

《Python程序设计》

Python数据类型

刘潇

机械科学与工程学院

2023年10月12日

2023秋季

本节要点

- **掌握数字类型的常用运算符、数学函数和模块**
- **掌握字符串类型的常用操作、方法、模块和格式化打印方法**
- **理解Python的不可变数据对象**

主要内容

1. Python内置数据类型

2. 数字类型

3. 字符串类型

Python数据类型

Python中常用的内置数据类型

		数据类型	类型名称	示例
		数字	int, float, complex	12, 1.1, 1.2e3, 1+2j
		字符串 (序列)	str	"hello world" , 'python'
容器	{	列表 (序列)	list	[1, 2, 3], ['a' , 'b' , 'c']
		元组 (序列)	tuple	(1, 2, 3), ('a' , 'b' , 'c')
		字典 (映射)	dict	{ 'username' : 'admin' , 'age' : 20 }

使用 `type()` 函数获取任何对象的数据类型

数字

- **整数(int)**: 与数学中整数的概念一致, 可正可负, 没有取值范围限制
- **浮点数(float)**: 包含小数的正数和负数, 可用科学计数法表示

数字类型转换

- **复数(complex)**: 用j表示虚部

用int(), float()和complex()实现
数字类型转换:

复数无法转化为其他类型

```
1 a = 10 # int
2 b = 5.5e5 # float
3 c = 1 + 2j # complex
4
5 #把整数转化为浮点数
6 print(float(a))
7 print(type(float(a)))
8
9 #把整数转化为复数
10 print(complex(a))
11 print(type(complex(a)))
12
13 #把浮点数转化为整数
14 print(int(b))
15 print(type(int(b)))
```

其他进制

- 十进制: 1010, 99, -217
- 二进制: 以0b或0B开头, 0b010, -0B101
- 八进制: 以0o或0O开头, 0o123, -0o456
- 十六进制: 以0x或0X开头, 0x9a, -0X89

其他进制

```
1 a = 1
2 print(a)
3 print(0b010)
4 print(0o123)
5 print(0x9a)
6
7 #移位运算符 <<
8 print(a<<2)
```

二进制: 0001



向左移位 << 2

二进制: 0100

1
2
83
154
4

利用bin、oct、hex函数转换数据类型打印其他进制数

运算符

算术运算符		比较运算符		赋值运算符	
+	加	==	等于	=	简单的赋值运算符
-	减	!=	不等于	+=	加法赋值运算符
*	乘	>	大于	-=	减法赋值运算符
/	除	<	小于	*=	乘法赋值运算符
%	取模	>=	大于等于	/=	除法赋值运算符

逻辑运算符

and	x and y	如果两个语句都为真，则返回 True
or	x or y	如果其中一个语句为真，则返回 True
not	not x	反转结果，如果结果为 true，则返回 False

位运算符

&	AND	如果两个位均为 1，则将每个位设为 1。
	OR	如果两位中的一位为 1，则将每个位设为 1。
<<	Zero fill left shift	通过从右侧推入零来向左移动，推掉最左边的位。

四则运算

四则运算_1

```
1  #Python3. x中1/2 = 0.5
2  a = 1
3  b = 2
4  print(a/b)
5
6  #运算顺序
7  c = 3
8  print(a+b*c)
9  print((a+b)*c)
10
11 print(-3**2)
12 print((-3)**2)
13
14 #不同类型间混合运算
15 e = 2.0
16 print(a+e)
```

除法运算结果为浮点数

与数学运算一致，圆括号改变顺序

乘方优先级高于求负！

运算结果为最宽的数字类型

整数 → 浮点数 → 复数

0.5
7
9
-9
9
3.0

四则运算

四则运算_2

```
1 #整除
2 print(10//3) #向下圆整
3 #取余
4 print(10%3)
5
6 #负数整除
7 print(10// -3)
8 print(-10//3)
9 print(-10// -3)
10 #负数取余
11 print(10% -3)
12 print(-10%3)
13 print(-10% -3)
```

```
3
1
-4
-4
3
-2
2
-1
```

整除时向下圆整，当结果为负时，圆整后将离0更远

四则运算

四则运算_3

```
1 #浮点数间运算存在不确定尾数
2 a = 0.1
3 b = 0.2
4 c = 0.3
5 print(a+b)          0.30000000000000004
6 print(a+c)          0.4
7 print(a+b == 0.3)   False
8 print(round(a+b,1) == 0.3) True
```

浮点数间运算存在不确定尾数

round(x,d)对x四舍五入，d是小数截取位数

二进制 0.0001**1001**10011001100110011001100110011001100110011001101

十进制 **0.10000000000000000**055511151231257827021181583404541015625

默认的是17位小数的精度

0.1的二进制表示小数，可以无限接近，但不完全相同

内置数学函数

函数及使用	描述
<code>divmod(x, y)</code>	商余, $(x//y, x\%y)$, 同时输出商和余数 <code>divmod(10, 3)</code> 结果为(3, 1)
<code>abs(x)</code>	x的绝对值 <code>abs(-10.01)</code> 结果为10.01
<code>pow(x, y[,z])</code>	幂余, $(x**y)\%z$, [...]表示参数z可省略 <code>pow(3, pow(3, 5), 100)</code> 结果为27
<code>round(x[, d])</code>	四舍五入, d是保留的小数位数, 默认值为0 <code>round(-10.123, 2)</code> 结果为-10.12
<code>max(x₁, x₂, ..., x_n)</code>	最大值, 返回x ₁ , x ₂ , ..., x _n 中的最大值, n不限 <code>max(1, 9, 5, 4, 3)</code> 结果为9
<code>min(x₁, x₂, ..., x_n)</code>	最小值, 返回x ₁ , x ₂ , ..., x _n 中的最小值, n不限 <code>min(1, 9, 5, 4, 3)</code> 结果为1

数学模块

math是Python提供的内置数学模块，提供了4个数学常数和44个函数

导入math模块

```
import math
```

以**模块名.函数名**的形式使用

```
from math import <函数名> # 直接使用函数名 （无法使用多个模块中具有相同名字的函数）
```



```
C:\Windows\system32\cmd.exe - python
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\xliu>python
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import math
>>> dir(math)
['__doc__', '__loader__', '__name__', '__package__', '__spec__', 'acos', 'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil', 'comb', 'copysign', 'cos', 'cosh', 'degrees', 'dist', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs', 'factorial', 'floor', 'fmod', 'frexp', 'fsum', 'gamma', 'gcd', 'hypot', 'inf', 'isclose', 'isfinite', 'isinf', 'isnan', 'isqrt', 'ldexp', 'lgamma', 'log', 'log10', 'log1p', 'log2', 'modf', 'nan', 'perm', 'pi', 'pow', 'prod', 'radians', 'remainder', 'sin', 'sinh', 'sqrt', 'tan', 'tanh', 'tau', 'trunc']
>>> print(math.__doc__)
This module provides access to the mathematical functions
defined by the C standard.
>>> _
```

dir() 查看属性和方法

4个数学常数

math.pi （圆周率）

math.e （自然对数）

math.inf （正无穷大）

math.nan （非浮点数标记）

利用限定符调用属性和方法

math模块函数

16个数值表示函数

<code>fabs</code>	绝对值
<code>fmod</code>	模
<code>fsum</code>	求和
<code>ceil</code>	向上取整
<code>floor</code>	向下取整
<code>factorial</code>	阶乘
<code>gcd</code>	最大公约数
<code>frep(x)</code>	$x = m * 2^e$
<code>ldexp(x,i)</code>	$x * 2^i$
<code>modf</code>	返回小数和整数
<code>trunc</code>	返回整数
<code>copysign(x,y)</code>	$ x * y / y$
<code>isclose</code>	相似性
<code>isfinite</code>	是否无穷大
<code>isinf</code>	正数或负数无穷大
<code>isnan</code>	是否NaN

16个三角双曲函数

<code>degree</code>	弧度转角度
<code>radians</code>	角度转弧度
<code>hypot(x,y)</code>	$\sqrt{(x^2+y^2)}$
<code>sin</code>	正弦
<code>cos</code>	余弦
<code>tan</code>	正切
<code>asin</code>	反正弦
<code>acos</code>	反余弦
<code>atan</code>	反正切
<code>atan2(y/x)</code>	y/x 的反正切
<code>sinh</code>	双曲正弦
<code>cosh</code>	双曲余弦
<code>tanh</code>	双曲正切
<code>asinh</code>	反双曲正弦
<code>acosh</code>	反双曲余弦
<code>atanh</code>	反双曲正切

8个幂对数函数

<code>pow(x,y)</code>	x^y
<code>exp(x)</code>	e^x
<code>expm1(x)</code>	$e^x - 1$
<code>sqrt(x)</code>	\sqrt{x}
<code>log(x,base)</code>	$\log_{base}(x)$
<code>loglp(x)</code>	$\ln(1+x)$
<code>log2</code>	$\log_2(x)$
<code>log10</code>	$\log_{10}(x)$

4个高等特殊函数

<code>erf(x)</code>	$\frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$
<code>erfc(x)</code>	$\frac{2}{\sqrt{\pi}} \int_0^\infty e^{-t^2} dt$
<code>gamma(x)</code>	$\int_0^\infty x^{t-1} e^{-x} dx$
<code>lgamma(x)</code>	$\ln(\text{gamma}(x))$

random模块

random模块用于生成随机数

random模块

```
1 import random
2 print( random.randint(1,10) )      # 产生 1 到 10 的一个整数型随机数
3 print( random.random() )          # 产生 0 到 1 之间的随机浮点数
4 print( random.uniform(1.1,5.4) )  # 产生 1.1 到 5.4 之间的随机浮点数, 区间可以不是整数
5 print( random.choice('tomorrow') ) # 从序列中随机选取一个元素
6 print( random.randrange(1,100,2) ) # 生成从1到100的间隔为2的随机整数
7
8 a=[1,3,5,6,7]                      # 将序列a中的元素顺序打乱
9 random.shuffle(a)
10 print(a)
```

```
1
0.8274195652516325
3.8248037463924995
r
31
[6, 7, 3, 1, 5]
```

计算圆周率

随机抛洒DARTS个点，计算点到圆心的距离，统计圆内点的数量

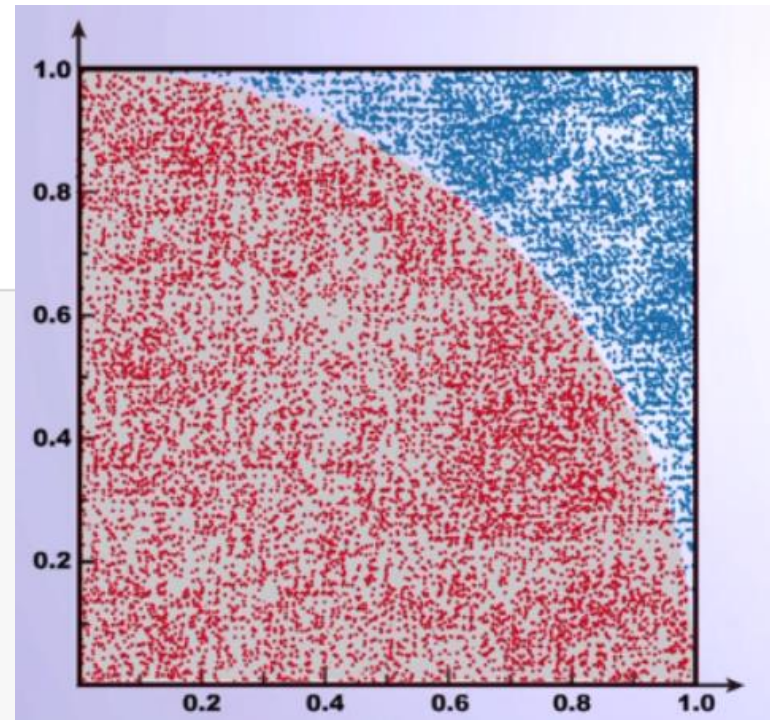
math和random模块计算圆周率

```
1 import random, math, time
2 DARTS = int(2**25)
3 hits = 0
4 start_time = time.perf_counter()
5 for i in range(1, DARTS):
6     x, y = random.random(), random.random()
7     dist = math.sqrt(x**2 + y**2)
8     if dist <= 1.0:
9         hits = hits + 1
10 pi = 4 * (hits/DARTS)
11 print("Pi的值是", pi)
12 during_time = time.perf_counter() - start_time
13 print("程序运行时间是", during_time, "s")
```

Pi的值是 3.141368865966797

程序运行时间是 25.428142207000064 s

蒙特卡洛方法



$\text{hits/DARTS} = \frac{1}{4}\text{圆面积}$

字符串

字符串是字符的**序列**，单行字符串由一对单引号或双引号表示，多行字符串可由一对三单引号或三双引号表示

字符串

```
1  #三单引号注释
2  '''
3  print("hello world!")
4  '''
5  #三单引号字符串赋值
6  a = '''
7  print("hello world!")
8  '''
9  print(a)
10
11 print("这里有个单引号' ")
12 print(' 这里有个双引号"')
13 print('\' 这里既有单引号\' , 还有双引号"\'\'')
```

三引号还可以在程序中表示多行注释

单引号、双引号、三单引号、三双引号可以互相嵌套，来表示复杂字符串

```
print("hello world!")
```

这里有个单引号'

这里有个双引号"

这里既有单引号' , 还有双引号"

转义字符

用转义符(\)在字符串中表达一些不可直接打印的信息

常用转义字符

\b	退格
\f	换页符
\n	换行符
\r	回车
\t	水平制表符
\v	垂直制表符
\\	一个斜线\
\'	单引号'
\"	双引号"
\ooo	3位八进制对应的字符
\xhh	2位十六进制对应的字符
\uhhhh	4位十六进制表示的 unicode字符

转义字符

```
1 print("Let's go!")
2 print('Let\'s go!')
3 print('Let\'s say "hello world!"')
4 print("C:\\Windows\\System32\\nrvivers")
5
6 #退格符(光标向前移动一格)
7 print("hello\bworld!")
8 #换行符(光标移动到下行首)
9 print("hello \nworld!")
10 #回车符(光标移动到本行首)
11 print("hello \rworld!")
12
13 #表达式换行
14 print \
15 ("hello world!")
16 print(1 + 2 \
17 + 3 + 4)
```

```
Let's go!
Let's go!
Let's say "hello world!"
C:\Windows\System32\nrvivers
hellworld!
hello
world!
world!
hello world!
10
```

原始字符串 r" "

字符串操作

字符串操作包括：索引、切片、相加、相乘和成员资格检查，
适用于所有序列类型（列表、元组）

字符串索引从0开始，一个
长度为L的字符串最后一个
字符的位置是L-1

允许使用负数从字符串右边
末尾向左边进行反向索引，
最右侧索引值是-1



索引

字符串[M]: 返回字符串中序号为M的单个字符 字符串索引

```
1 a = "hello world!"
2 print(a[1], a[5])
3 print(a[-2])
4
5 year = input("input Year:")
6 print(year[3])
7 print(type(year[3]))
8 print(year[4])
```

```
e
d
input Year:2020
0
<class 'str'>
```

返回为字符串类型

```
-----
IndexError                                Traceback (most recent call last)
<ipython-input-11-c7d54d279e27> in <module>
      6 print(year[3])
      7 print(type(year[3]))
----> 8 print(year[4])
```

```
IndexError: string index out of range
```

超出序号范围

切片

字符串[M:N]：返回字符串中特定范围内的字符

字符串切片

```
1 a = "hello world!"
2 print(a[1:3])
3 #字符串的长度
4 len(a)
5
6 tag = '<a href="http://www.python.org">Python web site</a>'
7 print(tag[9:30])
8 print(tag[32:-4])
9 #索引到末尾
10 print(tag[32:])
11 #从头索引
12 print(tag[:9])
13
14 #更大的步长
15 number = "123456789"
16 print(number[1:9:2])
17 print(number[9:1:-2])
18 print(number[::-2])
19 print(number[::-2])
```

**第一个索引指定的字符包含在切片内，
第二个索引指定的字符不包含**

**更大的步长，字符串[M:N:K]，K为
步长，默认值为1**

**步长为负数时，第一个索引必须比第
二个索引大**

相加和相乘

相加和相乘

```
1 a = "hello,"  
2 b = "world!"  
3 print(a+b)  
4 print(a*5)
```

相加：利用加法算符来**拼接字符串**

相乘：字符串与数x相乘时，将**重复字符串x次**

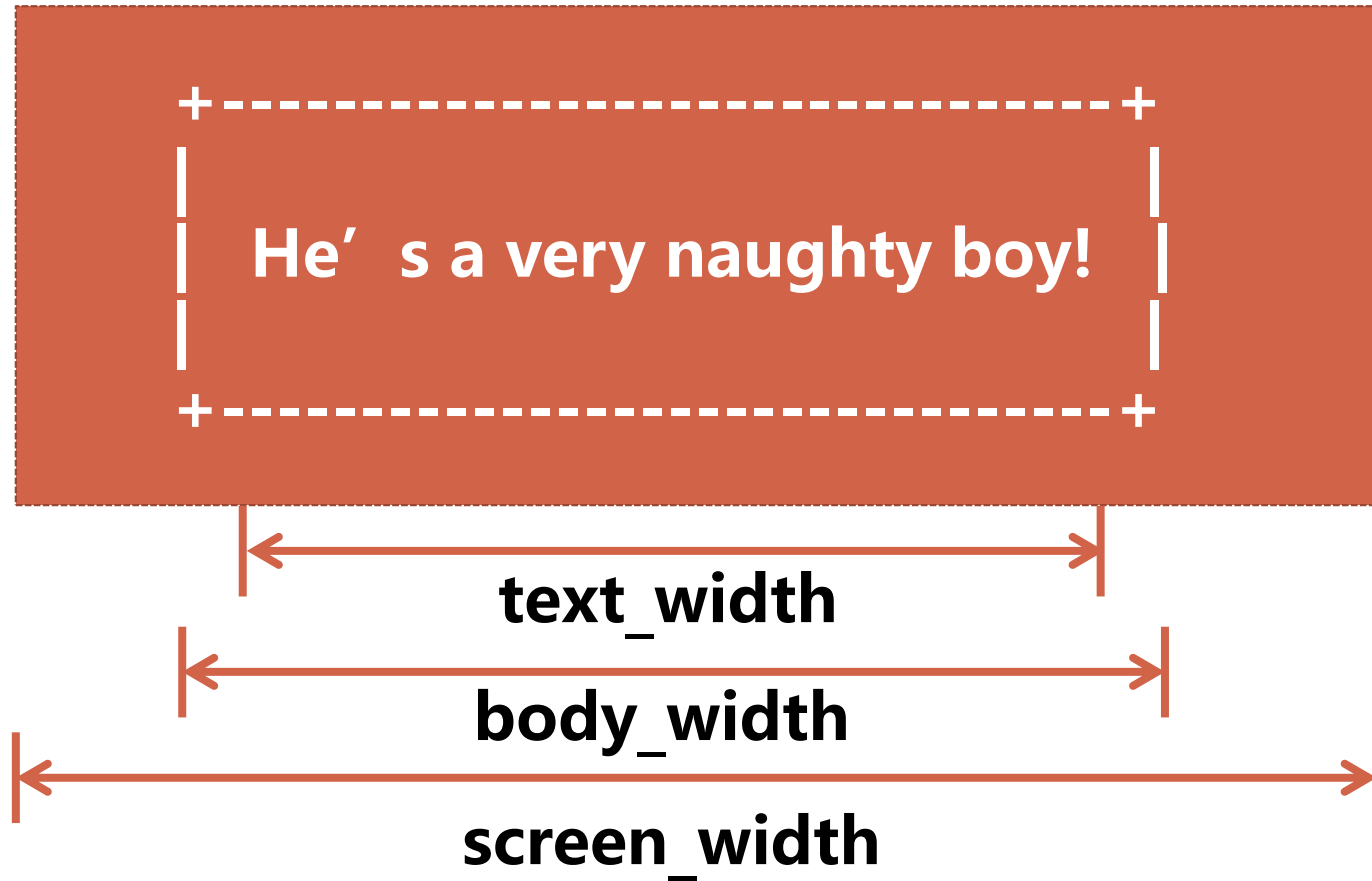
```
5  
6 #短字符串缓存, 长字符串分配内存  
7 c = a*1  
8 d = a*1  
9 e = a*5  
10 f = a*5  
11 print(id(a), id(c), id(d), id(e), id(f))  
12  
13 #重新赋值重新申请内存  
14 a = "replace"  
15 print(a, c, d)  
16 print(id(a), id(c), id(d))
```

内存管理

```
hello, world!  
hello, hello, hello, hello, hello,  
97927728 97927728 97927728 98279056 98511488  
replace hello, hello,  
32284400 97927728 97927728
```

相乘示例

在位于屏幕中央且宽度合适的方框内打印一个句子



成员资格检查

使用运算符in检查字符串x是否在字符串s序列中，如果x是s的子串返回True，否则返回False

成员资格检查

```
1 s = "hello world!"
2 print("e" in s)
3 print("el" in s)
4
5 for ch in "hello world!":
6     print(ch, end = " ")
```

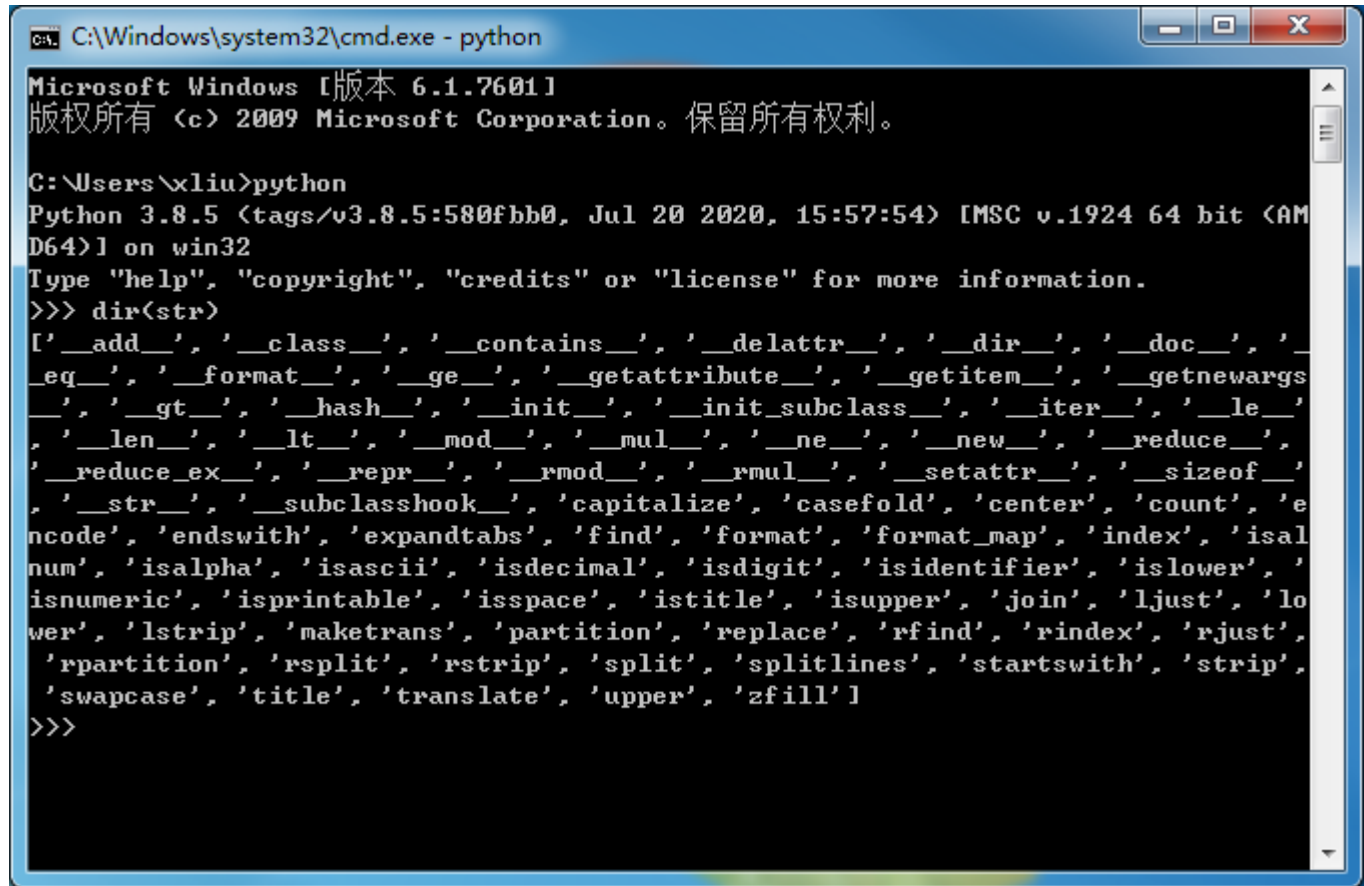
```
True
True
h e l l o   w o r l d !
```

通过for和in组成的循环来遍历字符串中的每个字符

字符串方法

方法：对象的函数，第一个参数是默认的self

字符串类：



```
C:\Windows\system32\cmd.exe - python
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

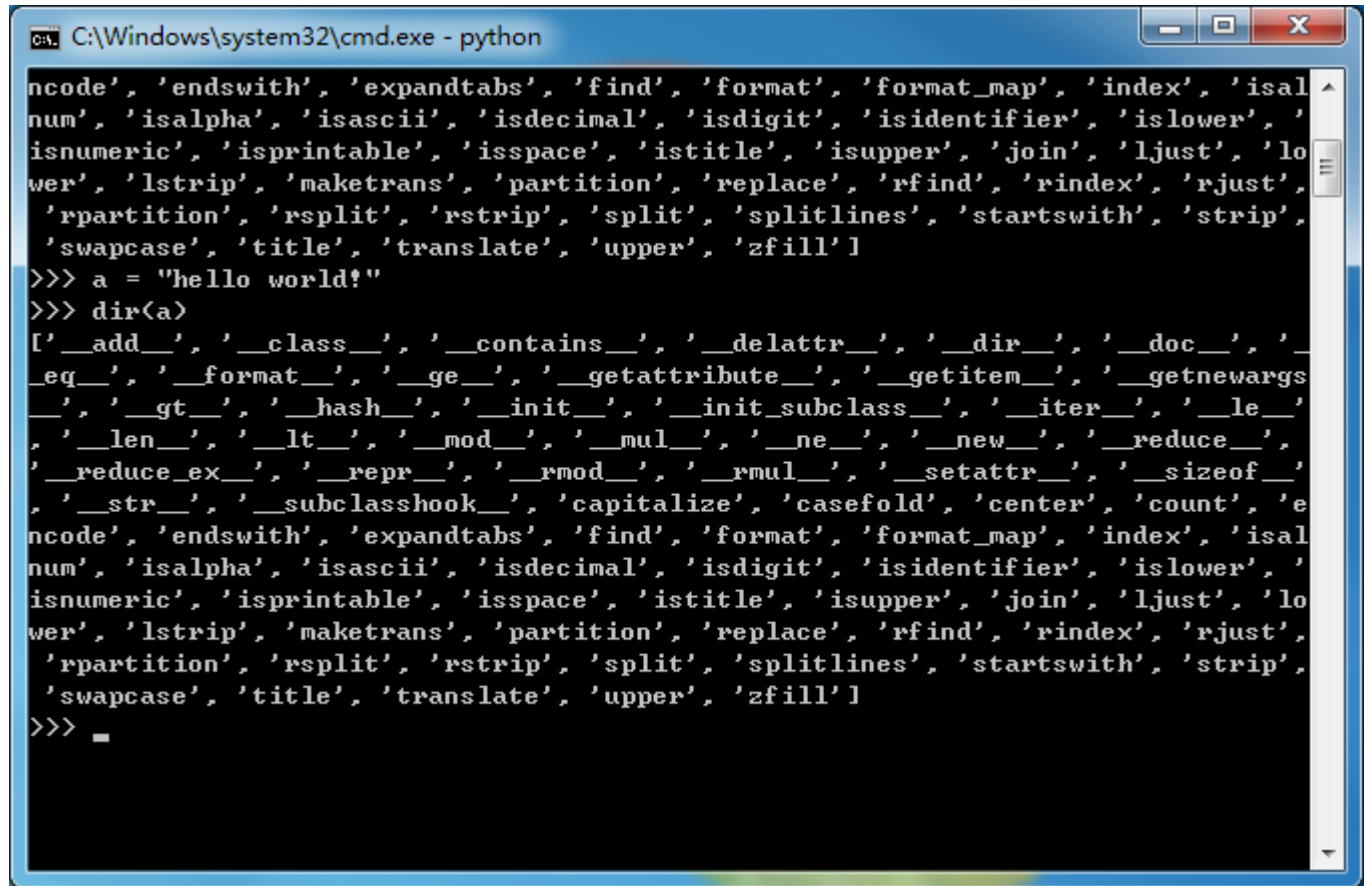
C:\Users\xliu>python
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> dir(str)
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
>>>
```


字符串对象

对象继承字符串类的属性和方法

方法的使用：“字符串a”.方法b()

字符串对象：



```
C:\Windows\system32\cmd.exe - python
ncode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isal
num', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', '
isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lo
wer', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
'partition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip',
'swapcase', 'title', 'translate', 'upper', 'zfill']
>>> a = "hello world!"
>>> dir(a)
['_add_', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__
eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs
__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le_
', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__',
 '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__
', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'count', 'e
ncode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isal
num', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', '
isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lo
wer', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
'partition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip',
'swapcase', 'title', 'translate', 'upper', 'zfill']
>>> _
```

Jupyter中可以利用Tab键弹出常用的方法

常用的字符串方法

center: 通过在两边添加填充字符，让字符串居中

str.center(width, fillchar) (fillchar默认为空格)

字符串方法center

```
1 text = input("Please input text:")
2
3 screen_width = 100
4 text_width = len(text)
5 body_width = (screen_width + text_width)/2
6
7 first_line = "+" + "-".center(int(body_width - 2), "-") + "+"
8 second_line = "| " + ".center(int(body_width - 2)) + "| "
9 third_line = "| " + text.center(int(body_width - 2)) + "| "
10
11 #打印格式化文本
12 print(first_line.center(screen_width))
13 print(second_line.center(screen_width))
14 print(third_line.center(screen_width))
15 print(second_line.center(screen_width))
16 print(first_line.center(screen_width))
```

常用的字符串方法

方法及使用	描述
str.find()	在字符串中查找子串。如果找到，就返回子串的第一个字符的索引，否则返回-1 <code>a = "hello world!"</code> <code>a.find("ll")</code> 结果为2
str.join()	在字符串除最后一个元素外每个元素后增加一个str <code>",".join("12345")</code> 结果为 <code>"1,2,3,4,5"</code>
str.split(sep)	返回一个列表，由str根据sep被分隔的部分组成 <code>"1,2,3,4,5".split(",")</code> 返回为 <code>["1" , "2" , "3" , "4" , "5"]</code>
str.lower()或者str.upper()	全部字符小写或者大写 <code>"AbCdefG".lower()</code> 结果为 <code>"abcdefg"</code>
str.strip(chars)	从str中去掉在其左侧和右侧chars中列出的字符 <code>"=python=" .strip("=np")</code> 结果为 <code>"ytho"</code>

字符串替换

str.replace()

将指定子串替换为另一个字符串，并返回替换后的结果

"This is a test".replace("is", "eez") 结果为
"Theez eez a test"

字符串替换replace

```
1 a = "This is a test"
2
3 #将a中的is替换为eez
4 b = a.replace("is", "eez")
5 print(b, id(a), id(b))
6
7 #能不能利用索引替换?
8 print(a[2:4])
9 a[2:4] = "eez"
```

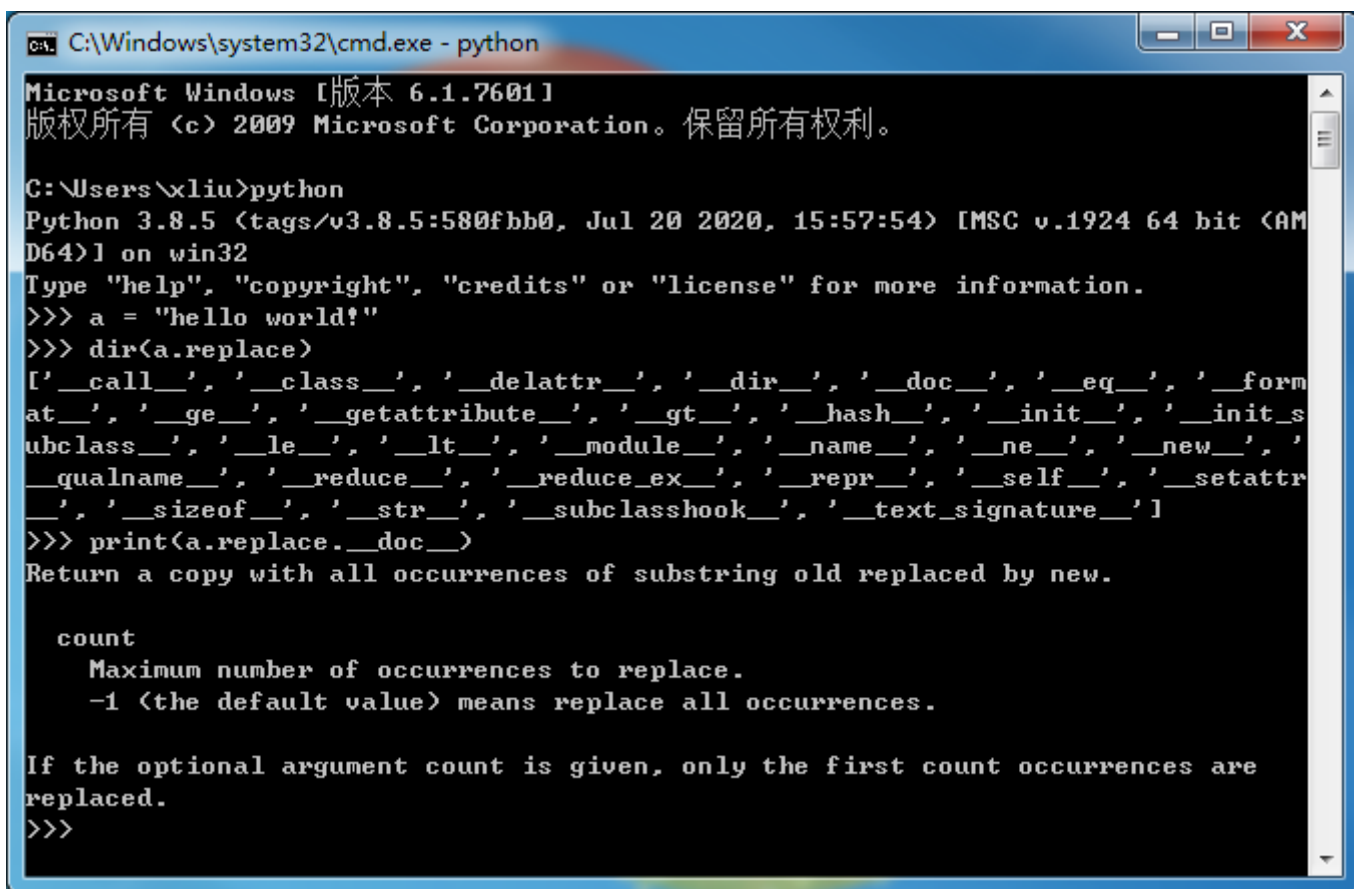
print(a) ?
字符串是不可变序列

```
Theez eez a test 98558704 95874944
is
```

```
-----
TypeError                                Traceback (most recent call last)
<ipython-input-14-6a1bbf7d9fca> in <module>
      7 #能不能利用索引替换?
      8 print(a[2:4])
----> 9 a[2:4] = "eez"

TypeError: 'str' object does not support item assignment
```

查看字符串方法的属性



```
C:\Windows\system32\cmd.exe - python
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\xliu>python
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> a = "hello world!"
>>> dir(a.replace)
['__call__', '__class__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattr__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__le__', '__lt__', '__module__', '__name__', '__ne__', '__new__', '__qualname__', '__reduce__', '__reduce_ex__', '__repr__', '__self__', '__setattr__', '__sizeof__', '__str__', '__subclasshook__', '__text_signature__']
>>> print(a.replace.__doc__)
Return a copy with all occurrences of substring old replaced by new.

    count
    Maximum number of occurrences to replace.
    -1 (the default value) means replace all occurrences.

If the optional argument count is given, only the first count occurrences are
replaced.
>>>
```

**print("This is a test".replace("is", "eez", 1))结果为
"Theez is a test"**

字符串模块

Python3中字符串的很多方法是从string模块中继承的，string模块中还包含几个常量

string模块常量	描述
string.digits	包含数字0~9的字符串
string.ascii_letters	包含所有ASCII字母（大写和小写）的字符串
string.ascii_lowercase	包含所有小写ASCII字母的字符串
string.printable	包含所有可打印的ASCII字符的字符串
string.punctuation	包含所有ASCII标点符号的字符串

re模块

re模块提供对正则表达式的支持，正则表达式通常被用来检索、替换那些**符合某个模式(规则)**的文本

正则表达式	待匹配字符	匹配结果	描述
python	python	python	普通字符串是最简单的正则表达式
pyt.	python	pyth	匹配除换行符以外的任意的一个字符（通配符）
python.*	python	python	*表示重复零次或多次
python.+	python	不匹配	+表示重复一次或多次
pyth.{1,2}	python	python	{1,2}匹配1到2次任意字符
pyth.{1,2}?	python	pytho	?惰性匹配

re模块中的函数

re.match()从字符串起始位置开始匹配子串，匹配成功返回匹配信息，不成功返回None

re模块 match函数

```
1 import re
2 a = "python"
3 print(re.match("python", a))
4 print(re.match("pyt.", a))
5 print(re.match("python.*", a))
6 print(re.match("python.+", a))
7 print(re.match("pyth. {1,2}", a))
8 print(re.match("pyth. {1,2}?", a))
9 print(re.match("pyth. {1,2}?", a).end())
```

```
<re.Match object; span=(0, 6), match='python'>
<re.Match object; span=(0, 4), match='pyth'>
<re.Match object; span=(0, 6), match='python'>
None
<re.Match object; span=(0, 6), match='python'>
<re.Match object; span=(0, 5), match='pytho'>
5
```



**re.match().span()
re.match().start()
re.match().end()
re.match().group()**

re模块中的函数

re.search()扫描整个字符串，并返回第一个成功的匹配。如果匹配失败，则返回None

re模块 search函数

```
1 import re
2 a = "python"
3 print(re.search("python", a))
4 print(re.match(".hon", a))
5 print(re.search(".hon", a))
6 print(re.search(".*python", a))
7 print(re.search(".*python", a))
8 print(re.search(".{1,2}thon", a))
9 print(re.search(".{1,2}?thon", a))
```

match()适合检查某个字符串的开头是否符合某个规定

```
<re.Match object; span=(0, 6), match='python'>
None
<re.Match object; span=(2, 6), match='thon'>
<re.Match object; span=(0, 6), match='python'>
None
<re.Match object; span=(0, 6), match='python'>
<re.Match object; span=(0, 6), match='python'>
```

其他正则表达式

正则表达式	可匹配字符	描述
<code>python\\.org</code>	<code>python.org</code>	转义字符 <code>\</code> ，或用原始字符串 <code>r ' '</code>
<code>[pj]ython</code>	<code>python</code> 或 <code>jython</code>	字符集只匹配一个字符， <code>[a-z]</code> ， <code>[a-zA-Z0-9]</code> ， <code>[^abc]</code>
<code>python perl</code>	<code>python</code> 或 <code>perl</code>	二选一或者子模式
<code>r '(http://)?(www\.)?python\.org'</code>	<code>http://www.python.org</code> 或 <code>http://python.org</code> 或 <code>www.python.org</code>	<code>()?</code> 可选字符串放在圆括号中，可出现或不出现
<code>^ht+p</code>	<code>http://python</code> 或 <code>http://python</code>	<code>^</code> 指定字符串开头， <code>\$</code> 指定字符串结尾

其他正则表达式

`^[A-Z]:\\{1,2}[^/:*\\?<>\\|\\]+\\.(jpg|gif|png|bmp)$`

re模块 正则表达式

```
1 pattern = re.compile(r'^[A-Z]:\\{1,2}[^/:\*\\?<>\\|\\]+\\.(jpg|gif|png|bmp)$', re.I)
2 a = r"c:\\123.bmp"
3 b = r"d:\\123.bmp"
4 c = r"E:\\test\\123.bmp"
5
6 print(pattern.match(a))
7 print(pattern.match(b))
8 print(pattern.match(c))
```

<re.Match object; span=(0, 10), match='c:\\123.bmp'>

<re.Match object; span=(0, 11), match='d:\\\\123.bmp'>

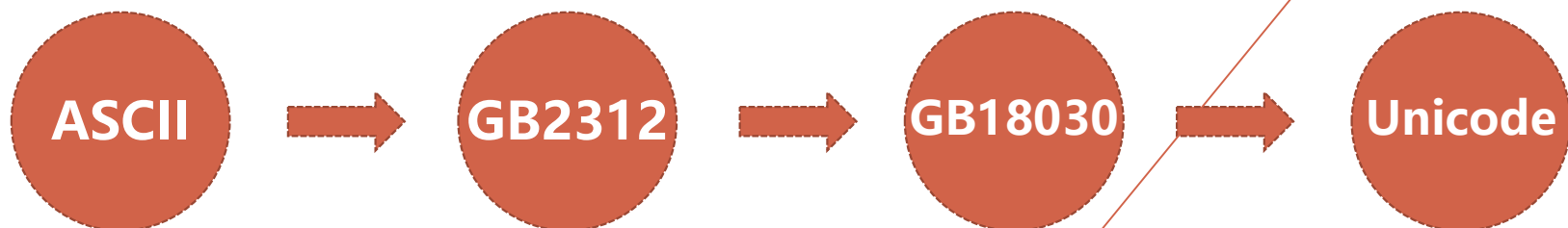
None

Unicode字符

编码过程是将字符集转换为计算机所能理解的二进制码

7位二进制表示英文字母、
数字和标点字符等

国家标准，采用多字节进行编码，
收入共收录70244个汉字



国家标准，采用两个字节对字符
集进行编码，并向下兼容ASCII编
码方式，收入6763个汉字和682
个非汉字图形字符个

国际标准，涵盖了世界上所有
的文字和符号字符，含有17各
组，每个组有65536个码位，
每个码位对应一个字符

需要编码和解码!!!

本地码



国际编码

UTF-8编码是针对不同范围的字符代码以字节为单位转化成不同长度的字符编码

Python3中字符串使用Unicode编码来表示文本

Unicode编码转换 (<https://unicode-table.com/en/>)

```
1 print("\u02B1")
2 print("\u2660")
3 print("\u21A2")
4 print("\u4E25")
5
6 #编码与解码
7 print("\u4E25".encode("utf-8"))
8 print("严".encode("utf-8"))
9 print(b'\xe4\xb8\xa5'.decode())
10
11 print("中国".encode("utf-8"))
12 print(b'\xe4\xb8\xad\xe5\x9b\xbd'.decode())
```

Unicode编码

♠

♠

←

严

b'\xe4\xb8\xa5'

b'\xe4\xb8\xa5'

严

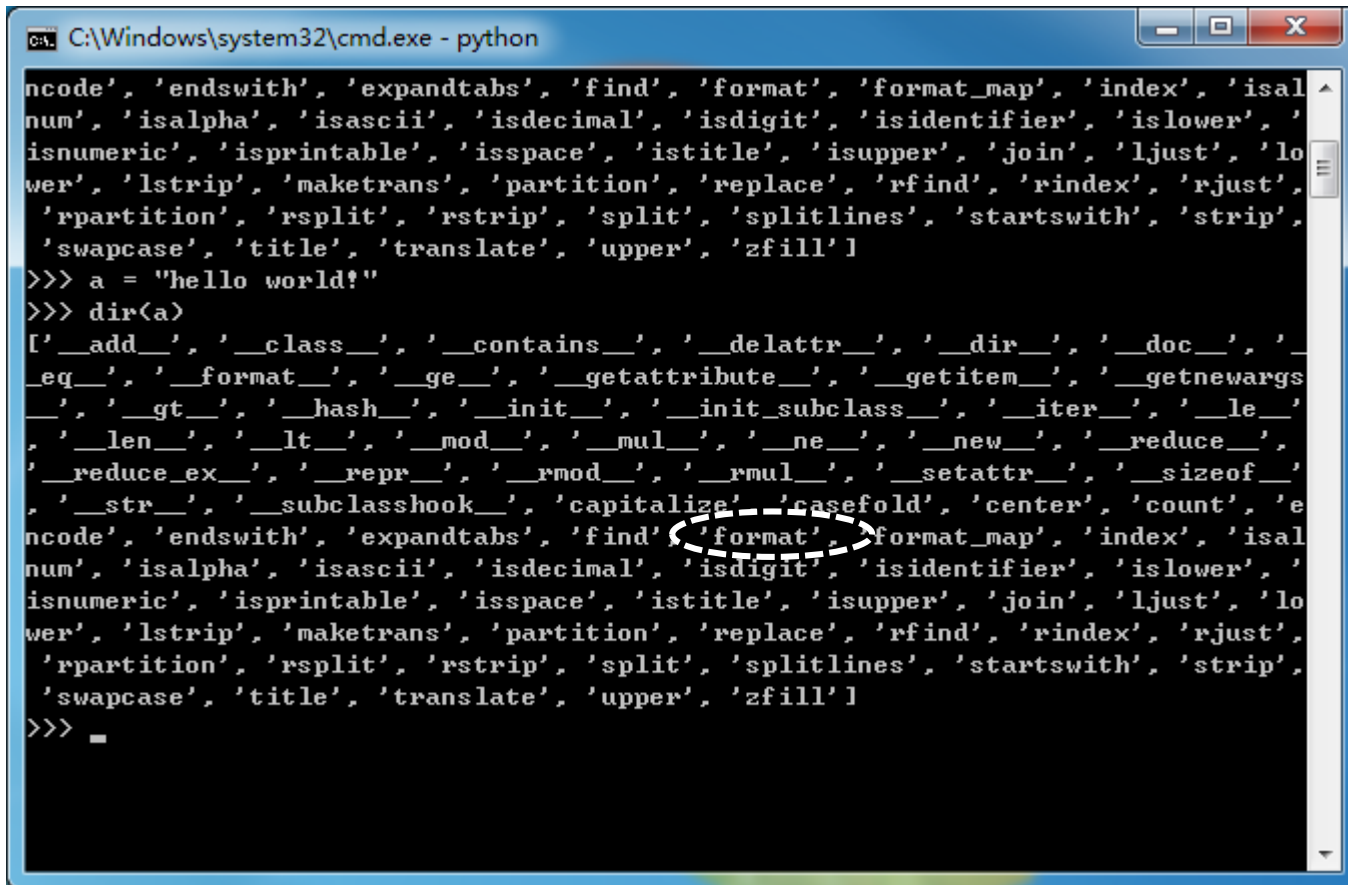
b'\xe4\xb8\xad\xe5\x9b\xbd'

中国

三个字节

六个字节

格式化字符串



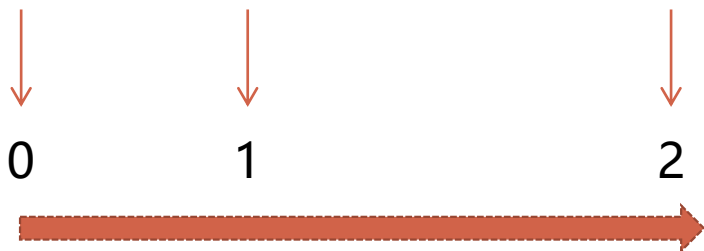
```
C:\Windows\system32\cmd.exe - python
ncode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isal
num', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', '
isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lo
wer', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
'partition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip',
'swapcase', 'title', 'translate', 'upper', 'zfill']
>>> a = "hello world!"
>>> dir(a)
['_add_', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '_
eq_', '__format__', '__ge__', '__getattr__', '__getitem__', '__getnewargs
__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le_
', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__',
 '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__
', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'count', 'e
ncode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isal
num', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', '
isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lo
wer', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
'partition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip',
'swapcase', 'title', 'translate', 'upper', 'zfill']
>>> _
```

对字符串调用format方法，以设定的格式表示：

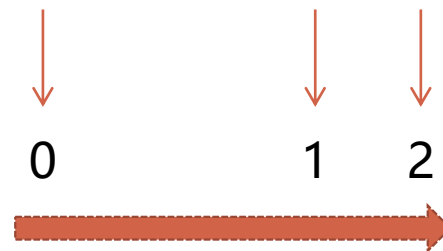
模板字符串.format(逗号分隔的参数)

替换字段

`"{}:计算机{}的CPU占用率为{}%".format("2020-09-01", "C", 10)`



字符串中未命名字段默认顺序



参数默认顺序

`"{1}:计算机{2}的CPU占用率为{0}%".format(10, "2020-09-01", "C")`

手动编号

`"{para1}:计算机{para2}的CPU占用率为{}%".format(10, para1 = "2020-09-01", para2 = "C")`

指定参数名称

格式说明符

指定最终格式包括：格式类型（字符串、浮点数、十六进制等）、字段宽度、数的精度、对齐填充方式等

{参数序号或名字：格式说明符}

:	填充	对齐	宽度	,	.精度	类型
引导符号	用于填充的单个字符	<左对齐 >右对齐 =居中对齐	设定总的输出宽度	数字的千分位分隔符	浮点数小数精度或字符串最大输出长度	整数类型 (b,c,d,o,x,X) 浮点 数类型 (e,E,f,%)

"{0:= >40}".format("Hello world!")

"{pi:,.10f}".format(pi = 3.14)

课后作业-格式化输出价格列表

课后作业1 格式化打印价格列表，请补全如下代码

```
1  #定义各输出字段的宽度
2  width = input("请输入价格表总字符宽度: ")
3  price_width = 10
4  #####补充代码#####
5
6  #####
7  item_width = width - price_width
8
9  #####补充代码#####
10 #定义格式化方式
11 head =
12 body =
13 #####
14
15 #打印价格列表
16 print("="*width)
17 print(head.format("Item", "Price"))
18 print("-"*width)
19 print(body.format("Apple", 4.0))
20 print(body.format("Watermelon", 2.0))
21 print(body.format("Peach", 5.0))
22 print("="*width)
```

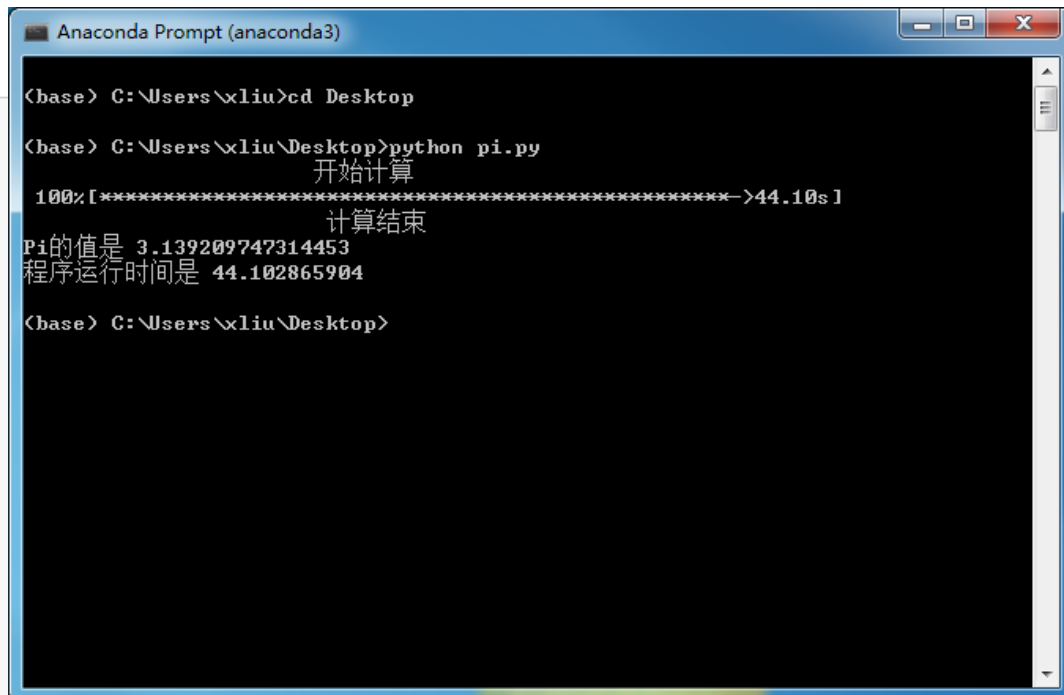
请输入价格表总字符宽度: 50

Item	Price
Apple	4.00
Watermelon	2.00
Peach	5.00

课后作业-打印格式化动态进度条

课后作业2 打印格式化动态进度条

```
1 import random, math, time
2 DARTS = int(2**20)
3 hits = 0
4
5 #####补充代码#####
6
7 #####
8
9 start_time = time.perf_counter()
10 for i in range(1, DARTS):
11     x, y = random.random(), random.random()
12     dist = math.sqrt(x**2 + y**2)
13     if dist <= 1.0:
14         hits = hits + 1
15     #进度百分比
16     percentage = round(i/DARTS*100)
17     star = "*" * round(i/DARTS*50)
18     dot = "." * (50 - round(i/DARTS*50))
19     during_time = time.perf_counter() - start_time
20     #####补充代码#####
21     print("\r{}".format(), end = "")
22     #####
23
24 #####补充代码#####
25
26 #####
27
28 pi = 4 * (hits/DARTS)
29 print("Pi的值是", pi)
30 during_time = time.perf_counter() - start_time
31 print("程序运行时间是", during_time)
```



```
Anaconda Prompt (anaconda3)

(base) C:\Users\xliu>cd Desktop
(base) C:\Users\xliu\Desktop>python pi.py
开始计算
100%[*****->44.10s]
计算结束
Pi的值是 3.139209747314453
程序运行时间是 44.102865904
(base) C:\Users\xliu\Desktop>
```

Python 数据类型-上答题卡

班级:

姓名:

学号:

课后作业1 格式化打印价格列表, 请补全如下代码

```
1 #定义各输出字段的宽度
2 width = input("请输入价格表总字符宽度: ")
3 price_width = 10
4 #####补充代码#####
5
6 #####补充代码#####
7 item_width = width - price_width
8
9 #####补充代码#####
10 #定义格式化方式
11 head =
12 body =
13 #####补充代码#####
14
15 #打印价格列表
16 print("-"*width)
17 print(head.format("Item", "Price"))
18 print("-"*width)
19 print(body.format("Apple", 4.0))
20 print(body.format("Watermelon", 2.0))
21 print(body.format("Peach", 5.0))
22 print("-"*width)
```

课后作业2 打印格式化动态进度条

```
1 import random, math, time
2 DARTS = int(2**20)
3 hits = 0
4
5 #####补充代码#####
6
7 #####补充代码#####
8
9 start_time = time.perf_counter()
10 for i in range(1, DARTS):
11     x, y = random.random(), random.random()
12     dist = math.sqrt(x**2 + y**2)
13     if dist <= 1.0:
14         hits = hits + 1
15     #进度百分比
16     percentage = round(i/DARTS*100)
17     star = "*" * round(i/DARTS*50)
18     dot = "." * (50-round(i/DARTS*50))
19     during_time = time.perf_counter() - start_time
20     #####补充代码#####
21     print("\r{}".format(star + dot), end = "")
22     #####补充代码#####
23
24 #####补充代码#####
25
26 #####补充代码#####
27
28 pi = 4 * (hits/DARTS)
29 print("Pi的值是", pi)
30 during_time = time.perf_counter() - start_time
31 print("程序运行时间是", during_time)
```

各班派一位学生代表收齐,
下次课前交上来

小结

□ 数字：四则运算、内置函数和模块

□ 字符串：操作、方法、模块和格式化打印

注意：字符串的操作是序列类型数据的通用操作（列表和元组），数字和字符串属于**不可变数据类型**

下一节：Python容器数据类型（列表、元组、字典）