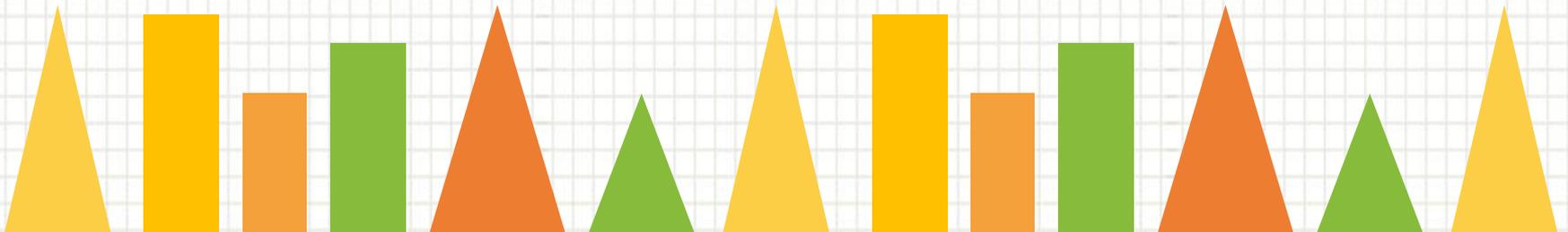


Python编程学习



Python介绍



Python（大蟒蛇）由Guido van Rossum于1989年开发，它是一种面向对象、解释型的程序设计语言。有人将Python归纳为以下特点：

Python语法简洁清晰，特色之一是强制用空白符(white space)作为语句缩进。

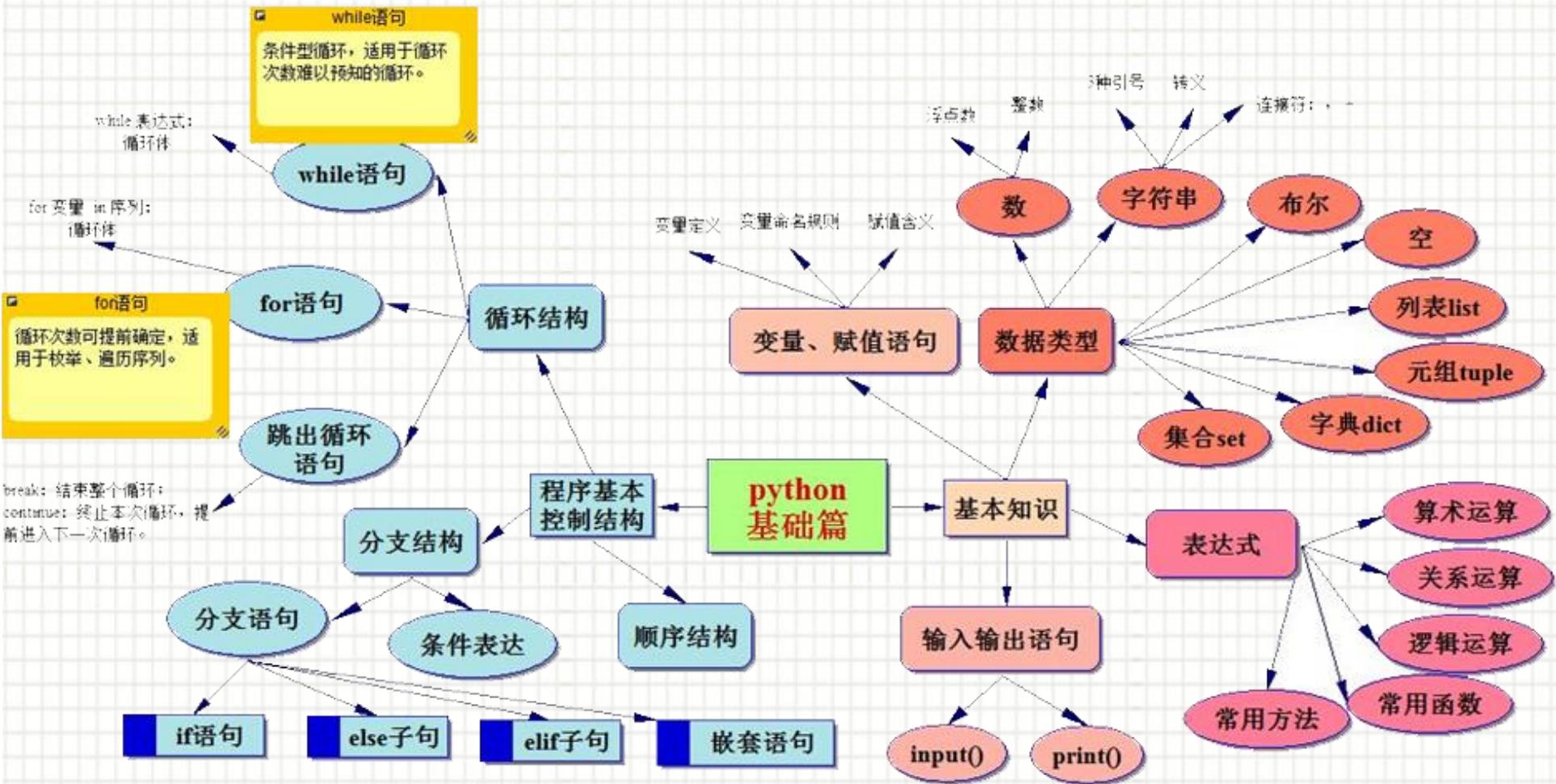
Python是一个高效的语言，读和写的操作都是很简单的，就像普通的英语一样。

Python是一个解释执行的语言，我们不需要去编译，我们只要写出代码即可运行。

Python是一个面向对象的语言，在Python里面一切皆对象。

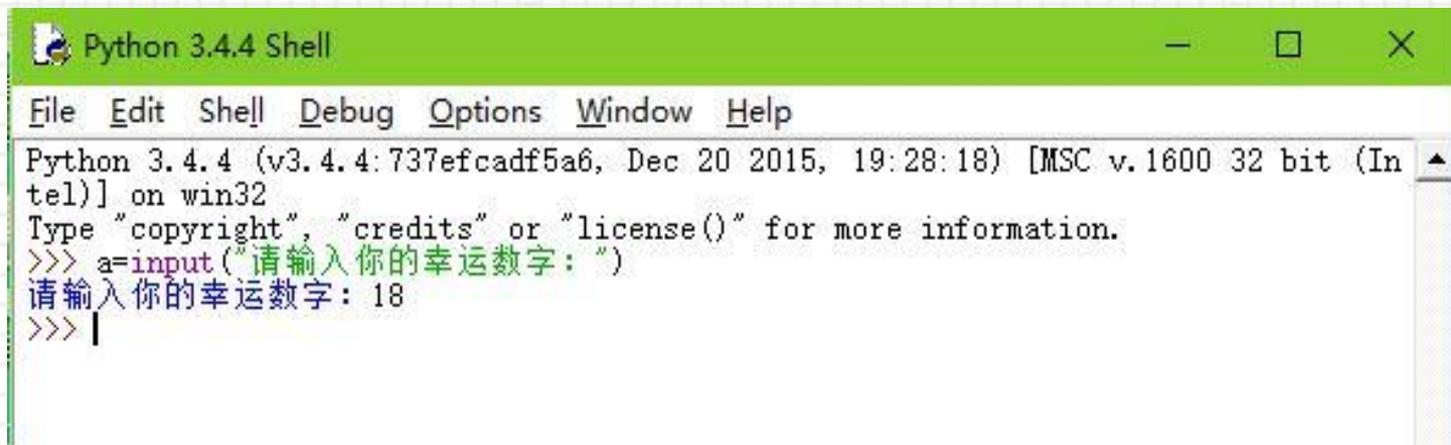


Python思维导图



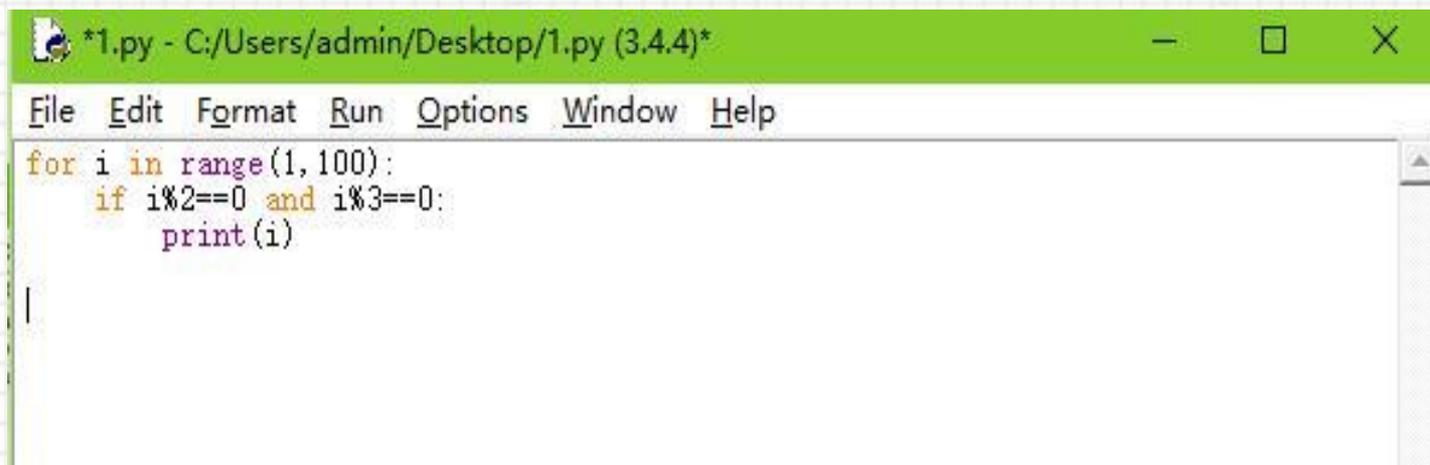
Python界面

交互式解释器=立即窗口



A screenshot of a Windows command prompt window titled "Python 3.4.4 Shell". The window has a green title bar and a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area shows the Python 3.4.4 startup message: "Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32". Below this, it says "Type 'copyright', 'credits' or 'license()' for more information." The user has entered the command ">>> a=input('请输入你的幸运数字: ')", and the prompt has responded with "请输入你的幸运数字: 18". The user has then entered ">>> |" and the prompt is waiting for more input.

```
Python 3.4.4 Shell
File Edit Shell Debug Options Window Help
Python 3.4.4 (v3.4.4:737efcadf5a6, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> a=input("请输入你的幸运数字: ")
请输入你的幸运数字: 18
>>> |
```



A screenshot of a Windows text editor window titled "*1.py - C:/Users/admin/Desktop/1.py (3.4.4)*". The window has a green title bar and a menu bar with "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The main text area contains Python code for finding numbers divisible by both 2 and 3 in the range 1 to 100. The code is: "for i in range(1,100):", " if i%2==0 and i%3==0:", " print(i)". The cursor is at the end of the code on the first line.

```
*1.py - C:/Users/admin/Desktop/1.py (3.4.4)*
File Edit Format Run Options Window Help
for i in range(1,100):
    if i%2==0 and i%3==0:
        print(i)
|
```

文本编辑器=代码窗口

目录

1

Python基础语句篇

- 1.1 孪生兄弟——输入与输出
- 1.2 多变的名词——变量
- 1.3 神奇的符号——运算符
- 1.4 奇妙的功能——函数
- 1.5 扩展的功能——模块

2

Python选择循环篇

- 2.1 艰难的抉择——if语句
- 2.2 循环中的高手——for语句
- 2.3 未知的循环——while语句

3

Python数据结构篇

- 3.1 Python的百宝箱——列表
- 3.2 被冻结的列表——元组
- 3.3 字符也疯狂——字符串
- 3.4 当索引不好用时——字典

第一部分

Python基础语句篇

■ 孪生兄弟——输入与输出

➤ 变量赋值

a = 160	-----	变量为int型
b = "test"	-----	变量为字符型
x=y=z=1	-----	多重赋值
<u>x, y, z = 1, 2, "string"</u>	-----	多元赋值

➤ 变量定义

- ① 变量名只能是由字母、数字和下划线组成。
 - ② 首字可以是字母和下划线。
 - ③ 系统关键字不能做变量名使用：if、int、def、str、print…
 - ④ 变量名区分大小写，如a与A是2个不同的变量。
- 例如：合法的变量 `__age`、`_name`、`x_1`

注意：python中变量没有数据类型，变量可以赋值任何的数据，就如往变量上贴一个标签而已。有的只是数据的转换函数。

➤ input () 外部输入

```
a=input("输入提示字符")
```

➤ input () 外部输入

```
print (表达式1,表达式2 ... )
```

int(): 转为整数类型

float(): 转为浮点数

str(): 转为字符串类型

bool(): 转为布尔型

True

False

注意：在python中区分大小写，基本是小写类型。

第一个小程序

实现功能：

使用 `input()` 函数提示输入你的幸运数字并保存到变量 `a` 中

①使用 `print()` 函数输出 `a*5`

②将变量 `a` 转换为浮点数，使用 `print()` 函数输出 `a*5`

③使用多元赋值 `b, c`，使用 `print()` 函数输出 `a, b, c` 等3个变量的积。

神奇的符号——运算符

表1—算术运算符

运算符	说明	运算符号	示例	结果	优先级
幂	返回x的y次幂	**	3**2 (3 ²)	9	高 ↓ 低
负	取负	-	a=5 b=-a	-5	
乘	两数相乘	*	4*6	24	
除	x除以y	/	6.0/3.0	2.0	
			6.0/3	2.0	
			6/3	2	
整除	返回商的整数部分	//	3//2	1	
加	两数相加或字符串相加	+	3+2	5	
减	两数相减	-	10-7	3	
取模	返回除法的余数 (见阅读材料)	%	10%4	2	
			-7%4	-3	
			5%-2	-1	
			-5%-2	-1	

神奇的符号——运算符

表2—关系运算符

运算符	运算符号	示例	结果
大于	>	1>2	False
小于	<	3<5	True
等于	==	21==75	False
不等于	!=	1!=2	True
大于等于	>=	13>=6	True
小于等于	<=	7<=2	False

表3—逻辑运算符

运算符	说明	运算符号	示例	结果
非	取反，真为假，假为真	not	not (8>6)	False
与	同时为真，才为真	and	1>2 and 3>2	False
或	1真，即为真	or	3<5 or 3<8	True

学习Python运算符



1. 把b1的值设置为 $17 < 118 \% 100$
 2. 把b2的值设置为 $100 == 33 * 3 + 1$
 3. 把b3的值设置为 $19 <= 2 ** 4$
 4. 把b4的值设置为 $-22 > = -18$
 5. 把b5的值设置为 $99 != 98 + 1$
- 实测以上5个赋值运算结果。

奇妙的功能——内建函数

表1-常用函数

函数名	说明	示例	结果
<code>abs(x)</code>	返回绝对值或者复数的模	<code>abs(-2)</code>	2
<code>divmod(x, y)</code>	返回两数的除数和余数	<code>divmod(5, 3)</code>	(1, 2)
<code>len(x)</code>	返回字符串或序列的长度	<code>len("Python")</code>	6
<code>range(start, stop, step)</code>	生成一个从start开始到stop的结束的数字序列	<code>range(1, 7, 2)</code>	[1, 3, 5]
<code>round(x)</code>	返回浮点数的四舍五入值	<code>round(7.6)</code>	8
<code>type(x)</code>	返回对象的类型	<code>x=32.3 type(x)</code>	float
<code>pow(x, y)</code>	返回x的y次方	<code>pow(3, 2)</code>	9

■ 奇妙的功能——内建函数

`range(start, stop, step)`



for `i=start to stop step` 步长



- ① `range(8)` : [0, 1, 2, 3, 4, 5, 6, 7]
- ② `range(2, 10)` : [2, 3, 4, 5, 6, 7, 8, 9]
- ③ `range(1, 10, 3)` : [1, 4, 7]

奇妙的功能——函数

表2-类型转换函数

函数名	说明	示例	结果
<code>chr(x)</code>	把ASCII码转换为对应的字符串	<code>chr(65)</code>	"A"
<code>ord(x)</code>	把字符串转换成对应的ASCII码	<code>ord("A")</code>	65
<code>float(x)</code>	把数字字符串或数字转换成浮点型	<code>float(15)</code>	15.0
<code>int(x)</code>	把数字转换成整型	<code>int(12.3)</code>	12
<code>str(x)</code>	把数字转换成字符串	<code>str(25.7)</code>	"25.7"

表3-序列、字符串处理函数

函数名	说明
<code>list(x)</code>	把序列对象转为列表
<code>str(x)</code>	把序列对象转为字符串
<code>tuple(x)</code>	把序列对象转为元组

奇妙的功能——自定义函数

格式:

```
def 函数名 (形参1, 形参2……) :  
    函数语句
```

实例1

```
def myfunction(x, y):  
    sum=x+y  
    print(sum)    #print意味着有输出  
    return(sum)  #return意味着只是返回一个
```

值给调用函数

```
a=myfunction(2, 3)    #2, 3为实参
```

实例2

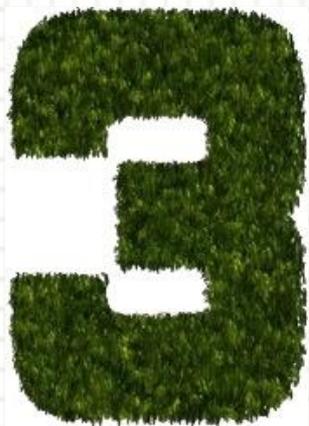
```
def a(x, y):  
    sum=x+y
```

```
c=a(2, 3)
```

```
print(c)
```

是否有结果?

运用Python函数



①Python下如何定义函数及变量？

在文本编辑器利用自定义函数求值：

②编写求两数的max和min函数，要求函数带形参。

③利用递归方法与自定义函数求5！

**注意：python具有强制缩进功能，必须要遵守层次的规则，否则无法执行代码。
注释用“#”。**

扩展的功能——模块

Python除了标准安装包里的模块外，还提供了第三方的模块供大家选用。

1. 引入模块

```
import 模块名  
import math
```

2. 引入模块下的函数

```
from 模块名 import 函数名1, 函数名2, ……  
from math import sqrt, fabs……
```

3. 引入模块下的所有函数

```
from 模块名 import *  
from math import *
```

实例 模块应用

```
import math as a          #引入math模块  
a=math.sqrt(15)          #使用math模块下得sqrt()函数  
print("%.3f"%a)
```



知识拓展

```
import math as m  
a=m.sqrt(16)
```

将模块名math赋予变量a，可以简化模块名，尤其是对于模块名比较复杂的情况。

实例 格式化输出

`print()` 的功能不仅仅局限于输出普通的数据, 如果你希望输出的形式更加多样, 可用 `%` 对字符串进行格式化输出。

格 式	描 述
<code>%d</code>	有符号整数(十进制)
<code>%f</code>	浮点数字(用小数点符号)
<code>%s</code>	字符串

`print` 格式化输出:

```
print("格式"%变量)
```

1. 格式化输出整数(integer)

```
a=12.012
```

```
print("%d"%a)
```

运行结果: 12

2. 格式化输出浮点数(float)

```
a=12
```

```
print("%.3f"%a)
```

"%.3f" 意味着3位小数

运行结果: 12.000

3. 格式化输字符串(string)

```
print("%.3s"%("Python"))
```

"%.3s" 意味着从左往右取3个字符

运行结果: Pyt

math模块提供了基本的数学运算，能够满足整数与浮点数的数学计算。

表1-常用数学函数

函数名	说明	示例	结果
<code>sqrt(x)</code>	求x的平方根	<code>math.sqrt(121)</code>	11
<code>fabs(x)</code>	求绝对值	<code>math.fabs(-31.2)</code>	31.2
<code>pow(x, y)</code>	求x的y次方	<code>math.pow(2, 3)</code>	8
<code>floor(x)</code>	对x向下取整，返回整数 小于或者等于x	<code>math.floor(3.14)</code>	3
<code>ceil(x)</code>	对x向上取整，返回整数 大于或者等于x	<code>math.ceil(3.14)</code>	4
<code>factorial(x)</code>	求x的阶乘	<code>math.factorial(4)</code>	24
<code>exp(x)</code>	求e的x次方	<code>math.exp(5)</code>	148.4131……

除了常用的数学函数，math还有三角函数、双曲函数和特殊函数。

random模块的应用范围是比较广泛的，可以利用产生随机数的功能来模拟或者用于产生随机数输出的程序，如猜谜小游戏、抽奖……

表2-常用random函数

函数名	说明	示例	结果
random()	生成[0, 1)的随机数	random.random()	0.5
uniform(a, b)	用于生成一个指定范围内的随机浮点数	random.uniform(1, 5)	2.326
randint(a, b)	用于生成一个指定范围内的整数	random.randint(1, 5)	3
randrange(a, b, step)	从指定范围内，按指定基数递增的集合中 获取一个随机数	random.randrange(1, 10, 2)	7
choice(序列)	从序列中获取一个随机元素	random.choice([3, "hello", 21, 3.2])	21
shuffle(序列)	用于将一个列表中的元素打乱	random.shuffle([3, "hello", 21, 3.2])	序列打乱
sample(序列, k)	从指定序列中随机获取指定长度片断	random.sample([3, 6, 7, 9], 2)	[7, 3]

体验random模块

选择random模块的部分函数，
在交互式解释器进行测试！



第二部分

Python选择循环篇

艰难的抉择——IF语句



➤ if语句格式

```
if 条件表达式:  
    语句块
```

```
if 条件表达式:  
    语句块1  
else:  
    语句块2
```

```
if 条件表达式1:  
    语句块1  
elif 条件表达式2:  
    语句块2  
elif 条件表达式3:  
    语句块3  
.....  
else:  
    语句块n
```

艰难的抉择——IF语句



➤ if语句简化



知识拓展

if...else可简化缩写为:

语句块1 if 条件表达式 else 语句块2

示例:

```
x,y=65,87
```

```
if x>y:
```

```
    max=x
```

```
else:
```

```
    max=y
```

等同于:

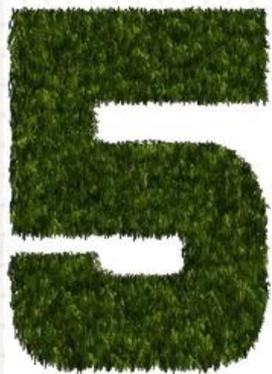
```
max=x if x>y else y
```

综合习题1

“水仙花数”是数学中非常经典的一个问题，如果一个三位数其各位数字的立方和等于该数本身，则这个数就是所谓的“水仙花数”。例如：370是一个“水仙花数”，因为 $370=3^3+7^3+0^3$ 。

编写一个Python程序，实现如下功能：

- ①使用input()函数输入一个三位数。
- ②使用if语句判断该数是否是“水仙花数”



循环中的高手——for语句



➤ for 语句格式

```
for 变量 in 序列:  
    语句块 (循环体)
```



温馨提示

“for 变量 in 序列”循环的实质就是把每个元素代入变量，然后执行缩进块的语句。

示例：

```
for i in range(1,10):  
    s=s+i  
    print(s)
```



温馨提示

“for”与“range()”是天生的一对！你懂得！

综合习题2

任选1道习题：

- ①请写一个Python程序打印出 0~100 所有的奇数。
- ②编写一个Python程序，实现如下功能： 判断101-200之间有多少个素数，并输出所有素数。
(素数是只能被1或者自己整除的自然数。)



■ 未知的循环——while语句



➤ while语句格式

```
while 表达式:  
    语句块 (循环体)
```

示例:

```
i,s=1,0  
while i<=10:  
    s=s+1  
    i=i+1  
print(s)
```



温馨提示

Python与VB区别:

- ①if没有endif
 - ②for没有next
 - ③while没有loop
- 均以:开头

未知的循环——while语句

➤ While跳出循环

while和continue命令，结合分支语句，实现当在特定条件得到满足时跳出循环。continue 用于跳过该次循环剩余的语句，进入下一次循环，break 则是用于退出当前所在循环。如下面代码用于找出小于100的最大素数：

```
for n in range(100, 1, -1):  
    for i in range(2, n):  
        if n%i==0:  
            break  
    else:  
        print(n)  
        break
```

输出：97

若删除上面程序中第二个break语句，则实现100以内全部素数的输出。

综合习题2

编写一个Python程序，实现如下功能：使用while循环语句求 $1+2!+3!+\dots+20!$ 的和。



第三部分

Python数据结构篇

Python提供的序列类型(sequence)可以说在所有程序设计语言的类似数据结构中是最灵活、功能最强大的。序列中的每个元素被分配一个序号——即元素的位置，也称为索引。





► 列表的格式

列表（list）是Python内置的**可变序列**，列表中每一个数据称为元素，所有元素放在一对中括号“**[]**”中，并使用逗号分隔出来。**同一个列表中元素的数据类型可以各不相同**。例如：

```
[1, 3, 5, 7, 9]
```

```
['Michael', 1.65, 50, 'male', [85, 90, 83.5]]
```

```
['cat', 'tiger', 'snake']
```

```
[ ]
```

第一个元素的索引是0，第二个元素的索引是1，以此类推，也可以使用负索引，最后一个元素的索引是-1，倒数第2个元素的索引是-2，依次类推，通过索引或以获取对应位置上的元素值。

List列表练习

使用list()函数将range(1, 10, 2)转为列表。

append(X) 方法:

将元素X添加到列表尾部，括号里参数只能是一个元素；

extend(L) 方法:

将列表L里的所有元素添加到列表尾部，A可以是一个列表名称或者一个具体的列表对象；

insert(x, k) 方法:

将元素X插入到索引为k的列表元素前面；

pop() 方法:

删除并返回指定（默认是最后一个）位置上的元素



List列表切片练习



切片是Python序列的一个重要操作，使用两个冒号分隔的3个数字来进行片，第一个数字表示切片的起始位置（默认为0），**第二个数字表示切片截止（但不包含）位置（默认是列表长度）**，第三个数字是切片步长（默认为1，此时第二个冒号可省略）。

有两个列表：

`name['Sally' , ' Jane' , 70, ' Bill' , 85, ' Tom']` , `score`为
`[65, 70, 85, 78]`，实现：将`score`里的65、78取出来，分别添加到
`name`列表里的‘Sally’和‘Tom’元素后面。（思考多种实现方法）

被冻结的列表——元组



► 元组的格式

元组 (tuple) 与列表类似，也是Python的一个重要序列结构。元组最重要特征是不能被修改，这就使得元组一旦被创建，就不能修改其元素值、也不能添加删除元素。

定义元组时，所有元素被放在一对圆括号“ () ”中，**用逗号隔开**。

例如：

(2, 4, 6, 8)

('Tom', 'Mike', 'Jane')

(1, 'ZJ', 2, 'SH')

1, 2, 3

被冻结的列表——元组



元组的创建

使用赋值号“=”将一个元组赋值给变量，就可以创建一个元组变量，但其元素的值是不能修改的，如：

```
>>>tuple1=(2, 3, 4, 5)           #创建普通元组
>>>tuple2=('Tom', 16, [80, 92, 88]) #创建混合元组，列表为其中一个元素
>>>tuple3=(5,)                   #创建只有一个元素的元组，逗号不能省略
>>>tuple4=()                       #创建空元组
```

还有一种创建元组的方法是，用`tuple()`函数，将一个列表或其它序列转换为元组，如：

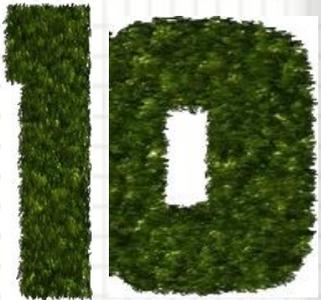
```
>>>alist=[5, 10, 15, 20]
>>>tuple(alist)
(5, 10, 15, 20)
```

元组测试

创建元组时最重要的符号是逗号，只要有逗号，即使不写括号，也可以创建一个元组对象。

测试1： 输入`a=1, 2, 3,`
将`a[2]`元素替换为`'mary'`

测试2： 创建一个10到20（包含10和20）的整数元组序列，思考并实践有几种实现方法。



当索引不好用时——字典



字典的概念

在Python里可以用字典来存储有映射关系的数据，比如通讯录系统，通过某人的姓名来快速查阅到他的通讯信息，即姓名和通讯信息形成了映射关系。

姓名称为“键”，通讯信息称为“值”。字典是无序的，它的元素由两部分组成：键（key）、值（value），所有元素放在大括号“{ }”中，每个元素的键和值用冒号“:”隔开，相邻元素之间用逗号隔开。例如：

字典的语法格式

```
{键1:值1, 键2:值2, 键3:值3, .....}
```

➤字典的创建方法

1. 用等号“=”直接将一个字典对象赋值给变量。

例如：

```
>>> a_dict={'A':65, 'B':66, 'C':66, 'a':97}
```

2. 使用dict()函数根据给定的键-值来创建字典。

例如：

```
>>> b_dict=dict(name='Liming', age=18)
```

3. 创建只有键没有值的空字典：

```
>>> d_dict=dict.fromkeys(['name', 'age', 'sex'])
```

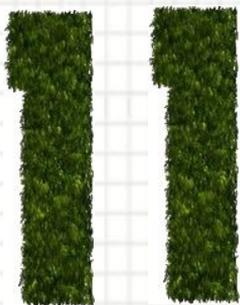
```
>>> d_dict
```

```
{'name': None, 'sex': None, 'age': None}
```

4. Del命令可以删除整个字典

排序算法练习

将 $m=[32, 45, , 18, 7, 21, 39]$ 分别利用冒泡排序和选择排序算法实现 m 从小到大排序





THANKS

Add your text in here

