day01笔记

1、变量

什么是变量?

```
print() ---打印输出
print(111)
```

```
变量：将程序运行的中间值，临时存储起来，以便再次使用
昵称
定义一个变量

name = 'yuan'
'yuan' ： 值（数据）
 =  ：赋值操作
name ：  变量名
print(name) 使用定义的变量名
```

```
a = "麻花腾"
print(a)

b = "码云"
print(b)

c1 = "李少卿"
print(c1)

a_1 = "朱德"
print(a_1)

a_b = "李宁"
print(a_b)

_a = "莉莉"
print(_a)

name = "猪哥"
print(name)
```

变量的命名规范

1. 数字，字母，下划线组成
2. 不能以数字开头
3. 禁止使用python中的关键字
4. 变量名要具有可描述性
5. 变量名区分的大小写

6. 不能使用中文和拼音
7. 推荐写法：
    1. 驼峰体
    2. 下划线（管方推荐）

以数字为开头命名变量会报错：

```
>>> 1a = 'alex'
  File "<stdin>", line 1
    1a = 'alex'
     ^
SyntaxError: invalid syntax
```

驼峰体和下划线命名变量示例，很显然下划线的命名法更直观些：

```
YuanOfOldboy = 18
yuan_of_oldboy = 18

名字 = "yuan"
print(名字)
nihao = "yuan"
print(nihao) #python现在支持中文当做变量， 但我们不可以用拼音，中文
name = "yuan"
print(name)

name = "yaun"
Name = "袁"
print(name)
print(Name)
```

python中的关键字

```
>>> import keyword
>>> print(keyword.kwlist)
['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']
>>>
```

<内置功能打印>

```
>>> print = print
>>> print(print)
<built-in function print>
>>>
```

```
a1 = "yuan"
b_a = "袁"
a_l_a = "goub"
# 命名要有意义，这样自己都看不懂，更别谈别人能看懂
print(a1)
print(b_a)
print(a_l_a)
```

```
age = 18
age1 = 20

print(age, age1) #print可以打印多个内容，以逗号分隔
```

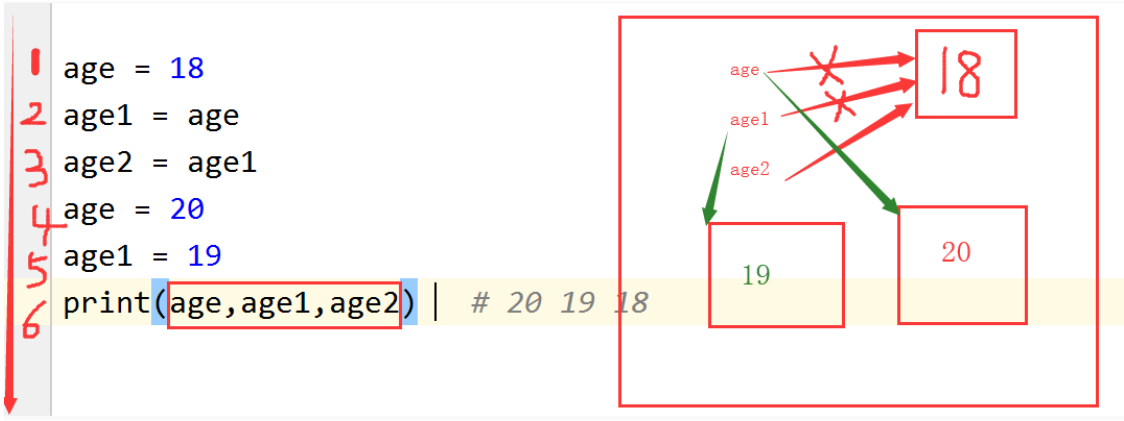
请看下面一段赋值操作：

```
age = 18
age1 = age
age2 = age1
age = 20
age1 = 19
print(age,age1,age2)
```

最终打印出来的结果是：

```
20 19 18
```

具体的赋值操作如下图所示



2、常量

怎么定义常量?

在Python中，没有严格意义的常量。大家约定俗成的是，变量名大写的变量就被视为常量，在开发过程中不会被轻易修改。例如：

```
ID = 1010120120  #(不建议修改)

print(ID)
```

常量：变量名大写的就是常量

变量：用于我们后期开发时使用

常量：用于配置文件中

3、注释

为什么要注释?

注释的作用是给一些晦涩难懂的代码进行标注或解释。注释后的代码不会被执行。

Python中的注释分为两种：

- 单行注释（当行注释）：用#开头表示
- 多行注释：用三对“ ”或‘ ’包裹，可以换行

具体示例为：

```
# 这个是单行注释的示例
# 换行之后要在开头加一个#

"""
窗前明月光，
玻璃好上霜.
要不及时擦，
整不好就脏.
"""


# PEP8:开发规范

"""
窗前明月光，
玻璃好上霜.
要不及时擦，
整不好就脏.
"""
```

4、基础数据类型

python中由哪些数据类型?

在python中基础的数据划分（数据类型）总共有7种，今天主要学习其中3种：str（字符串），int（整型），bool（布尔值）

整型

整型数据在python中的关键字为`int`。整型数据的主要用途是进行计算和比较。

整型数据的基本用法和操作如下：

```
a = 10
b = 5
print(a - b)
print(a + b)
print(a * b) # * 乘
print(a / b) # / 除
```

字符串

字符串在python中的关键字为`str`。在python中，只要用引号引起来的就是字符串。

字符串使用示例：

```
a = "你好"
b = '你好'
"""你好"""     # 三引号可以表示多行字符串，多行注释的原理就是一个未赋值的字符串
'''你好'''
print(a,b)
a = "123"
b = 123
```

需要注意的是，使用`print`函数时，不要在变量的两端加引号：

```
# 会出现的问题
a = "yuan"
b = "123"
print("a, b")
print(a, b)
```

字符串的`+`操作：

```
a = "yuan"
b = "三哥"
c = a + b
print(c)

a = "yuan dsb"
b = "三哥"
c = a + b    # 字符串拼接
print(c)
```

字符串的`*`操作

```
a = "坚持"
print(a * 8) #字符串的乘法

a = "坚强"
b = "python3"
print(a + b * 5)
```

字符串的操作小结：

- `+`拼接：必须都是字符串才能相加
- `*`拼接：字符串和数字相乘

布尔值

布尔值在编程中用用来表示真假。在python中`真`用`Ture`表示；`假`用`False`表示。只有python中的`True`和`False`的首字母是大写的。示例如下：

```
print(3 > 2) # Ture 成立
print(2 > 3) # False 不成立
```

5、用户交互

在python中，用`input()`函数实现用户和程序的交互。`input()`是输入的意思。使用示例如下：

```
qq_user = input("QQ账号：") #坑  --阻塞
qq_pwd = input("QQ密码：")
print(qq_user, qq_pwd)
```

当程序运行到`input`语句时，会发生阻塞，等待用户进行输入。程序会一直保持阻塞状态，除非用户输入内容或终止程序。

需要注意的是，在Python 3中`input`获取的内容全都是字符串。因为这样的原因，下面的程序会报错：

```
num = input("请输入数字：")
print(num + 5)

'''
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: can only concatenate str (not "int") to str
'''
```

这两行代码的本意是，当用户输入一个数字之后，程序自动输出一个比输入数字大5的数字。但是因为`input`获取的内容是字符串，字符串是不能和整型数字进行加和操作的，故而程序报错。我们可以使用`tpye()`函数查看变量的数据类型

```
num = input("请输入数字")
print(type(num))  # 查看数据类型


'''
>>> print(type(num))
<class 'str'>
>>>
'''
```

通过使用`int()`函数可以将字符串转化为整型数据。同样地，也可以使用`str()`函数，把整型数据转换成字符串。

```
a = int("12") #字符串转成整型
b = str(23) #整型转成字符串
```

需要注意的是，使用`int()`函数将字符串转换为整型时，字符串中的内容必须全部都是数字，否则会报错：

```
>>> int("jjj1737")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'jjj1737'
>>>
```

那么我们上面的例子就可以改成这个样子：

```
num = input("请输入数字：")
a = int(num)
print(a + 5)
```

这样的话，当我们输入一个数字之后，通过`int()`函数，输入内容转换成为整型，然后就可以进行加和操作了：

```
请输入数字:3
8
```

也可以把`inpu()`操作直接放到`int()`函数中：

```
num = int(input("请输入数字:"))
print(num + 5)
```

用户交互总结：

- `input()`是输入，获取到的内容都是字符串
- `type()`函数用来查看数据类型
- `int('字符串')`可以将字符串转换成整型，字符串中的内容必须全部都是数字
- `str(整型)`函数可以将整型转换成字符串

6、流程控制语句

流程控制语句，也就是条件语句，通过选择判断，决定下一步的操作内容。例如：如果是男的，就来看我。

流程控制语句的关键字是`if`，是`如果`的意思。流程控制语句使用冒号`：`表示语句结束。

Python中使用缩进体现代码间的从属关系。一般使用四个空格或一个`Tab`键代表一次缩进。需要注意的是，编程时，不要混合使用`Tab`键和空格，否则一旦报错，很难找到问题的所在。

知识点补充：在Python中，`=`表示的是`赋值`操作，会将等号右边的值赋值给等号左边的变量。而`==`表示的是判断两边的值是否相等，也就是`等于`的意思。

单if

单`if`流程控制语句的伪代码格式为：

```
如果 条件:
缩进 结果
```

具体示例如下：

```
sex = "男"
if sex == "男":
    print("就来看我")
print(sex)

输出的结果为:
就来看我
男
```

if else 二选一

`if else`二选一流程控制语句的伪代码格式为:

```
如果 条件:
缩进 结果
否则:
缩进 结果
```

具体示例如下:

```
print(111)
if 3>2:
    print(11)
    print(22)
else:
    print(333)
print(444)

打印出的结果为:
111
11
22
444
```

if elif elif 多选一或零

对于`if elif elif`多选一或零流程控制语句而言，只要有一个条件成立，其他的语句将不被执行。其伪代码格式为:

```
如果 条件:
缩进 结果
再如果 条件:
缩进 结果
再如果 条件:
缩进 结果
再如果 条件:
缩进 结果
```

具体示例如下:

```
if 3>5:
    print(1)
elif 3>7:
    print(2)
elif 5>2:
    print(4)
elif 3>1:
    print(5)

输出结果为:
4
```

if elif elif else 多选一

`if elif elif else`多选一流程控制语句的伪代码格式为:

```
如果 条件:
缩进 结果
再如果 条件:
缩进 结果
再如果 条件:
缩进 结果
否则:
缩进 结果
```

具体示例如下:

```
if 3>12:
   print(1)
elif 3>11:
    print(2)
elif 4>12:
    print(3)
else:
    print(5)

输出的结果为:
5
```

if 嵌套:

`if`嵌套流程控制语句的伪代码格式为:

```
如果 条件:
缩进 如果 条件:
      缩进 结果
```

具体示例如下:

```
sex = "男"
age = 48
if sex == "女":
    if age == 18:
        print("进来坐坐")
    else:
        print("隔壁找三哥")
else:
    print("去对门找yuan")

输出结果为:
去对门找alex
```

if if if 多选多

`if if if`多选多流程控制语句的伪代码格式为:

```
如果 条件:
缩进 结果
如果 条件:
缩进 结果
如果 条件:
缩进 结果
```

具体示例如下:

```
if 43>1:
    print(11)
if 43>2:
    print(11)
if 43>3:
    print(11)

输出的结果为:
11
11
11
```

补充内容:`and`是`和`的意思。只有当`and`两端的值都为`真`时,会返回`True`,否则都会返回`False`:

```
user = input("username:")
pwd = input("password:")
and 是和
if user == "yuan" and pwd == "yuan123":
    print(111)
```

7、python3 与python2 的区别

python2:

- 源码不统一
- 有重复代码
- 整型的除法:整型
- print 不加括号也可以

- input() 输入什么类型是什么类型
- raw_input() 获取到的都是字符串

python3

- 源码统一
- 没有重复代码
- 整数的除法：浮点数（小数）
- print()
- input() 获取到的都是字符串

8、今日内容总结

1.变量命名规则 (重要)
  1.数字,字母,下划线组成
  2.不能以数字开头
  3.禁止使用python中的关键字
  4.变量名要具有可描述性
  5.变量名要区分大小写
  6.不能使用中文和拼音
  7.推荐写法:
    7.1 驼峰体
    7.2 下划线 (官方推荐)

2.常量: 用于配置文件使用
3.注释:单行注释 # 多行注释 """ """ (建议使用""" """)

4.基础数据类型:
  int -- 整型: 用于计算和比较的
    + - * / > < ==(等于)

```
str -- 字符串: 存储数据的
    +拼接: 同类型
    *拼接: 字符串乘数字

bool -- 布尔值:判断真假
    True  - 真
    False - 假
int("字符串")
str(整型)
```

5.用户交互(input)
  python2和python3的区别
  input:获取到的内容都是字符串
  type() 查看数据类型

6.流程控制语句 if

```
单if
if else
if elif elif
if elif elif else
if 嵌套
if if
```