



CS001 编程零基础 Python 语言入门

第七讲



作业回顾



第一题： 第二大的数

答案文件名week06solution01a.py

```
def second(x):  
    x.remove(max(x))  
    return max(x)
```

```
print(second([1,2,3,4,5]))  
print(second([10,9,8,7,6]))
```

答案文件名week06solution01b.py

```
def second(x):  
    x.sort()  
    return x[-2]
```

```
print(second([1,2,3,4,5]))  
print(second([10,9,8,7,6]))
```

第二题：递归练习

答案文件名 `week06solution02.py`

```
def sumsquared(a):
```

```
    if a==0:
```

```
        return 0
```

```
    else:
```

```
        return a*a+sumsquared(a-1)
```

```
n=int(input("n = "))
```

```
print(sumsquared(n))
```

递归版本

非递归版本

```
n=int(input("n = "))
```

```
x=[i*i for i in range(1,n+1)]
```

```
print(sum(x))
```


附加题： 质数判定函数

我们希望为 Python 定义一个新功能：判定质数。

第一步：请定义一个函数 `isPrime()`，这个函数的输入参数为一个正整数 `n`，程序能通过这个函数的返回值判断 `n` 是不是质数。

第二步：请利用这个 `isPrime()` 函数，写程序可以自动判断输入的数是否为质数。用标准输入让用户输入一个正整数，判断结果使用 `print(isPrime(n))` 作为输出语句。

输入样例：5

输出样例：True

输入样例：100

输出样例：False

输入样例：1

输出样例：False

附加题： 质数判定函数

答案文件名isPrime.py

注意
缩进

```
def isPrime(n):  
    if n<2:  
        return False  
    for i in range(2,n):  
        if n % i == 0:  
            return False  
    return True
```

如果 $n < 2$ 返回不是质数

如果找到约数 i ， 返回不是质数

始终没有找到2到 $n-1$ 间的质约数， 返回是质数

```
n=int(input("Please input a number: "))  
print(isPrime(n))
```



编程语法综合运用



自定义求最小值功能

Python有一个内置功能函数min()可以返回一个列表中最小的元素，请定义一个函数能够模拟min()实现同样的功能

注意：不能直接使用min()功能

```
>>> min([3, 5, 7, 2, 4, 6, 0, 1, 2])  
0  
>>> min([10, 9, 8, 7, 6, 1, 2, 3, 4, 5])  
1
```


自定义求最小值功能

文件名 mymin1.py

```
def mymin(x):  
    if len(x)==1:  
        return x[0]  
    else:  
        m=mymin(x[1:])  
        if x[0]<m:  
            return x[0]  
        else:  
            return m
```

```
print(mymmin([3,5,7,2,4,6,0,1,2]))  
print(mymmin([10,9,8,7,6,1,2,3,4,5]))
```

用递归的方法实现求最小值

自定义求最小值功能

文件名 mymin2.py

```
def mymin(x):
```

```
    m=x[0]
```

```
    for i in x[1:]:
```

```
        if i<m:
```

```
            m=i
```

```
    return m
```

```
print(mymmin([3,5,7,2,4,6,0,1,2]))
```

```
print(mymmin([10,9,8,7,6,1,2,3,4,5]))
```

用非递归的方法实现求最小值

自定义排序功能

Python有一个内置功能函数sorted()可以返回一个列表排序后的状态，请定义一个函数能够模拟sorted()实现同样的功能

注意：不能直接使用sorted()功能

```
>>> sorted([3, 5, 7, 2, 4, 6, 0, 1, 2])  
[0, 1, 2, 2, 3, 4, 5, 6, 7]  
>>> sorted([10, 9, 8, 7, 6, 1, 2, 3, 4, 5])  
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

自定义排序功能

文件名 mysorted1.py

```
def mysorted(x):  
    if x==[]:  
        return []  
    else:  
        y=x[:]   
        m=min(y)  
        y.remove(m)  
        return [m] + mysorted(y)
```

```
print(mysorted([3,5,7,2,4,6,0,1,2]))  
print(mysorted([10,9,8,7,6,1,2,3,4,5]))
```

用递归的方法实现排序

自定义排序功能

文件名 `mysorted2.py`

```
def mysorted(x):  
    y=x[:]    
    z=[]  
    for i in range(len(y)):  
        m=min(y)  
        y.remove(m)  
        z.append(m)  
    return z
```

```
print(mysorted([3,5,7,2,4,6,0,1,2]))  
print(mysorted([10,9,8,7,6,1,2,3,4,5]))
```

用非递归的方法实现排序

用循环的方法实现排序

自定义颠倒功能

Python有一个内置功能函数`reversed()`可以返回一个列表颠倒后的状态，请定义一个函数能够模拟`reversed()`实现同样的功能

注意：不能直接使用`reversed()`功能

```
>>> list(reversed([3, 5, 7, 2, 4, 6, 0, 1, 2]))  
[2, 1, 0, 6, 4, 2, 7, 5, 3]  
>>> list(reversed([10, 9, 8, 7, 6, 1, 2, 3, 4, 5]))  
[5, 4, 3, 2, 1, 6, 7, 8, 9, 10]
```

自定义颠倒功能

文件名 myreversed.py

```
def myreversed(x):  
    return x[::-1]
```

用列表反向切片的方法实现颠倒功能

```
print(mysorted([3,5,7,2,4,6,0,1,2]))  
print(mysorted([10,9,8,7,6,1,2,3,4,5]))
```

著名的角谷猜想：又叫 $3n+1$ 猜想

1976年《华盛顿邮报》头版头条报道了一条数学新闻。文中记叙了美国各所名牌大学校园内，人们都像发疯一般，夜以继日废寝忘食地玩一种数学游戏。

游戏十分简单：任意写出一个自然数 N ，按照以下规律变换：

如果是个奇数，则下一步变成 $3N+1$ 。

如果是个偶数，则下一步变成 $N/2$ 。

不单单是学生，甚至连教师、研究员、教授与学究都纷纷加入。为什么这种游戏的魅力经久不衰？因为人们发现，无论 N 是怎样一个数字，最终都无法逃脱回到谷底1。

$N=12$ 12 -> 6 -> 3 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1.

$N=19$ 19 -> 58 -> 29 -> 88 -> 44 -> 22 -> 11 -> 34 -> 17 -> 52 -> 26 -> 13 -> 40 -> 20 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1.

著名的角谷猜想：又叫 $3n+1$ 猜想

游戏十分简单：任意写出一个自然数 N ，按照以下规律变换：

如果是个奇数，则下一步变成 $3N+1$ 。

如果是个偶数，则下一步变成 $N/2$ 。

无论 N 是怎样一个数字，最终都无法逃脱回到谷底1。

$N=12$ 12 -> 6 -> 3 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1.

$N=19$ 19 -> 58 -> 29 -> 88 -> 44 -> 22 -> 11 -> 34 -> 17 -> 52 -> 26 -> 13 -> 40 -> 20 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1.

请写一个程序
检验角谷猜想

程序检验：角谷猜想(又叫 $3n+1$ 猜想)

文件名Collatz-conjecture.py

```
for n in range(2,100):
```

```
    x=n
```

```
    while x!=1:
```

```
        print(str(x)+"-",end="")
```

```
        if x % 2 == 0:
```

```
            x=x//2
```

```
        else:
```

```
            x=x*3+1
```

```
    print(1)
```

循环判断每一个数字 n

对于每一个 n 作为变量 x 的初始值，
如果变量 x 不等于1，就一直循环更新变量 x

如果是个偶数，则下一步变成 $x//2$
如果是个奇数，则下一步变成 $3x+1$

N=12 12 -> 6 -> 3 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1.

N=19 19 -> 58 -> 29 -> 88 -> 44 -> 22 -> 11 -> 34 -> 17 -> 52 -> 26 -> 13 -> 40 -> 20 -> 10 -> 5 -> 16 -> 8 -> 4 -> 2 -> 1.

世界三大数学猜想之一：哥德巴赫猜想

哥德巴赫1742年给欧拉的信中，哥德巴赫提出了以下猜想：任一大于2的偶数都可写成两个质数之和。但是哥德巴赫自己无法证明它，于是就写信请教赫赫有名的大数学家欧拉帮忙证明，但是一直到死，欧拉也无法证明。

In 1742, Christian Goldbach, a German amateur mathematician, sent a letter to Leonhard Euler in which he made the following conjecture:

Every even number greater than 4 can be written as the sum of two odd prime numbers.

$$8 = 3 + 5$$

$$20 = 3 + 17$$

$$42 = 5 + 37$$

80=?+?

98=?+?

128=?+?

fabrum, nicht hochstufen, ob nicht aber schon mal hochstufenfab,
 * namque singula series luteri numeros unico modo in duo quadrata
 divisibiles habet, auf solche Weise will ich eine conjecture
 hazardieren: daß jede Zahl welche aus zusammen numeros primis
 zusammengesetzt ist ein aggregatum derialen numerorum
 primorum seig als man will: die unitatem mit jeder quadrat
 habend, die congeriem omnium unitatum*. zinn Beispiel

$$4 = \begin{cases} 1+1+1+1 \\ 1+1+2 \\ 1+3 \end{cases} \quad 5 = \begin{cases} 2+3 \\ 1+1+3 \\ 1+1+1+2 \\ 1+1+1+1+1 \end{cases} \quad 6 = \begin{cases} 1+5 \\ 1+2+3 \\ 1+1+1+3 \\ 1+1+1+1+2 \\ 1+1+1+1+1+1 \end{cases}$$
 LCC
 Daraus folgen nun ganz observationes so demonstrieren werden.
 Von Poisson:
 Si v. sit functio ipsius x. eiusmodi ut facta $v = c$. numero cui-
 cunque, determinari possit x per c. et reliquas constantes in functio-
 ne expressas, poterit etiam determinari valor ipsius x. in æ-
 quatione $v^{n+1} = (2v+1)(v+1)^{n-1}$ | $\frac{v^{n+1} - (2v+1)(v+1)^{n-1}}{v^{n+1} - (2v+1)(v+1)^{n-1}}$ das ist $v-v-1$
 Si incipiat curva cuius abscissa sit x. applicata vero sit
 summa seriei $\frac{x^n}{n \cdot 2^n}$ posita x. pro exponente terminorum, hoc est,
 applicata $= \frac{x}{1 \cdot 2} + \frac{x^2}{2 \cdot 2^2} + \frac{x^3}{3 \cdot 2^3} + \frac{x^4}{4 \cdot 2^4} + \text{etc.}$ dico, si fuerit
 abscissa = 1, applicatum fore $= \frac{1}{2} = \frac{1}{2}$ | $\frac{1}{2}$ das ist $y = \frac{1}{2}$
 2 $\frac{1}{2}$
 3 $\frac{1}{4}$
 4 vel major infinitum.
 Ich schreibe mit aller Aufmerksamkeit das fest, was
 Poisson hierhergebrachten
 Moskau d. 1. Jun. St. N. 1742. J. Erdmann

世界三大数学猜想之一：哥德巴赫猜想

哥德巴赫1742年给欧拉的信中，哥德巴赫提出了以下猜想：任一大于2的偶数都可写成两个质数之和。但是哥德巴赫自己无法证明它，于是就写信请教赫赫有名的大数学家欧拉帮忙证明，但是一直到死，欧拉也无法证明。

In 1742, Christian Goldbach, a German amateur mathematician, sent a letter to Leonhard Euler in which he made the following conjecture:
Every even number greater than 4 can be written as the sum of two odd prime numbers.

$$8=3+5$$

$$20=3+17$$

$$42=5+37$$

$$80=?+?$$

$$98=?+?$$

$$128=?+?$$

请写一个程序
检验哥德巴赫猜想

程序检验：哥德巴赫猜想

```
def isPrime(a):  
    if a<2:  
        return False  
    for i in range(2,a):  
        if a % i == 0:  
            return False  
    return True
```

质数判断功能函数

输入猜想检验的范围n

```
x=int(input("Please input a number: "))
```

```
for n in range(6,x+1,2):
```

用枚举法检验猜想

```
    for i in range(3,x):
```

```
        if isPrime(i) and isPrime(n-i):
```

```
            print(n,"=",i,"+",n-i)
```

```
            break
```

程序检验：哥德巴赫猜想

文件名Goldbach-conjecture.py

```
def isPrime(a):  
    if a<2:  
        return False  
    for i in range(2,a):  
        if a % i == 0:  
            return False  
    return True
```

质数判断功能函数

```
x=int(input("Please input a number: "))
```

```
primelist=[2]  
for i in range(3,x,2):  
    if isPrime(i):  
        primelist.append(i)
```

```
for n in range(6,x+1,2):  
    for i in range(3,x):  
        if (i in primelist) and ((n-i) in primelist):  
            print(n,"=",i,"+",n-i)  
            break
```

输入猜想检验的范围n

建立质数列表

用枚举法检验猜想

讨论题：解一元一次方程

有一家专门从事计算器改良与升级的实验室，最近该实验室收到了某公司所委托的一个任务：需要在该公司某型号的计算器上加上解一元一次方程的功能。实验室将这个任务交给了一个刚进入的新手ZL先生。为了很好的完成这个任务，ZL先生首先研究了一些一元一次方程的实例：

$$\begin{aligned}4+3x&=8 \\ 6a-5+1&=2-2a \\ -5+12y&=0\end{aligned}$$

在计算器上键入的一个一元一次方程中，只包含整数、小写字母及+、-、=这三个数学符号（符号“-”既可作减号，也可作负号）。方程没有括号，也没有除号，方程中表示未知数的是单个小写字母。请编写程序，解输入的一元一次方程，将解方程的结果输出至屏幕。

输入样例：6a-5+1=2-2a

输出样例：a=0.75

输入样例：12-5y=0

输出样例：y=2.4

输入样例：5x+x=6

输出样例：x=1.0