作业

作业的目的是为了让大家熟悉程序语言，锻炼将思路转换成程序逻辑。

九九乘法表

help(print)

```
# 1 方阵
for i in range(1, 10):
    line = ''
    for j in range(1, 10):
        line += str(i) + '*' + str(j) + '=' + str(i*j) + ' '
    print(line)
print('-' * 30)

# 2 九九乘法表
for i in range(1, 10):
    for j in range(1, 10):
        if i >= j:
            print(str(j) + '*' + str(i) + '=' + str(i*j), end=' ')
    print()
print('-' * 30)

# 条件合并
for i in range(1, 10):
    for j in range(1, i+1):
        print(str(j) + '*' + str(i) + '=' + str(i*j), end=' ')
    print()
print('-' * 30)

# 3 九九乘法表 对齐
for i in range(1, 10):
    for j in range(1, i+1):
        product = i * j
        product = str(product) + ' ' if j > 1 and product < 10 else str(product)
        print(str(j) + '*' + str(i) + '=' + product, end=' ')
    print()
print('-' * 30)

# 4 九九乘法表 制表符对齐
for i in range(1, 10):
    for j in range(1, i+1):
        print(str(j) + '*' + str(i) + '=' + str(i*j), end='\t')
    print()
print('-' * 30)

# 5 使用字符串format方法
for i in range(1, 10):
    line = ''
```

```
    for j in range(1, i+1):
        line += '{0}*{1}={2} '.format(j, i, i*j)
    print(line)
print('-' * 30)

# 5 对齐
for i in range(1, 10):
    line = ''
    for j in range(1, i+1):
        line += '{0}*{1}={2:<2} '.format(j, i, i*j)
    print(line)
print('-' * 30)

# 5 对齐改进
for i in range(1, 10):
    line = ''
    for j in range(1, i+1):
        product = i * j
        line += '{}*{}={}{}'.format(j, i, product, '  ' if j > 1 and product < 10 else ' ')
    print(line)
print('-' * 30)
```

{2:<2}对应i*j，:<2冒号是分割符号，<表示左对齐，2表示宽度

扩展题：

```
1*1=1  1*2=2  1*3=3  1*4=4  1*5=5  1*6=6  1*7=7  1*8=8  1*9=9
       2*2=4  2*3=6  2*4=8  2*5=10 2*6=12 2*7=14 2*8=16 2*9=18
              3*3=9  3*4=12 3*5=15 3*6=18 3*7=21 3*8=24 3*9=27
                     4*4=16 4*5=20 4*6=24 4*7=28 4*8=32 4*9=36
                            5*5=25 5*6=30 5*7=35 5*8=40 5*9=45
                                   6*6=36 6*7=42 6*8=48 6*9=54
                                          7*7=49 7*8=56 7*9=63
                                                 8*8=64 8*9=72
                                                        9*9=81
```

请打印成上面的形式

```
# 打印九九乘法表方阵的上半部分
for i in range(1, 10):
    line = ''
    print(' '*7*(i-1), end='') # 前置空格
    for j in range(i, 10):
        product = i * j
        line += '{}*{}={}{}'.format(i, j, product, '  ' if product < 10 else ' ')
    print(line)
```

下面采用右对齐方式，且分割均匀

```
for i in range(1, 10):
    line = ''
    for j in range(i, 10):
        line += '{}*{}={:<{}}'.format(i, j, i * j, 2 if j < 4 else 3)
    print('{:>66}'.format(line))
```

打印如下菱形

```
   *
  ***
 *****
*******
 *****
  ***
   *
```

思路：

行号	星个数	前空格数	总空格数
1	1	3	6
2	3	2	4
3	5	1	2
4	7	0	0
5	5	1	2
6	3	2	4
7	1	3	6

看到规律了吗？

```
for i in range(-3,4):
    if i<0:
        prespace = -i
    else:
        prespace = i
    print(' '*prespace + '*'*(7-prespace*2))
```

把if语句改成三元表达式的样子，也可以使用abs()

```
for i in range(-3, 4):
    print(' ' * abs(i) + '*' * (7 - 2 * abs(i)))
```

打印对顶三角形

```
*******
 *****
  ***
   *
  ***
 *****
*******
```

序号	对称序列	星号数	总空格	前置空格	后置空格
1	3	7	0	0	0
2	2	5	2	1	1
3	1	3	4	2	2
4	0	1	6	3	3
5	1	3	4	2	2
6	2	5	2	1	1
7	3	7	0	0	0

可以看出，只跟前导空格、起点终点有关

```
n = 7
e = n // 2

for i in range(-e, n - e):
    prespace = -i if i < 0 else i
    print(' ' * (e - prespace) + '*' * (2 * prespace + 1))
```

思路2

居中打印

```
n = 7
e = n // 2

for i in range(-e, n - e):
    print('{:^{}}'.format('*' * (2 * abs(i) + 1), n))
```

当然菱形也可以居中打印，请自行完成

打印闪电

```

```

```
   *
  **
 ***
*******
   ***
   **
   *
```

行号	个数	前空格	后空格数	总空格数	数据
1	1	3	3	6	-3
2	2	2	3	5	-2
3	3	1	3	4	-1
4	7	0	0	0	0
5	3	3	1	4	1
6	2	3	2	5	2
7	1	3	3	6	3

```python
for i in range(-3, 4):
    if i < 0:
        print(' ' * (-i) + '*' * (4 + i))
    elif i > 0:
        print(' ' * 3 + '*' * (4 - i))
    else:
        print('*' * 7)
```

斐波那契数列，100以内

https://baike.baidu.com/item/%E6%96%90%E6%B3%A2%E9%82%A3%E5%A5%91%E6%95%B0%E5%88%97?fromtitle=%E6%96%90%E6%B3%A2%E6%8B%89%E5%A5%91%E6%95%B0%E5%88%97&fromid=10078434

斐波那契数列：1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...
如果设F(n）为该数列的第n项（n∈N*），那么这句话可以写成如下形式：:F(n)=F(n-1)+F(n-2)
F(0)=0，F(1)=1, F(n)=F(n-1)+F(n-2)
这是一个线性递推数列

```
print(0)
print(1)
a = 0
b = 1
while True :
    c = a + b
    if c > 100 : break
    a = b
    b = c
    print(c)
```

求斐波那契数列第101项

```
a = 1
b = 1
print('index={}, fib={}'.format(0, 0))
print('index={}, fib={}'.format(1, a))
print('index={}, fib={}'.format(2, b))
index = 2

while True:
    c = a + b
    index += 1
    print('index={}, fib={}'.format(index, c))
    if index == 101: break
    a = b
    b = c

# index=101, fib=573147844013817084101
```

求10万内的所有素数

此题的目的是为了让大家注意效率问题

```
for x in range(2,100):
    for i in range(2,x):
        if x % i == 0:
            break
    else:
        print(x)
```

为什么到一个数的“一半”就可以了

```
for x in range(2,100000):
    for i in range(2,int(x ** 0.5)+1):
        if x % i == 0:
            break
    else:
        print(x)
```

下面这段代码是错误代码，用x=4测试，因为内层循环缺少2，那么偶数就出了问题

```
for x in range(2,100000):
    for i in range(3,int(x ** 0.5)+1,2):
        if x % i == 0:
            break
    else:
        print(x)
```

修改为

```
for x in range(3,100000,2): # 舍弃掉所有偶数
    for i in range(3, int(x ** 0.5) + 1, 2): # 为什么从3开始，且step为2？
        if x % i == 0:
            break
    else:
        print(x)
```

> 为什么从3开始，且step为2？ 既然没有偶数，就不用和2取模了。 奇数%偶数能整除吗

利用素数性质：所有大于10的质数中，个位数只有1,3,7,9。

```
count = 1
for x in range(3, 100000, 2):  # 舍弃掉所有偶数
    if x > 10 and x % 10 == 5: # 所有大于10的质数中，个位数只有1,3,7,9。意思就是大于5，结尾是5就能被5整除了
        continue
    for i in range(3, int(x ** 0.5) + 1, 2):
        if x % i == 0:
            break
    else:
        count += 1
        print(x, count) # 9592
```

如何计算时间，import datetime

```
count = 0
for x in range(2,100000):
    for i in range(2,x):
        if x % i == 0:
            break
    else:
        count += 1
print(count)
# 9592
```

```
count = 0
for x in range(2,100000):
    for i in range(2,int(x ** 0.5)+1):
        if x % i == 0:
            break
    else:
        count += 1
print(count)
# 9592
```

应用在密码学领域，都要使用大素数。

```
# 两种算法的对比的完整代码
import datetime

upper_limit = 100000
delta = [0,0]
counts = [0,0]

start = datetime.datetime.now()
for _ in range(10):
    counts[0] = 0
    for x in range(2,upper_limit):
        for i in range(2,int(x ** 0.5)+1):
            if x % i == 0:
                break
        else:
            #print(x)
            counts[0] += 1
delta[0] = (datetime.datetime.now() - start).total_seconds()

start = datetime.datetime.now()
for _ in range(10):
    counts[1] = 1
    #print(2)
    for x in range(3,upper_limit,2):
        for i in range(3,int(x ** 0.5)+1,2):
            if x % i == 0:
                break
```

```
        else:
            #print(x)
            counts[1] += 1
delta[1] = (datetime.datetime.now() - start).total_seconds()

print(delta, sep="\t")
print(counts, sep="\t")
```

解决猴子吃桃问题

猴子第一天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了一个。第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第10天早上想吃时，只剩下一个桃子了。求第一天共摘多少个桃子。

```
total = x
     剩下
1    x/2-1
2    d1/2-1
3    d2/2-1
...
9    d8/2-1
10   1
```

```
peach = 1
for i in range(9):
    peach = 2 * (peach + 1)
print(peach) # 1534
```