Python中总共有5种方法用于格式化字符串输出。

假设我们有变量errno=50159747054. name=”Bob’, 假设需要输出信息为’Hey Bob, there is a 0xbadc0ffee error!’. 其中50159747054的16进制表示为badc0ffee

**方法1: 手动拼接字符串**

手动拼接字符串比较繁琐，特别在处理需要空格，转变数据输出格式，都需要手动处理。而且在源代码上由于使用+进行多次的拼接，所以显示的不直接。

例如 print(“Hey “ + name + “, there is a “ + hex(errno) + “ error!”)

**方法2： 使用“Old Style”的%操作符**

类似于C语言中的printf, 使用%运算符进行格式化输出。在源字符串中使用%类型表示插入的类型，然后以%分割参数字符串，%后可以接入tuple形式参数字符串按顺序表达，或者以字典形式进行key-value形式表达。

例如：

'Hey %s, there is a 0x%x error!' % (name, errno) # 源字符串中%s表示此位置插入字符串，%x表示插入16进制数字表达， %分隔符号， （name, errno）以tuple形式按顺序插入。

'Hey %(name)s, there is a 0x%(errno)x error!' % {"name": name, "errno": errno }# 源字符串中%（name）s表示此位置插入key名字为name的类型为字符串，%(errno)x表示此位置插入key名字为errno的16进制数字表达， %分隔符号， {"name": name, "errno": errno }以字典形式插入，可以不按顺序。

**方法3：使用str.format方法（Python3.6前，Python2推荐）**

从python2.7以及python3开始，引入了str.format方法进行格式化字符串。在这种方式中不需要使用%格式符。我们可以按位置以tuple形式对源字符串进行插入，或者不按顺序，以按key-value字典形式对源字符串进行插入。

例如：

'Hey {0}, there is a 0x{1:x} error!'.format(name, errno) # 不使用%符号。使用.format，然后按位置，参数以tuple形式进行插入。并且在源字符串中以{1:x}， 表示插入为后面参数tuple中index=1, 格式为x 16进制形式。

'Hey {name}, there is a 0x{errno:x} error!'.format(name=name, errno=errno) # 不使用%符号。使用.format，然后按key-value字典形式进行插入。并且在源字符串中以{ errno:x }， 表示插入为后面参数key=errno部分，为x 16进制形式。

**方法4： 使用f-strings方法(Python3.6后推荐)**

在Python3.6以后Python推出新的格式串字符串方法f-strings.在f-strings格式化字符串中，采取的风格是直接将变量插入源字符串当中，而不需要采用%或者.format将源字符串和参数列表进行分隔。

例子1：

name = “Bob”

>>> f'Hello, {name}!' # 在字符串中以f表示f-strings, 并且以{变量名}插入变量

'Hello, Bob!'

例子2：

>>> def greet(name, question):

... return f"Hello, {name}! How's it {question}?" # 在字符串中以f表示f-strings, 并且以{变量名}插入变量

...

>>> greet('Bob', 'going')

"Hello, Bob! How's it going?"

例子3：

>>> f"Hey {name}, there's a {errno:#x} error!" # 在字符串中以f表示f-strings, 并且以{变量名}插入变量

，并且以:#x插入16进制形式

"Hey Bob, there's a 0xbadc0ffee error!"

例子4：

>>> a = 5

>>> b = 10

>>> f'Five plus ten is {a + b} and not {2 \* (a + b)}.' # 在字符串中以f表示f-strings, 并且以{变量名}插入变量，并且以{}内允许采用expression，输出'Five plus ten is 15 and not 30.'

**方法5： Template Strings（Standard Library）**

使用template strings进行格式化字符串时候，需要引入string模块中的Template方法。流程是，将源字符串传入模板中，然后传入变量对该模板进行渲染，得出结果。

例子1：

>>> from string import Template

>>> t = Template('Hey, $name!') # 读取模板字符串

>>> t.substitute(name=name) # 使用模板传入变量，进行渲染

'Hey, Bob!'

例子2：

>>> templ\_string = 'Hey $name, there is a $error error!'

>>> Template(templ\_string).substitute(name=name, error=hex(errno))

# 返回结果'Hey Bob, there is a 0xbadc0ffee error!'