实验 序列数据类型-列表和元组

```
# 打印你的学号和姓名
print('姓名：梁皓然')

print('学号：2018302100035')
```

```
姓名：梁皓然
学号：2018302100035
```

1. 编写程序，生成包含20个随机数的列表，然后将前10个元素按升序排序，后10个元素按降序排序，并输出处理后的列表。

```
import random
x=[random.randint(1,100) for i in range(20)]
a=sorted(x[:10])
b=sorted(x[10:],reverse=True)
x=a+b
print(x)
```

```
[6, 13, 15, 18, 26, 43, 71, 74, 86, 98, 100, 92, 91, 81, 62, 62, 30, 30, 20, 19]
```

2. 计算并输出杨辉三角形（要求打印出 10 行）。

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

```
def triangle():
    N = [1]
    while True:
        yield N      #generator特点在于：在执行过程中，遇到yield就中断，下次又继续执行
        N.append(0)  #每次都要在最后一位加个0，用于后续的叠加
        N = [N[i]+N[i-1] for i in range(len(N))]

def print_triangle(x):
    a = 0
    for t in triangle():  #这里可以每次调用一个N（得力于Yield函数）
        print(t)
        a += 1
        if a ==x:
            break
print_triangle(10) #打印10行
```

```
[1]
[1, 1]
[1, 2, 1]
[1, 3, 3, 1]
[1, 4, 6, 4, 1]
[1, 5, 10, 10, 5, 1]
[1, 6, 15, 20, 15, 6, 1]
[1, 7, 21, 35, 35, 21, 7, 1]
[1, 8, 28, 56, 70, 56, 28, 8, 1]
[1, 9, 36, 84, 126, 126, 84, 36, 9, 1]
```

3. 编写程序，生成一个列表，包含25个1-10之间的随机数，并统计列表中有几个不一样的数及其出现的次数。

```
import random

list_1 = []
#随机生成包含25个1-10的整数
for i in range(25):
    list_1.append(random.randint(1,10))
print('生成25个随机整数的列表为：',list_1)
#计算列表中每个元素出现的次数
for i in range(1,11):
    list_2 = []
    #计算元素个数并添加到列表list_2中
    list_2.append(list_1.count(i))
    #将列表转换成集合
    s_1 = set(list_2)
    print('其中元素%s出现的次数为:'%(i),s_1)
```

```
生成25个随机整数的列表为： [1, 8, 10, 4, 6, 8, 9, 3, 2, 1, 4, 5, 4, 6, 8, 4, 7, 5, 9,
2, 1, 1, 10, 7, 7]
其中元素1出现的次数为: {4}
其中元素2出现的次数为: {2}
其中元素3出现的次数为: {1}
其中元素4出现的次数为: {4}
其中元素5出现的次数为: {2}
其中元素6出现的次数为: {2}
其中元素7出现的次数为: {3}
其中元素8出现的次数为: {3}
其中元素9出现的次数为: {2}
其中元素10出现的次数为: {2}
```

4. 你可以编写更多程序。（选做，希望尽力完成）

（1）编写程序降序输出7家全国零售百强电商的销售额。

下面是2017年全国零售百强电商销售额排名靠前的7家企业，请对7家企业的销售额进行降序排名并输出。

聚美优品，58亿元；京东，12945亿元；亚马逊中国，391亿元；当当网，357亿元；唯品会，728亿元；考拉严选，116亿元；天猫，21086亿元。

降序排名的输出样式：

```
企业  销售额（亿元）
天猫  21086
京东  12945
```

...

```
x_brands = [
    ["聚美优品",58,],
    ["京东",12945],
    ["亚马逊中国",391],
    ["当当网",357],
    ["唯品会",728],
    ["考拉严选",116],
    ["天猫",21086],
]

x_title = ["企业","销售额（亿元）"]
sorted_numbers=sorted(x_brands, key=lambda brand: brand[1], reverse=True)
sorted_numbers.insert(0,x_title)
for m in range(8):              #生成0、1、2、3、4、5、6、7的列表（第一维）
    for n in range(2):          #生成列表
        print(sorted_numbers[m][n],end=' ')  #m行、n列，换行
    print()       #打印完一行，换行
```

```
企业 销售额（亿元）
天猫 21086
京东 12945
唯品会 728
亚马逊中国 391
当当网 357
考拉严选 116
聚美优品 58
```

（2）埃拉托斯特尼筛法是一种优雅的算法，用于确定不超过 n 的所有素数。

基本思想是首先创建从 2 到 n 的数字列表。第一个数字从列表中删除，并作为素数公布，而且将该数字的所有倍数从列表中删除。此过程一直持续到列表为空。

例如，如果我们希望找到不超过 10 的所有素数，该列表最初将包含 2、 3、 4、 5、 6、 7、8、9、 10。 2 被删除并宣布为素数。然后 4、 6、 8 和 10 被删除，因为它们是 2 的倍数。这会留下 3、5、 7、 9。重复该过程， 3 被宣布为素数并删除，并且 9 被删除，因为它 3 的倍数。这会留下 5 和 7。算法继续宣布 5 是素数，并将它从列表中删除。最后， 7 被宣布和删除，我们完工了。

编写一个程序提示用户输入 n，然后用筛选算法找出小于或等于 n 的所有素数。

```python
def sieve_of_eratosthenes(n):#埃拉托色尼筛选法，返回少于n的素数
    primes = [True] * (n+1)#范围0到n的列表
    p = 2#这是最小的素数
    while p * p <= n:#一直筛到sqrt(n)就行了
        if primes[p]:#如果没被筛，一定是素数
            for i in range(p * 2, n + 1, p):#筛掉它的倍数即可
                primes[i] = False
        p += 1
    primes = [element for element in range(2, n) if primes[element]]#得到所有少于n的素数
    return primes
print(sieve_of_eratosthenes(100))
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97]
```