```
"""
1、随机函数返回值为0,1，随机生成0~1000的数
"""
import math
from random import randint

import pytest

@pytest.mark.parametrize('nums',[100])
def test_random(nums):

a=0
for i in range(0,nums):
a=randint(0,1)+a

print(a)



"""
2、判断一个数是不是素数
"""

class Test_shushu():
"方法一：直接遍历法"
@classmethod
def test1(self,nums):
if nums<=1 :
return False
for i in range(2,nums):
if nums %i ==0:
return False

return True

@classmethod
def test2(self,nums):
if nums<=1 :
```

return False

a= int(math.sqrt(nums))

for i in range(2,a):

if nums %i ==0:

return False

return True

"""

3、获取1-100之间的全部的素数

"""

def get_allshushu():

result=[]

for i in range(2,101):

if i <= 2:

continue

flag=True

for j in range(2, i):

if i % j == 0:

flag=False

break

if flag:

result.append(i)

return result

4、回文数

```
class Solution(object):
    def isPalindrome(self, x):
        """
        :type x: int
        :rtype: bool
        """

        if x < 0 or (x % 10 ==0 and x!=0) :
            return False
        result = 0
        temp=x
```

```
        while x > 0:
            result = result*10+x%10
            x= x//10

        return True if temp == result else False
```

```
1
```

5、求一个区间素数的个数

```
def get_allshushu():
result=[]
for i in range(2,101):
# if i <= 2:
# continue
flag=True
for j in range(2, i):
if i % j == 0:
flag=False
break
if flag:
result.append(i)

return result
```

6、找到出现次数超过一半的数字（对算法题测试）

```
class Solution:
def majorityElement(self, nums: List[int]) -> int:
nums.sort()
return nums[len(nums) // 2]
```

7、长度大于等于3的回文（编写测试算法的测试用例）

8、一个数组 查找目标数字最后一次出现的位置（这个数组写测试用例，写出来所有可能得情况）

9、输出并统计1~99999的回文数(这个在自我介绍结束后）

10、随机函数返回值为0,1，随机生成0~1000的数

```
from random import randint
def test_random(nums):
a=0
for i in range(0,nums):
a=randint(0,1)+a
print(a)
```

)

11、生成6位的随机密码锁（参考门禁）

12、136. 只出现一次的数字

1、数组的方式；

时间复杂度：O(n^2)

) 。我们遍历 \text{nums}nums 花费 O(n)O(n) 的时间。我们还要在列表中遍历判断是否存在这个数字，花费 O(n)O(n) 的时间，所以总循环时间为 O(n^2)O(n

2

) 。

空间复杂度：O(n)O(n) 。我们需要一个大小为 nn 的列表保存所有的 \text{nums}nums 中元素=-

2、hash表的用法：查找和存取的速度都很块，查找的最佳操作

数组方式

```
class Solution(object):
def singleNumber(self, nums):
"""
:type nums: List[int]
:rtype: int
"""
new_list=[]
for i in nums:
if i not in new_list:
new_list.append(i)
else :
new_list.remove(i)
return new_list[0]
```

hasnMap的方式：

```
class Solution {
public int singleNumber(int[] nums) {
Map<Integer,Integer> map=new HashMap<>(); //哈希表的方式
if(nums!=null && nums.length>0){
```

```
for(int n:nums){
if(!map.containsKey(n)){
map.put(n,1);
}else{
map.remove(n);
}
}
for (Integer i : map.keySet()) {
Integer count = map.get(i);
if (count == 1) {
return i;
}
}
}
return -1;
}
}
```

3、异或的方式：

```
class Solution {
public int singleNumber(int[] nums) {
int temp=0;
if(nums!=null && nums.length>0){
for(int n:nums){
temp=temp^n;
}

return temp;
}
return -1;
}
}
```

13、输入形如‘20220601’，算出这是今年的第几天