CTF Class 1: **Python入门**

Python for Fresh CTFers

Representer: Sike

5 min

Preparations

Choice 1 在线Python: https://www.jyshare.com/compile/9/

Choice 2 VSCode + Python (Homework)

Python+Vscode 安装教程

lz哆啦A梦 · 9-15

安装与配置教程：Python+Vscode 安装教程
https://www.bilibili.com/video/BV1ohpTz9E8P

More Choices IDLE; Pycharm; Python REPL; Jupyter; …

Representer: Sike

2 min

Topic 0: What is Python?

Python 是一种**高级、通用、解释型**的**编程语言**。它由吉多·范罗苏姆在1991年创造，以其简洁易读的语法而闻名，极大地强调了代码的可读性。

可以把Python理解成一种用来**与计算机沟通的语言**，它让你能够**编写指令（即程序或脚本），让计算机去执行各种任务**。

Representer: Sike

3 min

Topic 0: What is Python?

1. 可交互的（解释型语言）：

```
PS C:\Users\sike> python
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 1 + 2
3
>>> 1 - 2
-1
>>> 1 * 2
2
>>> 1 / 2
0.5
>>> 1 ** 2
1
>>>
```

注：REPL (Read-Eval-Print Loop)：Python REPL **读取**了我们输入的命令表达式，并**进行了计算**，然后将最后的表达式的结果的值以**打印**输出的形式返回给我们。

Topic 0: What is Python?

1. 可交互的（解释型语言）：

```
PS C:\Users\sike> python
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 1 + 2
3
```

```
PS C:\Users\sike> python
Python 3.13.5 (tags/v3.13.5:6cb20a2, Jun 11 2025, 16:15:46) [MSC v.1943 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 1 + 2
3
>>> print(1 + 2)
3
>>>
```

几乎是一个意思（调研或思考：原理上真的是一样吗？）

注：REPL (Read-Eval-Print Loop)：Python REPL **读取**了我们输入的命令表达式，并**进行了计算**，然后将最后的表达式的结果的值以**打印**输出的形式返回给我们。

Topic 0: What is Python?

2. 也可以编写静态的代码，然后交给计算机运行，得到结果：

```
#!/usr/bin/python
# Write Python 3 code in this online editor and run it.
print(1 + 2)
print(1 - 2)
print(1 * 2)
print(1 / 2)
print(1 ** 2)
```

```
3
-1
2
0.5
1
```

Python在线平台：https://www.jyshare.com/compile/9/

Representer: Sike

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

Representer: Sike

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

```
print(1 + 2)
```

注：REPL (Read-Eval-Print Loop)： Python REPL 读取了我们输入的命令表达式，并进行了计算，然后将最后的表达式的结果的值以打印输出的形式返回给我们。

Representer: Sike

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

① 读取输入

```
print(1 + 2)
```

注：REPL (Read-Eval-Print Loop)：Python REPL 读取了我们输入的命令表达式，并进行了计算，然后将最后的表达式的结果的值以打印输出的形式返回给我们。

Representer: Sike

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

① 读取输入 — ② 进行计算

```
print( 3 )
```

注：REPL (Read-Eval-Print Loop)： Python REPL 读取了我们输入的命令表达式，并进行了计算，然后将最后的表达式的结果的值以打印输出的形式返回给我们。

Representer: Sike

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

① 读取输入 — ② 进行计算

```
print(  3  )
```

③ 将计算得到的值进行打印

注：REPL (Read-Eval-Print Loop)： Python REPL 读取了我们输入的命令表达式，并进行了计算，然后将最后的表达式的结果的值以打印输出的形式返回给我们。

Representer: Sike

Topic 1: Calculator

知识点一：函数调用

函数调用的结构：

函数名(函数参数)

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

函数名　　函数参数

```
print(  3  )
```

函数调用运算符

① 读取输入 — ② 进行计算 — ③ **将计算得到的值进行打印**

Topic 1: Calculator

知识点一：函数调用

函数调用的结构：

函数名(函数参数)

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

函数名　　函数参数

```
print(  3  )
```

函数调用运算符

Python中函数的功能：接受**函数参数**（也可以是0个参数），根据参数进行某一系列**操作**，并**返回一个值**。

① 读取输入 — ② 进行计算 — ③ **将计算得到的值进行打印**

Topic 1: Calculator

知识点一：函数调用

函数调用的结构：

函数名(函数参数) -> 返回值

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

```
print(  3  )
```

与计算机底层进行沟通，并把函数参数的值打印到屏幕的恰当位置（输出流）。

```
None
```

函数返回值

Python中函数的功能：接受**函数参数**（也可以是0个参数），根据参数进行某一系列**操作**，并**返回一个值**。

① 读取输入 — ② 进行计算 — ③ **将计算得到的值进行打印**

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

体验：打印下述几个函数的返回值：

```
print()
type(333)
type("18")
type([1, 2, 3])
type((4, 5, 6))
type(set([7, 8, 9]))
type({1: "a", 2: "b"})
ord("A")
chr(97)
```

① 读取输入 — ② 进行计算 — ③ **将计算得到的值进行打印**

Topic 1: Calculator

知识点二：数据类型

数据类型的作用：

- 让计算机知道如何操作数据
- 不同类型的数据有不同的功能

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

int	整型	12
float	浮点型	0.5
bool	布尔型	True, False
bytes	字节型	b”123”, b’456’
str	字符串	“12”, ‘12’
list	列表	[1, 2, 3]
tuple	元组	(1, 2, 3)
set	集合	set([1, 2, 3])
dict	字典	{1: “a”, 2: ”b”}

练习：了解不同数据类型之间的转换方法。

① 读取输入 — ② 进行计算 — **③ 将计算得到的值进行打印**

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

```
print(1 + 2)
```

① 读取输入 — **② 进行计算** — ③ 将计算得到的值进行打印

Topic 1: Calculator

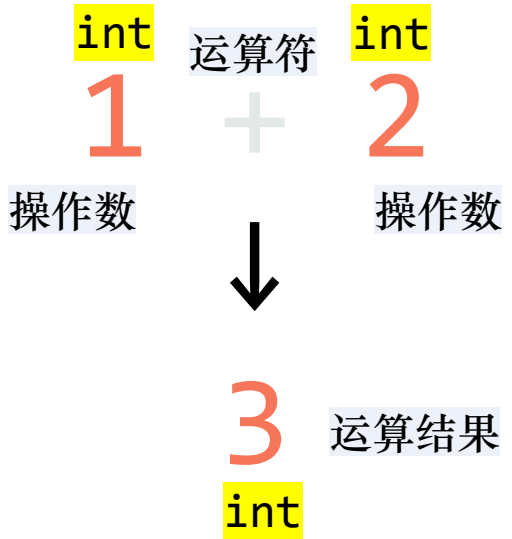
知识点三：运算

运算把数据通过特定的规则变成新的数据：

操作数 运算符 操作数

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器



① 读取输入 — **② 进行计算** — ③ 将计算得到的值进行打印

Representer: Sike

Topic 1: Calculator

知识点三：运算

运算把数据通过特定的规则变成新的数据：

操作数 运算符 操作数

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

算术运算	加、减、乘、除	1 + 2, 1 - 2, 1 * 2, 1 / 2
	模（取余数）	7 % 2
	乘方	2 ** 4
比较运算	大于、小于	1 < 2, 8 > 1, 5 <= 16, 51 >= 6
	等于、不等于	5 == 5, 1 != 6
逻辑运算	与	True and False
	或	True or True
	非	not True
位运算	与、或、非、异或	3 & 1, 2 | 1, ~2, 3 ^ 1
其他	…	…

① 读取输入 — **② 进行计算** — ③ 将计算得到的值进行打印

Topic 1: Calculator

知识点三：运算

运算把数据通过特定的规则变成新的数据：

操作数 运算符 操作数

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

```
1    print(3 * "2")
2    print(6 * [0])
3    print([1] + [3, 4])
4    print("23" + "45")
5    print(int("0xd", 16))
```

练习：了解不同数据类型之间的常见运算。

① 读取输入 — **② 进行计算** — ③ 将计算得到的值进行打印

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 c
3 print(1 + 2)
4 print(1 - 2)
5 print(1 * 2)
6 print(1 / 2)
7 print(1 ** 2)
```

任务一：尝试搭建一个简易的计算器

```
print(1 + 2)
```

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Topic 1: Calculator

知识点四：变量与赋值

- 变量：存储数据的量
- 赋值：赋予变量数据

变量名 = 值

```
1 #!/usr/bin/python
2 # Write Python 3 cod
3 a = 1
4 b = 2
5 print(a + b)
6 print(a - b)
7 print(a * b)
8 print(a / b)
9 print(a ** b)
```

任务一：尝试搭建一个简易的计算器

变量名 a = 1 值

赋值运算符

赋值语句

```
b = 2
print(a + b)
```

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Representer: Sike

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 co
3 a, b = 1, 2
4 op = "+"
5 print(a op b)?
6
7
8
9
```

任务一：尝试搭建一个简易的计算器

```
print(a + b)
print(a - b)
print(a * b)
print(a / b)
print(a ** b)
```

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Topic 1: Calculator

```
1 #!/usr/bin/python
2 # Write Python 3 co
3 a, b = 1, 2
4 op = "+"
5 print(a op b)
6
7
8
9
```

任务一：尝试搭建一个简易的计算器

```
print(a + b)
print(a - b)
print(a * b)
print(a / b)
print(a ** b)
```

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Topic 1: Calculator

知识点五：条件语句

```
if 条件：
    条件体
```

```
3 a = 1
4 op = "+"
5 b = 2
6 if op == "+":
7     print(a + b)
8 if op == "-":
9     print(a - b)
10 if op == "*":
11     print(a * b)
12 if op == "/":
13     print(a / b)
14 if op == "**":
15     print(a ** b)
```

请补全

任务一：尝试搭建一个简易的计算器

```
if op == "+":
    print(a + b)
```

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

函数	输入类型	功能	输出类型	例子
input	str	从用户获取输入，显示提示信息并等待用户输入	str	name = input("请输入姓名: ")
int	bool, str, float	将其他类型的数据转换为整数	int	int("123") -> 123

现在请大家完成一个计算器，能够接收用户的输入，然后打印计算结果。（样例如下）

```
Please input the first number > 123
Please input the second number > 5
Please input the operator(`+`, `-`, `*`, `/`, `**`) > **
28153056843
```

练习：学习运算符in，str类型的split()函数以及list类型的相关操作，实现一行算式（输入形如`123 + 5`）的计算。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Representer: Sike

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

```
a, b = int(input("number1 > ")), int(input("number2 > "))
op = input("Operator(`+`, `-`, `*`, `/`, `**`) > ")
if op == "+": print(a + b)
elif op == "-": print(a - b)
elif op == "*": print(a * b)
elif op == "/": print(a / b)
elif op == "**": print(a ** b)
else: print("Unknown operator!")
```

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

知识点六：循环语句

while循环：

```
while 条件：
    循环体
```

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

```python
while True:
    a, b = int(input("number1 > ")), int(input("number2 > "))
    op = input("Operator(`+`, `-`, `*`, `/`, `**`) > ")
    if op == "+": print(a + b)
    elif op == "-": print(a - b)
    elif op == "*": print(a * b)
    elif op == "/": print(a / b)
    elif op == "**": print(a ** b)
    else: print("Unknown operator!")
```

注：如果循环退不出来按 ctrl+C

练习：学习运算符in以及list类型，学习什么是for循环。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

知识点七：函数

```
def 函数名(函数参数):
    函数体
    return 返回值
```

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

a$b = a * b + a - b　一个自定义运算

```
def mycalc(a, b):
    c = a * b + a - b
    return c
```

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Representer: Sike

知识点七：函数

```
def 函数名(函数参数):
    函数体
    return 返回值
```

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

a$b = a * b + a - b　一个自定义运算

```
def mycalc(a, b):
    return a * b + a - b
```

注：没有return则返回值默认为None；函数一定要有函数体（对pass关键字进行学习）

练习：将这个功能添加到先前写好的计算器中。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Representer: Sike

知识点七：函数

```
def 函数名(函数参数):
    函数体
    return 返回值
```

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

a$b = a * b + a - b　一个自定义运算

```
def mycalc(a, b):
    return a * b + a - b
```

```
mycalc = lambda a, b: a * b + a - b
```

注：没有return则返回值默认为None；函数一定要有函数体（对pass关键字进行学习）

练习：将这个功能添加到先前写好的计算器中。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

我想要实现开方运算…

但是我好像我不知道怎么实现开方运算…/实现很复杂我不太想亲自实现…

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

知识点八：库与库函数的调用

```
import 库名
from 库名 import 函数名
import 库名 as 自定义库名
from 库名 import 函数名 as 自定义函数名
```

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

我想要实现开方运算…

但是我好像我不知道怎么实现开方运算…/实现很复杂我不太想亲自实现…

我们可以使用他人已经写好的函数！（或其他东西）

库 ——→ 库函数

```
from math import sqrt
```

练习：将开平方功能添加到先前写好的计算器中。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

知识点八：库与库函数的调用

```
import 库名
from 库名 import 函数名
import 库名 as 自定义库名
from 库名 import 函数名 as 自定义函数名
```

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

库 ——→ 库函数

```
from math import sqrt

a = 3
print(sqrt(a))
```

练习：将开平方功能添加到先前写好的计算器中。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

知识点八：库与库函数的调用

`from` 库名 `import` 函数名

函数名直接调用

`import` 库名

函数调用：

库名.函数名(函数参数)

库 ⟶ 库函数

```
from math import sqrt

a = 3
print(sqrt(a))
```

```
import math

a = 3
print(math.sqrt(a))
```

点号操作符

练习：将开平方功能添加到先前写好的计算器中。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Topic 1: Calculator

任务一：尝试搭建一个简易的计算器

知识点八：库与库函数的调用

```
from 库名 import 函数名 as 自定义函数名
```

自定义函数名直接调用

```
import 库名 as 自定义库名
```

函数调用：

```
自定义库名.函数名(函数参数)
```

库 ——→ 库函数

```
from math import sqrt as sr

a = 3
print(sr(a))
```

```
import math as mt

a = 3
print(mt.sqrt(a))
```

练习：将开平方功能添加到先前写好的计算器中。

① 读取输入 — ② 进行计算 — ③ 将计算得到的值进行打印

Review

1.输入/输出	**Input/Output (I/O)**
2.赋值与运算	**Assignment & Arithmetic**
3.数据类型	**Data Types**
4.条件语句	**Conditional Statements**
5.循环语句	**Loops**
6.函数调用	**Function Calls**
7.函数	**Functions**
8.库与库函数调用	**Libraries & Library Functions**

Representer: Sike

Topic 2: Exercise

（请借助网络资源完成下述练习）

- 请学习 Python 中 **print 的 end 选项和 sep 选项；**
- 请学习 Python 中的 **条件语句（if、elif、else）**、**循环语句（for，while）**。

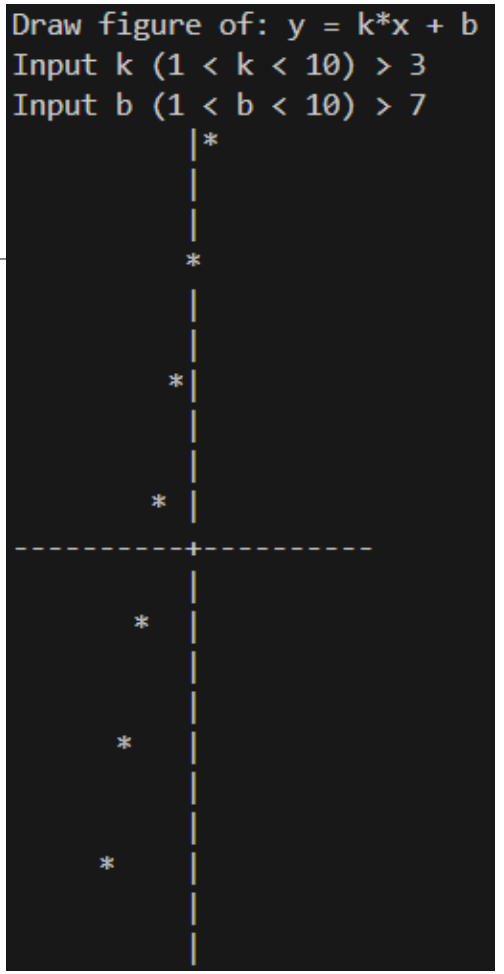
Ex1.1：输入自然数n，输出1~n之中所有完全平方数；

Ex1.2：输入自然数n，k，每行十个地输出1~n之中的所有完全k方数；

Ex1.3：输入自然数n，k，以下三角的形式输出1~n之中的所有完全k方数。

Ex2：输入正整数n，判断它是否是素数。（hint:某个正整数n是素数当且仅当比n/2小的能整除它的正整数数只有1）

Ex3：绘制一次函数（y=kx+b，-10<=x<=10，-10<=y<=10，-10<k<10，-10<b<10）的字符画图像。示例如图所示：



Topic 2: Exercise

（请借助网络资源完成下述练习）

- 请学习 Python 中的**列表类型 list**;
- 请学习 Python 中的**类型标注（Type Hint）**。

Ex4.1: 写一个函数 VecAdd(a: list, b: list) -> list，输入两个向量，实现这两个向量的加法;

Ex4.2: 写一个函数 VecDot(a: list, b: list) -> list，输入两个向量，实现这两个向量的点乘运算;

Ex4.3: 写一个函数 MatAdd(a: list, b: list) -> list，输入两个矩阵，实现这两个矩阵的加法;

Ex4.4: 写一个函数 MatPrd(A: list, B: list) -> list，输入两个矩阵，实现这两个矩阵的乘法运算;

Ex4.5: 写一个函数 VecApp(a: list, x: int | float) -> list，输入一个向量和一个数（int或float），在这个向量的最后添加x。

Ex4.6: （附加题）写一个函数 MatDet(A: list) -> int | float，输入一个矩阵，计算它的行列式。

注：使用双重列表模拟矩阵，注意对输入的向量/矩阵的大小（长度）进行判断。

Topic 2: Exercise

（请借助网络资源完成下述练习）

- 请继续学习 str类型 的 切片操作。

Ex5.1: 将字符串 `hello world` 反转。

Ex5.2. 检查一个字符串是不是回文串（回文串指的是从后向前看和从前向后看都一样的字符串，比如 `abccba`）。

Ex5.3. 截取一个字符串前两个字符。

Ex5.4. 截取一个字符串后两个字符。

Ex5.5. 将一个字符串的偶数索引字符提取出来，组成一个新字符串。

Topic 2: Exercise

（请借助网络资源完成下述练习）

- 请学习 **int函数，hex函数** 和 **bin函数，ord函数** 和 **chr函数；**
- 请学习字符串的 **join 方法与 zfill 方法；**
- （进阶）请学习 **map() 函数**。

Ex6：凯撒密码

凯撒密码是一种替换加密的技术， 明文中的所有字母都在字母表上向后（或向前）按照一个固定数目进行偏移后被替换成密文。如当偏移为向后取2时，明文”abcd”被加密为”cdef”。

现在请你写一个程序实现凯撒密码的加密和解密，其中固定偏移为向后取3，输入输出均为字符串。请你将这里给到程序补充完整。

```
def encrypt(s: str) -> str :
    pass

def decrypt(s: str) -> str :
    pass

if __name__ == '__main__':
    print("凯撒密码加解密")
    print("(1) 加密")
    print("(2) 解密")
    pass
```

Topic 2: Exercise

（请借助网络资源完成下述练习）

- 请学习 Python 中的**唯一标识符函数 id()**。
- 请学习 Python 中 **bytes 类型**的相关操作。

Ex7：（附加）大小端序

数据在计算机里是如何被解释的？在有的CPU架构里，采用大端序，另一部分采用小端序。

大端序指的是将数据的最高有效字节（Most Significant Byte，MSB）存储在最低的内存地址中，而最低有效字节（Least Significant Byte，LSB）存储在最高的内存地址中。**小端序**则反之。

比如一个32bit整数 0x12345678 是小端序，那么它的大端序应该是 0x78563412 。

现在请你用 id() 函数查看一个变量的地址，将它用16进制打印出来，并把它转换大小端序（你的机器不出意外的话应该是64bit，地址是一个64位的整数），再打印出来。

Topic 2: Exercise

Ex8：（附加）类型转换

输入上接收一串二进制数据（不以 0b 开头），将它转换为十六进制数据，和可见字符串，并以一定格式输出。一些具体的要求如下：

- **末尾自动补0；**
- **输出上分两栏，左栏是十六进制数据，右栏是可见字符串（ascii 32~126），左右栏之间相隔 6 个空格；**
- **输出的左栏的十六进制数据以 1 个字节为单位用空格隔开，每行最多 10 个字节；**
- **输出的右栏的字符串与左栏的十六进制数据一一对应，如果字符无法显示则用 . 代替（包括换行符、制表符等）；**
- **最后将数据转换为 bytes 类型进行输出。**

示例见下一页

Topic 2: Exercise

Ex8：（附加）类型转换（续）

输入：

```
0100011000111010110100111011001101000001100000111110010010110000010100101111101110101000101000011110000100111010100100101010101010001111010100111100011000100100011001000101010101111101011100101011000111000110110011100101101000010001101000111101111000101110001101111010101001100010010011010001101011001100101010011111011000000110110111101110010111101111111001110101000110011101101010100001100110100000000011011000000100110100101000010111111
```

输出：

```
46 3a d3 b3 41 83 e4 b0 52 f7      F:..A...R.
1e a7 8c 48 c8 aa fa e5 63 8d      ...H....c.
6f 54 c4 9a 35 99 53 ec 0d bd      oT..5.S...
66 80 36 04 d2 8b f0               f.6....
b'F:\xd3\xb3A\x83\xe4\xb0R\xf7QC\xc2u%U\x1e\xa7\x8cH\xc8\xaa\xfa\xe5c\x8d\x9c\xb4#G\xbc\\oT\xc4\x9a5\x99S\xec\r\xbd\xcb\xdf\x9dFv\xa8f\x806\x04\xd2\x8b\xf0'
```

不用管字体颜色