依次对应。

##### 序列传参

实参用\*将序列拆解后与形参的位置依次对应

#### 关键字传参

实参根据形参的名字进行对应。

##### 字典传参

实参用\*\*将字典拆解与形参的名字进行对应。

### 形参传递方式parameter

#### 默认(缺省)参数

1. 语法：

def 函数名(参数1 = 默认值1):

函数体

1. 说明：

-- 缺省参数必须自右至左依次存在，如果一个参数有默认值，则右侧所有参数必须都有默认值。

-- 缺省参数可以有0个或多个。

练习：根据天/分钟/小时，计算总秒数。

代码：day09/exercise01

#### 位置形参

代码：day09/code02

语法：def 函数名(参数1)：

函数体

##### 星号元组形参

1. 语法：def 函数名(\*args):

函数体

1. 作用：收集多余的位置形参
2. 说明：

-- 一般命名为 \*args

-- 形参列表最多只能有一个

#### 命名关键字形参

1. 语法：def 函数名(\*,参数名):

函数体

def 函数名(\*args,参数名):

函数体

1. 作用：强制实参使用关键字传递

##### 双星号字典形参

代码：day09/code03

1. 语法：def 函数名(\*\*kwargs):

函数体

1. 作用：收集多余的关键字传参
2. 说明：

-- 一般命名kwargs

-- 形参列表中最多只能有一个

### 参数自左至右的顺序

位置形参 --> 星号元组形参 --> 命名关键字形参 --> 双星号字典形参

练习：

def fun1(a,b,\*args,c,d,\*\*kwargs):

print(a,b, args,c,d, kwargs)

作业1：自学容器函数。

作业2：定义数值累加函数。

预习：面向对象

扩展作业

#扩展作业1：定义向左移动的函数

#扩展作业2：定义向下移动的函数

#扩展作业3：定义向右移动的函数

### 可变/不可变类型在传参时的区别：

代码：

day10/code01

day10/exercise01

day10/exercise02

day10/exercise03

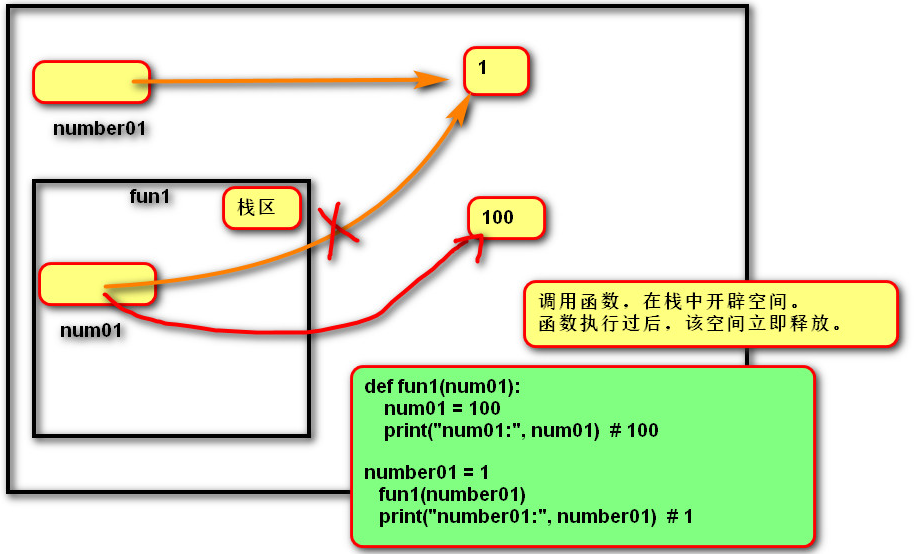
1. 不可变类型：

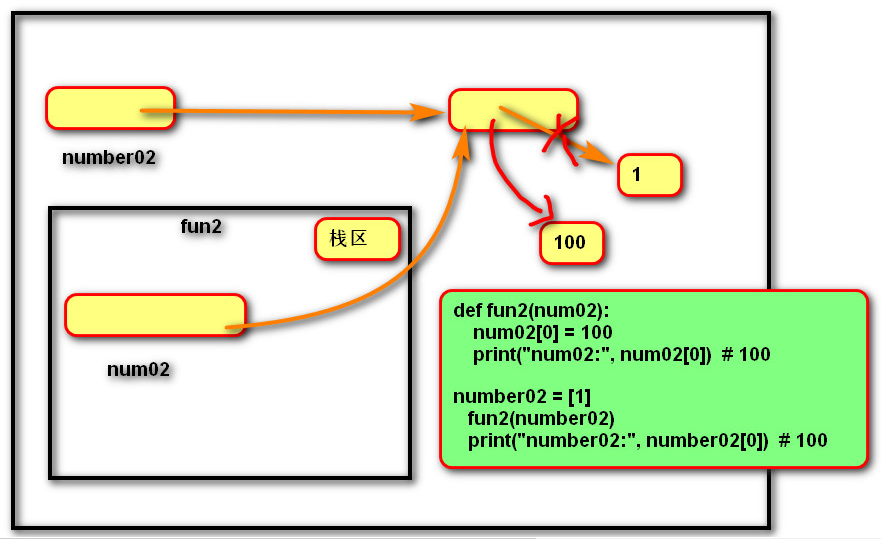
字符串、元组、固定集合、整数、浮点数、复数

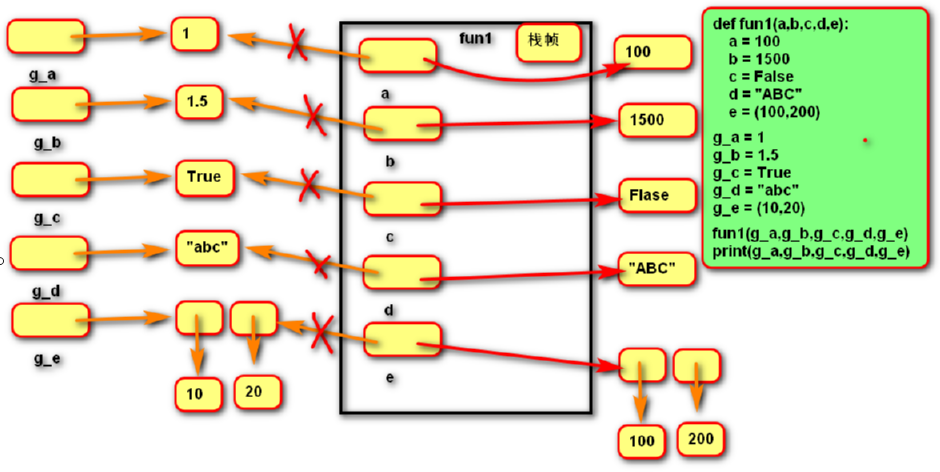
1. 可变类型：

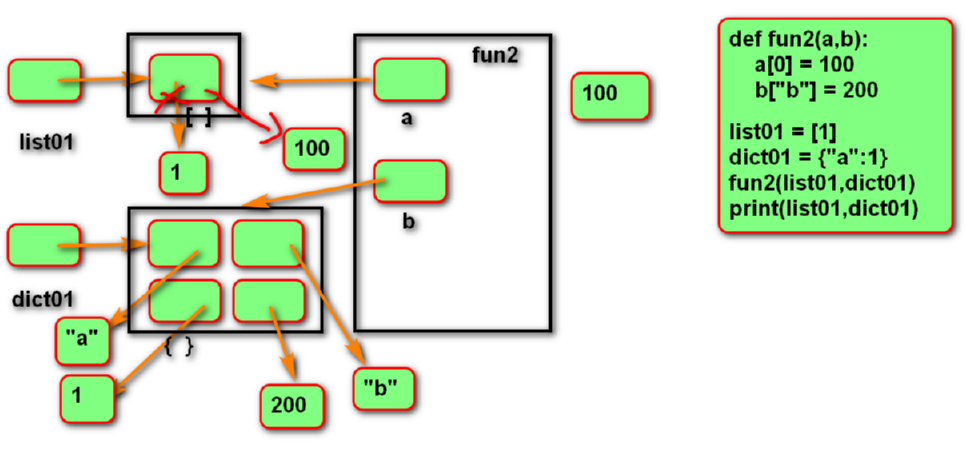
列表、字典、集合

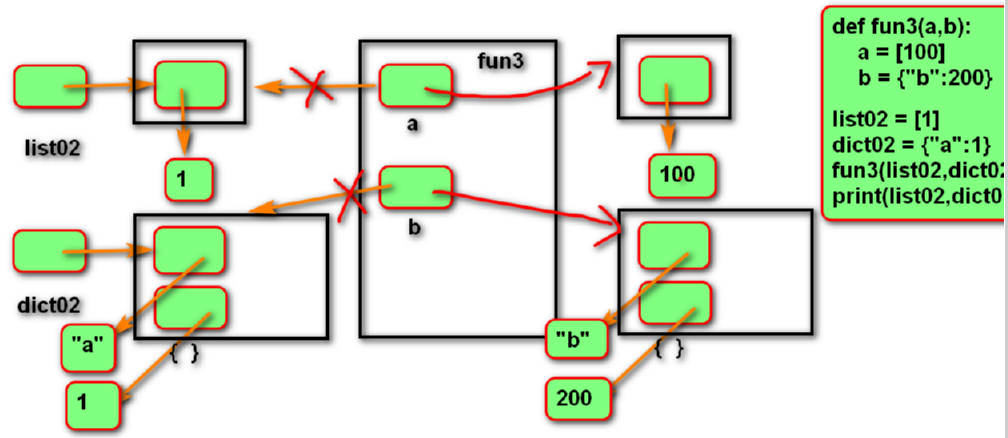
1. 不可变类型的数据传参时，函数内部不会改变原数据的值。
2. 可变类型的数据传参时，函数内部可能改变原数据的值。











## 作用域LEGB

代码：day10/code02

1. Local 局部作用域：函数内部。
2. Encolsing ……..
3. Global 全局作用域：py文件内部
4. Builtins内建模块作用域：builtins.py

### 变量名查找规则

由内到外

L ---> E ---> G ---> B

### 局部变量

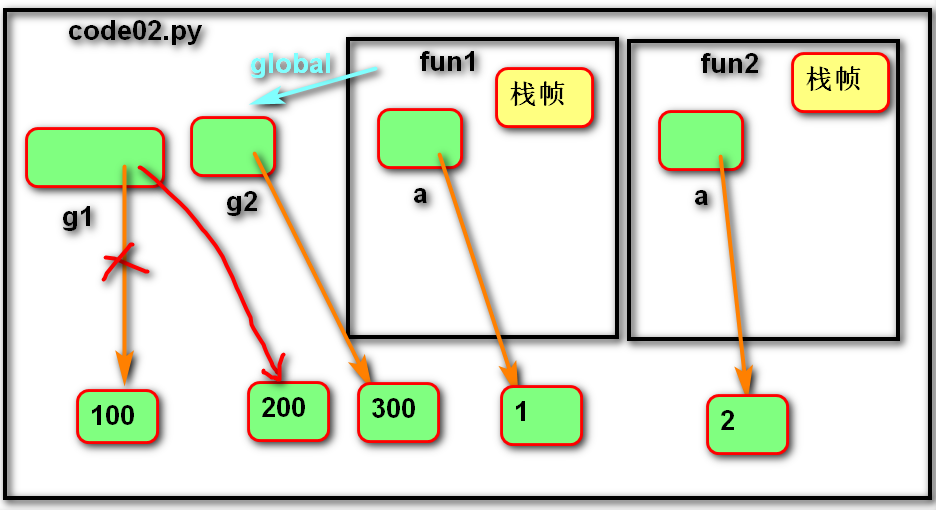
在方法体内部定义的变量

调用函数时才被创建，函数结束后自动销毁。

### 全局变量

定义在.py文件中的变量

函数体内部可以访问，但是不能直接修改（先使用global语句声明才能修改）。



练习：

统计hello函数调用的次数。