Python基础一

一、课前准备

1. 按照预习视频安装好Python环境以及编辑器；
2. 根据课程大纲提前预习Python基础语法；

二、课堂主题

本小节主要学习Python的基础，比如：输出和输入，变量、数据类型、运算符、关键字以及判断语句和循环语句等Python基础知识点。

三、课堂目标

1. 掌握Python语法结构以及书写特点；
2. 掌握Python基础知识点；
3. 根据判断语句和循环语句充分了解程序的执行顺序。

kaikeba.com

四、知识要点

1、语句与注释

语句： 程序进行编写， 执行的每一行代码，叫做语句。

注释： 对代码的解释和说明， 可以提供代码的可读性。

注释分为单行注释和多行注释

单行注释是以 # 开始

多行注释，可以使用三对双引号""" """ 或者三对单引号 ''' '''

```
# 定义字符串变量name
name = "西施"

'''
    我是多行注释
    我是多行注释
    我是多行注释
'''
age = 18

"""
    我也是多行注释
    我也是多行注释
    我也是多行注释
"""
sex = '女'
```

注意： ctr + / 单行注释快捷键

2、变量与赋值

2.1、 变量的定义

变量： 通俗理解就是存储程序数据的容器。

变量定义的格式：变量名 = 数据 （变量名尽量有含义，方便理解 ）

```
name = "马超"
print(name)

skill = 450  # 定义了一个变量名字叫做skill，存储的数据是450
print(skill)

money = 1.98
print(money)

is_ok = True
print(is_ok)

# 提示： 在python里面不需要指定数据的类型，会根据数据自动推导出数据类型

# 通过type查看变量的类型
name_type = type(name)
```

开课吧 kaikeba.com

```
print(name_type)

skill_type = type(skill)
print(skill_type)

money_type = type(money)
print(money_type)

print(type(is_ok))
```

总结： 常用的数据类型 int, str, float, bool, list, tuple, dict, set

2.2、 变量的命名规则

变量名：是由字母、数字、下划线组成,注意是只有这三种，但是不能以数字开头。

错误的使用：

```
3name = '太乙真人'
print(3name)
name!age = 108
```

> 变量命名方式： 驼峰命名法 和 下划线命名法
>
> 驼峰命名法: 小驼峰和大驼峰
> 小驼峰: 第一个单词首字母要小写，其它单词首字母都大写
> 大驼峰: 每个单词首字母都大写

> 下划线命名： 单词都使用小写字母，单词之间使用下划线进行分割, 比如: my_name

```
hero_name = '鲁班' # 下划线命名法 -> 推荐使用
heroName = '刘备' # 小驼峰
HeroName = '元芳' # 大驼峰
```

2.3 、关键字

关键字： 在python里面具有特殊功能的标识符（理解成变量名、函数名）， 关键字不能作为变量名使用。

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break',
'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for',
'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or',
'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

2.4、 常用的数据类型转换

开课吧 kaikeba.com

函数	说明
int(x [,base])	将x转换为一个整数
float(x)	将x转换为一个浮点数
complex(real [,imag])	创建一个复数，real为实部，imag为虚部
str(x)	将对象 x 转换为字符串
repr(x)	将对象 x 转换为表达式字符串
eval(str)	用来计算在字符串中的有效Python表达式,并返回一个对象
tuple(s)	将序列 s 转换为一个元组
list(s)	将序列 s 转换为一个列表
chr(x)	将一个整数转换为一个Unicode字符
ord(x)	将一个字符转换为它的ASCII整数值
hex(x)	将一个整数转换为一个十六进制字符串
oct(x)	将一个整数转换为一个八进制字符串
bin(x)	将一个整数转换为一个二进制字符串

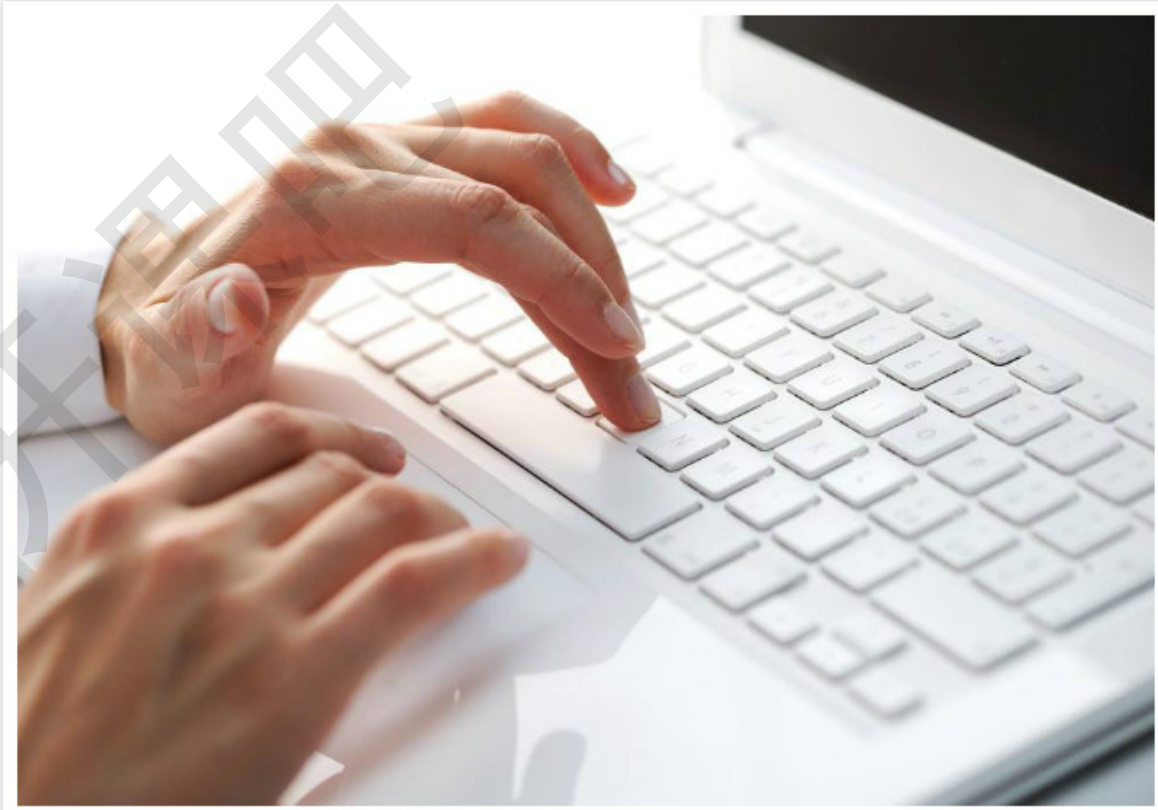
3、输入和输出

生活中无时无刻其实都存在输入和输出的例子,我们先举例看一下生活中有哪些输入输出的情况,在来对比Python中的输入输出和生活中的有什么相同和不同之处。

3.1、生活中的输入

生活中的输入无处不在，例如你打出的文字、说出的语音。

开课吧 kaikeba.com



3.2、Python中的输入

和输出同理我们也可用到控制台来记录输入结果,同样用到一个函数`input()`，我们可以利用这段代码在控制台输入，然后在利用刚刚学的`print()`函数把你输入的结果在输出出来。

```
name = input()
print(name)
```

> 当你运行完毕`name = input ()`代码并将鼠标光标移动到控制台，Python交互式命令就在等你的输入了,你可以输入任意字符,然后按回车完成输入。

开课吧 kaikeba.com

3.3、生活中的输出

我们看到的内容、听到声音,这就是**生活中**最简单的**输出方式。**

3.4、Python中的输出

Python中的输出和生活中的输出是一样的原理，只不过Python中的输出，特指是在在控制台中输出,或者是将你准备要输出的内容相应的输出到你的设备上，如你在手机上看到的文字、图片、视频等数据,其实本质上也是我们敲打代码输出到手机上的.那我们先来看一下第一种，如何将你想要输出文字输出到控制台。

开课吧 kaikeba.com

例如我们将一段文字"Hello World"输出到控制台，这里面利用的`print()`函数进行输出,以后我们会讲到函数的概念，先暂时知道print()可以帮助我们进行输出。

```
print('Hello World')
```

我们可以试试输出100+100会是什么结果

```
print(100 + 100)
```

3.5、Python中的格式化输出

```
# 格式化符号： %s, %d, %f, %x
# %s: 输出字符串
# %d: 输出int类型数字
# %f: 输出浮点数
# %x: 输出16进制数据

PI = 3.1415926
print("π的值是%.2f" %PI)

job = "数据分析师"
money = 25000
print("%s的薪资是：%d" %(job,money))
```

4、运算符

4.1、 算数运算符

Python支持以下几种运算符:x=10, y=20, z = 25

运算符	描述	实例
+	加	两个对象相加 x + y输出结果 30
-	减	得到负数或是一个数减去另一个数 x - y输出结果 -10
*	乘	两个数相乘或是返回一个被重复若干次的字符串x *y 输出结果 200
/	除	y / x输出结果 2
//	取整除	返回商的整数部分 9//2 输出结果 4 , 9.0//2.0 输出结果 4.0
%	取余	返回除法的余数 z % x 输出结果 5
**	指数	x**y 为10的20次方， 输出结果 100000000000000000000

> 混合运算时，优先级顺序为： `**` 高于 `*` `/` `%` `//` 高于 `+` `-` ，为了避免歧义，建议使用 `()` 来处理运算符优先级。并且，不同类型的数字在进行混合运算时，整数将会转换成浮点数进行运算。

4.2、 赋值运算符

运算符	描述	实例
=	赋值运算符	把 = 号右边的结果 赋给 左边的变量，如 num = 1 + 2 * 3，结果num的值为7

```
# 单个变量赋值
num = 1000

# 多个变量赋值
num1, f1, str1, b1  = 100, 3.14, "hello" ,True
print(str1)
```

4.3、 复合赋值运算符

运算符	描述	实例
+=	加法赋值运算符	z += x 等效于 z = z + x
-=	减法赋值运算符	z -= x 等效于 z = z - x
*=	乘法赋值运算符	z *= x 等效于 z = z * x
/=	除法赋值运算符	z /= x 等效于 z = z/ x
%=	取模赋值运算符	z%= x 等效于 z = z% x
**=	幂赋值运算符	z** **= x 等效于 z= z****x
//=	取整除赋值运算符	z //= x 等效于 z = z // x

4.4、 比较（关系）运算符

Python语言支持比较运算符，以下假设变量 x 为 10, y为 20

kaikeba.com

运算符	描述	实例
==	x等于 - 比较对象是否相等	(x == y) 返回 False
!=	不等于 - 比较两个对象是否不相等	(x != y) 返回 True
>	大于 - 返回x是否大于y	(x > y) 返回 False
<	小于 - 返回x是否小于y。所有比较运算符返回1表示真，返回0表示假。	(x < y) 返回 True
>=	大于等于 - 返回x是否大于等于y	(x >= y) 返回 False
<=	小于等于 - 返回x是否小于等于y	(x <= y) 返回 True

4.5、 逻辑运算符

Python语言支持逻辑运算符，以下假设变量 x 为True, y为 False

运算符	描述	实例
and	布尔"与"	x and y 返回False
or	布尔"或"	x or y 返回 True
not	布尔"非"	not x 返回False

5、 if语句

计算机之所以能做很多自动化的任务，因为它可以自己做条件判断。

比如，输入用户年龄，根据年龄打印不同的内容，在Python程序中，用`if`语句实现：

```
age = 20
if age >= 18:
    print('your age is', age)
    print('adult')
```

根据Python的缩进规则，如果`if`语句判断是`True`，就把缩进的两行print语句执行了，否则，什么也不做。

也可以给`if`添加一个`else`语句，意思是，如果`if`判断是`False`，不要执行`if`的内容，去把`else`执行了：

开课吧 kaikeba.com

```
age = 6
if age >= 18:
    print('你的年龄', age)
    print('成年')
else:
    print('你的年龄', age)
    print('未成年')
```

注意不要少写了冒号`:`。

当然上面的判断是很粗略的，完全可以用`elif`做更细致的判断：

```
age = 6
if age >= 18:
    print('成年')
elif age >= 7:
    print('青少年')
else:
    print('儿童')
```

`elif`是`else if`的缩写，完全可以有多个`elif`，所以`if`语句的完整形式就是：

```
if <条件判断1>:
    <执行1>
elif <条件判断2>:
    <执行2>
elif <条件判断3>:
    <执行3>
else:
    <执行4>
```

`if`语句执行有个特点，它是从上往下判断，如果在某个判断上是`True`，把该判断对应的语句执行后，就忽略掉剩下的`elif`和`else`，所以，请测试并解释为什么下面的程序打印的是`teenager`：

```
age = 20
if age >= 7:
    print('青少年')
elif age >= 18:
    print('成年')
else:
    print('儿童')
```

6、 循环

我们来试想这样一种情况,现在让你们在控制台输出100条`hello world`,本质上,我们写一百条`print`函数输出就可以了,但是如果一千条一万条呢.这就要用到循环语句了.

6.1、while 循环

while循环语句语法

开课吧 kaikeba.com

```
while 条件:
    条件满足时，做的事情1
    条件满足时，做的事情2
    条件满足时，做的事情3
    ...(省略)...
```

例如输出100条`helloworld`

```
i = 1
while i <=100:
    print('hello Python')
    i= i + 1
```

相对应,在while循环语句里面,每执行一次循环语句,`i`就会加1,直到`i`等于101时不满足`i<=100`的条件,循环就结束了

6.2、for循环

for循环和while一样同样可以进行循环,并且是运用最多的循环方式,而且它有一项非常厉害的功能——`遍历`,在Python中`for`循环可以遍历任何序列项目,如字符串,或者今后会学到的列表,**例如我们遍历字符串,就特指将字符串的所有字符全部访问一遍**

```
names = ['Mike', 'HuaHua', 'Jay']
for name in names:
    print(name)
```

执行这段代码，会依次打印`names`的每一个元素：

```
Mike
HuaHua
Jay
```

所以`for x in ...`循环就是把每个元素代入变量`x`，然后执行缩进块的语句。

再比如我们想计算1-10的整数之和，可以用一个`sum`变量做累加：

```
sum = 0
for x in [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]:
    sum = sum + x
print(sum)
```

Python还提供了一个range()函数,可以配合我们的for循环使用,例如:

```
for i in range(10):
    print(i)

#效果等同于 while 循环的：

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

开课吧 kaikeba.com

6.3、break

在循环中，break 语句可以提前退出循环。例如，本来要循环打印1～50的数字：

```
m = 1
while m <= 50:
    print(m)
    m = m + 1
print('END')
```

上面的代码可以打印出1~50。

如果要提前结束循环，可以用 break 语句：

```
m = 1
while m <= 50:
    if m > 10: # 当m = 11时，条件满足，执行break语句
        break # break语句会结束当前循环
    print(m)
    m = m + 1
print('END')
```

执行上面的代码可以看到，打印出1~10后，紧接着打印 END，程序结束。

可见 break 的作用是提前结束循环。

6.4、continue

在循环过程中，也可以通过 continue 语句，跳过当前的这次循环，直接开始下一次循环。

```
n = 0
while n < 10:
    n = n + 1
    print(n)
```

上面的程序可以打印出1～10。但是，如果我们想只打印奇数，可以用 continue 语句跳过某些循环：

```
n = 0
while n < 10:
    n = n + 1
    if n % 2 == 0: # 如果n是偶数，执行continue语句
        continue # continue语句会直接继续下一轮循环，后续的print()语句不会执行
    print(n)
```

执行上面的代码可以看到，打印的不再是1～10，而是1，3，5，7，9。

可见 continue 的作用是提前结束本轮循环，并直接开始下一轮循环。

开课吧 kaikeba.com

五、总结

1. 本节课的所有知识点全是重点，后面的学习离不开基础语法；
2. 需要理解的是break和continue在循环中作用，并且这两个关建字只能在循环中使用；
3. 所有的代码都要多敲几遍，练习是学习编程最简单的途径！！！

开课吧
kaikeba.com