python格式化输出

占位符

`%[key][flags][width][.precision][length type]conversion type`

1. conversion type:主要有三个—— s(字符串)、d(十进制整数)、f(十进制浮点数,默认保留6位)
2. precision： `.precision` 设置精度
3. key:映射的键，由带括号的字符序列组成,搭配字典的values使用

In [1]:
```
# conversion type
print("%s  %s  %s" % ("hello", 3, 3.1415))
print("%s  %d  %d" % ("hello", 3, 3.1415))
print("%s  %d  %f" % ("hello", 3, 3.1415))
print("%s  %f  %f" % ("hello", 3, 3.1415))
```

```
hello  3  3.1415
hello  3  3
hello  3  3.141500
hello  3.000000  3.141500
```

In [2]:
```
# precision
print('%f'%3.14)
print('%.1f'%3.14)
print('%.8f'%3.14)
```

```
3.140000
3.1
3.14000000
```

In [3]:
```
# key
print('%(name)s'%{'name':'lily'})
```

```
lily
```

format

format是字符串格式化方法之一，允许多个替换、值格式化 `"... {[field_name][!conversion][:format_spec]} ...".format(arguments)`

1. `arguments` :位置参数、关键字参数
2. `field_name` : 选填。字段名，常使用其基础格式arg_name来指定使用arguments哪一个。对于关键词参数，arg_name必须为其中的关键字，（此时该字段是必填项）
3. `conversion` :选填。变换，不常用。指定时要用!来开头，指定后会在格式化之前将arguments中对应的值进行类型变换。其有三个值可以指定，分别为s(`str` 方法)、r(`repr` 方法)、a(`ascii` 方法)
4. `format_spec` :选填，格式化具体规范，核心内容，超常用。填写时要用:来开头，填写后，会按照其指定的规则来进行格式化。

其详细语法为

`[[fill]align][sign][#][0][width][grouping_option][.precision][type]`

- `fill`：填充内容，如果指定了宽度，但变量长度不够，会使用该字段值进行填充。设置了fill，后面必须显式设置align。
- `align`：对齐方式，有以下值：`<`(强制左对齐)、`>`(强制右对齐)、`=`(强制将填充内容放在符号(如果有)之后但数字之前)、`^`(强制居中对齐)
- `sign`:符号展现格式，仅对数字类型有效。有以下值：`+`、`-`、` `
- `0`：当没有设置对齐方式align时，在宽度字段前面加一个零('0')字符，将等价于填充字符fill为0且对齐方式align为`<`。
- `width`：最小字段宽度，不设置则字段宽度将始终与填充它的数据长度相同（此时对齐方式align没有意义）。
- `grouping_option`：分组选择，有两个选项可选:`,`(表示使用逗号作为千位分隔符)、`-`
- `precision`：精度,指定时要用.来开头,是一个十进制数，指定用'f'和'f'格式化的浮点值在小数点后应该显示多少位,即保留几位小数
- `type`：类型，决定数据应该如何显示。s(字符串)、d(十进制整数)、f(十进制浮点数,默认保留6位)

In [4]:
```
# 位置参数
print("{} {}".format("Li hua", 24))
print("{0} {1} {1} {0}".format("Li hua", 24))
```

```
Li hua 24
Li hua 24 24 Li hua
```

In [5]:
```
# 关键字参数
print("{name} {age}".format(name="Li hua", age=24))
print("{name} {age} {age} {name}".format(name="Li hua", age=24))
```

```
Li hua 24
Li hua 24 24 Li hua
```

In [6]:
```
print("{0[name]} {0[age]}, {1[0]} {1[1]}".format({"name": "Li hua", "age": 24},
```

```
Li hua 24, Zhang san 24
```

In [7]:
```
# [:format_spec]
print("{:4}{:6},{:10}".format("1", "2", 3.14))  # set width

print("{:4}{:>6}, {:^10}".format("1", "2", 3.14))  # set width, align

print("{:_<4}{:0>6}, {:^10}".format("1", "2", 3.14))  # set width, align, fill

print("{:_<4}{:0>6}, {:^10.4f}".format("1", "2", 3.14))  # set width, align, fil
```

```
1   2     ,      3.14
1        2,    3.14
1___000002,    3.14
1___000002,   3.1400
```

f-表达式

f-表达式是前缀为"f"或"F"，用花括号{}包裹替换字段的字符串文字。

其简易格式为： f'{name} is {age} years old'。 f表达式中的字符串内容，是由任意个 `literal_char` 、 `{{、}}` 、 `replacement_field` 自由组成的

```
f'(literal_char | {{ | }} | replacement_field)*'
```

```
F'(literal_char | {{ | }} | replacement_field)*'
```

- `literal_char` 是除花括号{}外的任意字符或空
- f表达式中要表示花括号{}文本，需要进行转义，转义方式为 `{{, }}`
- `replacement_field` 是替换字段，是f表达式的核心,其格式是 `{f_expression[=][!conversion][:format_spec]}`

In [8]:
```
line = "The output will have the expression text"
print(f"{line = }" ) # use "=" sign, require python 3.8 or above
width = 10
precision = 4
value = 12.34567
print(f"result: {value:{10}.{4}}")  # set format_spec
print(f"result: {value:{width}.{precision}}")  # nested fields
```

```
line = 'The output will have the expression text'
result:      12.35
result:      12.35
```