* **ex5**

使用round()函数实现四舍五入

例：

print(round(1.7333)) # 输出结果为2

* **ex6**

字符串内部使用format()函数可内嵌其他变量，嵌入一组{}处ss

例：

hilarious = False

joke\_evaluation = "Isn't that joke so funny?! {}"

print(joke\_evaluation.format(hilarious))

# 输出结果为Isn't that joke so funny?! False

* **ex7**

print()函数输出字符串结尾，使用end参数

例：

print("hello", end='\*\*\*')

print("world", end='!!!')

# 输出结果为hello&&&world!!!

* **ex9**

三引号（"""xxx"""）用于块注释，可使用print()函数输出块注释

例：

print("""

There's something going on here.

With the three double-quotes.

We'll be aboe to type as much as we like.

Even 4 lines if we want, or 5, or 6.

""")

# 输出结果为上述一段文字，分四行显示

* **ex10**

1、三引号既可以使用双引号，也可以使用单引号。若块注释内需要出现双引号，则可以使用一对3个单引号（'''X"X"X'''）将要注释的内容包裹在其中。

例：

print('''

I'll do a list:

\t"\* Cat food"

\t""\* Fishies""

\t"""\* Catnip\n\t\* Grass"""

''')

输出结果如下：

# I'll do a list:

# "\* Cat food"

# ""\* Fishies""

# """\* Catnip

# \* Grass"""

2、Python转义序列笔记

|  |  |
| --- | --- |
| 转义字符 | 功能 |
| \\ | 反斜杠（\） |
| \' | 单引号（'） |
| \" | 双引号（"） |
| \a | ASCII响铃符（BEL） |
| \b | ASCII退格符（BS） |
| \f | ASCII进纸符（FF） |
| \n | ASCII换行符（LF） |
| \N{name} | Unicode数据库中的字符名，其中name是它的名字，仅Unicode适用 |
| \r | ASCII回车符（CR） |
| \t | ASCII水平制表符（TAB） |
| \uxxxx | 值为16位十六进制值xxxx的字符 |
| \Uxxxxxxxx | 值为32位十六进制值xxxxxxxx的字符 |
| \v | ASCII垂直制表符（VT） |
| \ooo | 值为八进制值ooo的字符 |
| \xhh | 值为十六进制值hh的字符 |

3、转义序列和格式化字符串的组合

例：

# 将转义序列和格式化字符串组合在一起

hw = "\aCR\_test\r:\x68\145llo,\nworld"

print(f"test1{hw}")

print("test2{}".format("\aworld\bb2\\ hello"))

输出结果如下：

:hello,\_test

world

test2worlb2\ helloex11

* **ex12**

1、使用input()函数进行标准输入（键盘输入），默认返回类型为字符串

例：

str = input("Input some text here: ")

str2 = input("more things here: ")

print("The text you input is", str + str2)

输出结果如下：

Input some text here: 5

more things here: 6

The text you input is 56

# 输出的结果为56，而非“5 + 6”或“11”

2、可以使用强制类型转换，将输入的字符串转换为其他类型

例：

num1 = int(input("Input a number here: "))

num2 = int(input("Input another number here: "))

print("num1 + num2 =", num1 + num2)

输出结果如下：

Input a number here: 55

Input another number here: 66

num1 + num2 = 121

3、可以使用pydoc查看函数的使用方法，pydoc使用方法如下：

例如Windows端使用pydoc查询input()函数使用方法，终端输入如下：

>python -m pydoc input

* **ex13**

使用import接收参数，若从系统接收参数，则编译前便要添加命令行参数，且第一个参数为python文件本身

例：

from sys import argv

script, first, second, third = argv

print("The script is called:", script)

print("Your first variable is:", first)

print("Your second variable is:", second)

print("Your third variable is:", third)

正确编译（带有三个参数）：

>python .\ex13.py first 2nd 3rd

输出结果为：

The script is called: .\ex13.py

Your first variable is: first

Your second variable is: 2nd

Your third variable is: 3rd

错误编译1（不带参数，直接编译）

>python .\ex13.py

编译结果：

Traceback (most recent call last):

File ".\ex13.py", line 2, in <module>

script, first, second, third = argv

ValueError: not enough values to unpack (expected 4, got 1)

错误编译2（带有两个参数，少一个）：

>python .\ex13.py first 2nd

编译结果：

Traceback (most recent call last):

File ".\ex13.py", line 2, in <module>

script, first, second, third = argv

ValueError: not enough values to unpack (expected 4, got 3)

错误编译3（带有四个参数，多一个）：

>python .\ex13.py first 2nd 3rd 4th

编译结果：

Traceback (most recent call last):

File ".\ex13.py", line 2, in <module>

script, first, second, third = argvValueError: too many values to unpack (expected 4)

* **ex14**

注释块中也可以引入变量，通过格式化字符串输出

例：

likes = "yes"

lives = "GZ"

computer = "Surface Pro"

# 以下注释块引入上面所定义的三个变量

print(f"""

Alright, so you said {likes} about liking me.

You live in {lives}. Not sure where that is.

And you have a {computer} computer. Nice.

""")

输出结果如下：

Alright, so you said yes about liking me.

You live in GZ. Not sure where that is.

And you have a Surface Pro computer. Nice.

* **ex16**

写入文件使用open()函数，需要加参数"w"

下表为open()函数的参数（[来源](https://www.runoob.com/python/python-func-open.html) ）

|  |  |
| --- | --- |
| **模式** | **描述** |
| r | 以只读方式打开文件。文件的指针将会放在文件的开头。这是默认模式。 |
| rb | 以二进制格式打开一个文件用于只读。文件指针将会放在文件的开头。这是默认模式。 |
| r+ | 打开一个文件用于读写。文件指针将会放在文件的开头。 |
| rb+ | 以二进制格式打开一个文件用于读写。文件指针将会放在文件的开头。 |
| w | 打开一个文件只用于写入。如果该文件已存在则打开文件，并从开头开始编辑，即原有内容会被删除。如果该文件不存在，创建新文件。 |
| wb | 以二进制格式打开一个文件只用于写入。如果该文件已存在则打开文件，并从开头开始编辑，即原有内容会被删除。如果该文件不存在，创建新文件。 |
| w+ | 打开一个文件用于读写。如果该文件已存在则打开文件，并从开头开始编辑，即原有内容会被删除。如果该文件不存在，创建新文件。 |
| wb+ | 以二进制格式打开一个文件用于读写。如果该文件已存在则打开文件，并从开头开始编辑，即原有内容会被删除。如果该文件不存在，创建新文件。 |
| a | 打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。 |
| ab | 以二进制格式打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。也就是说，新的内容将会被写入到已有内容之后。如果该文件不存在，创建新文件进行写入。 |
| a+ | 打开一个文件用于读写。如果该文件已存在，文件指针将会放在文件的结尾。文件打开时会是追加模式。如果该文件不存在，创建新文件用于读写。 |
| ab+ | 以二进制格式打开一个文件用于追加。如果该文件已存在，文件指针将会放在文件的结尾。如果该文件不存在，创建新文件用于读写。 |