**2021211950-汪超-信安21-1**

1、编写代码：利用 if 判断来制作一个猜数字的小游戏

问题描述：程序运行时，系统在指定范围内生成一个随机数字，然后用户进行猜测，并

根据用户输入进行必要的提示（right, too large, too small），如果猜对则提前结束程序，

如果未有猜对，提示游戏结束并给出正确答案。

**import random**

**count = 3**

**n = random.randint(1, 100)**

**print('你一共有3次机会')**

**for i in range(count):**

**x = int(input('请输入猜测的数字：'))**

**if x == n:**

**print('right')**

**break**

**elif x < n:**

**print('too small')**

**else:**

**print('too large')**

**if count-i-1 != 0:**

**print(f'你还有{count-i-1}次机会')**

**else:**

**print(f'正确答案为：{n}')**

****

2、编写代码：利用 while 循环判断来制作一个猜数字的小游戏

问题描述：程序运行时，系统在指定范围内生成一个随机数字，然后用户进行猜测，并

根据用户输入进行必要的提示（right, too large, too small），如果猜对则提前结束程序，

如果次数用完仍没有猜对，提示游戏结束并给出正确答案。

**import random**

**x = random.randint(1, 10)**

**print(x)**

**print('请猜一个[1, 10]之间的数字,而且你有三次机会：')**

**i = 0**

**while i < 3:**

**i = i + 1**

**y = int(input())**

**if y == x:**

**print('right, finish!')**

**break**

**elif y > x:**

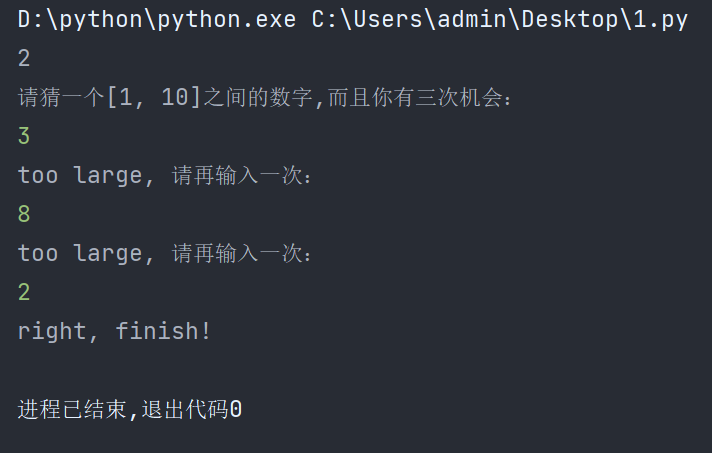
**print('too large, 请再输入一次：')**

**elif y < x:**

**print('too small, 请再输入一次：')**

**if i == 3:**

**print('game is over, the right number is ', x)**



3、有 1、2、3、4 个数字，能组成多少个互不相同且无重复数字的三位数？都是多少？

**count = 0**

**for i in range(1, 5):**

**for j in range(1, 5):**

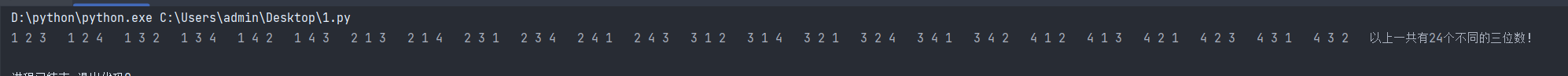
**for k in range(1, 5):**

**if i != j and i != k and j != k:**

**print(i, j, k)**

**count = count + 1**

**print(f'以上一共有{count}个不同的三位数！')**



4、一个数如果恰好等于它的因子之和，这个数就称为"完数"。例如 6=1＋2＋3.编程找

出 1000 以内的所有完数

**import math**

**def demo(n):**

**lst = []**

**for i in range(1, n):**

**if n % i == 0:**

**lst.append(i)**

**return lst**

**for a in range(2, 1001):**

**lst1 = demo(a)**

**if sum(lst1) == a:**

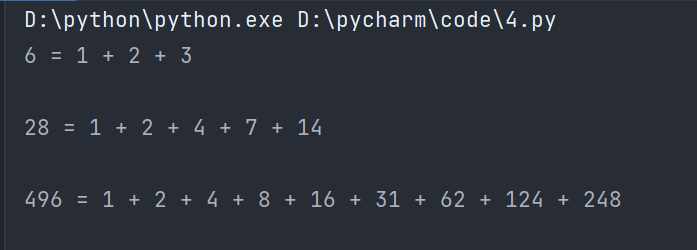
**print(a, "= 1", end="")**

**for b in lst1:**

**if b != 1:**

**print(" +", b, end="")**

**print('\n')**



5、编写程序，生成一个包含 50 个随机整数的列表，然后删除其中所有奇数（提示：从

后向前删除）

**import random**

**lst = [random.randint(1, 100) for i in range(50)]**

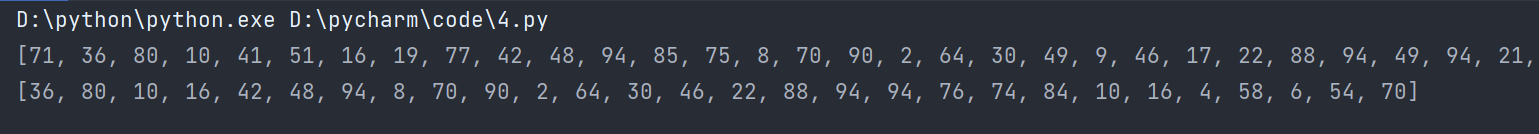
**print(lst)**

**for i in range(len(lst)-1, -1, -1):**

**if lst[i] % 2 == 1:**

**del lst[i]**

**print(lst)**



6、编写程序，生成一个包含 20 个随机整数的列表，然后对其中偶数下标的元素进行降

序排列，奇数下标的元素不变。

**import random**

**lst = [random.randint(1, 100) for i in range(20)]**

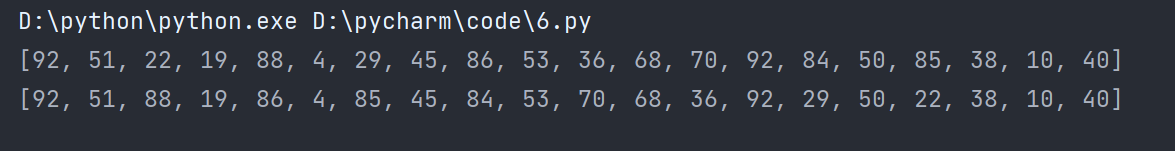
**print(lst)**

**lst1 = lst[::2]**

**lst1.sort(reverse=True)**

**lst[::2] = lst1**

**print(lst)**



7、编写函数用来计算斐波那契数列中小于参数 n 的所有值。

**def fib(m):**

**a, b = 1, 1**

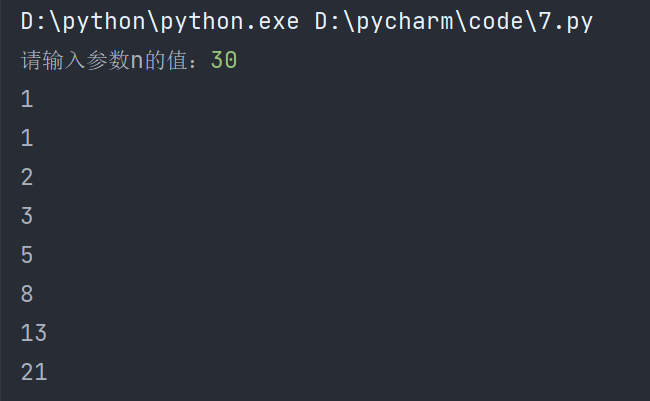
**while a < m:**

**print(a)**

**a, b = b, a + b**

**n = int(input('请输入参数n的值：'))**

**fib(n)**



8、问题描述：编写程序，输入一个大于 2 的自然数，然后输出小于该数字的所有素数组成的列表。

**import math**

**def demo(m):**

**for i in range(2, int(math.sqrt(m))+1):**

**if m % i == 0:**

**return 0**

**return 1**

**n = int(input('请输入一个大于2的自然数：'))**

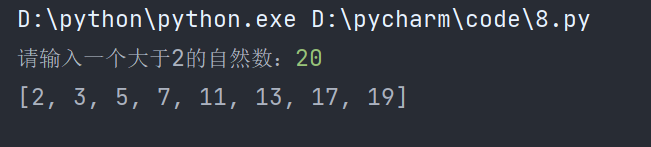
**lst = []**

**for a in range(2, n):**

**if demo(a):**

**lst.append(a)**

**print(lst)**



1. 编写函数：判断回文，也就是正读反读都一样的字符串。 例如：abcba”

**def demo(a):**

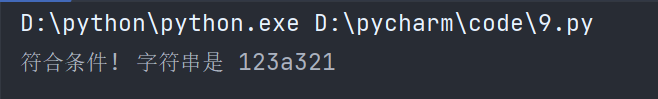
**if a == a[::-1]:**

**print('符合条件! 字符串是', a)**

**else:**

**print('不符合条件！')**

**demo('123a321')**

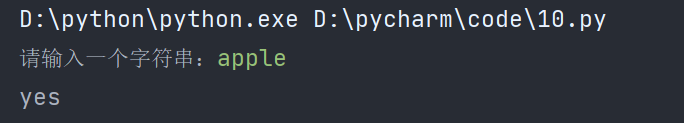


10、对于给定的一个字符串，如果其中包含"apple"（apple 不区分大小写)则输出 yes，

否则输出 no。

**s = input('请输入一个字符串：')**

**print('yes' if s.lower().count('apple') else 'no')**



11、编写函数：随机产生包含 n 个整数的列表，返回一个元组，其中第一个元素为所

有参数的平均值，其他元素为所有参数中大于平均值的整数。

例如： 随机产生长度为 3 的列表[2, 1, 3]，输出为（2.0, 3）

**def demo(a):**

**avg = sum(a)/len(a)**

**n = [avg]**

**for i in a:**

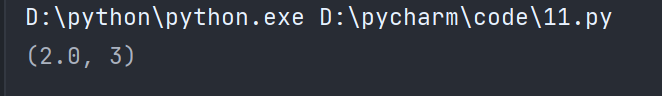
**if i > avg:**

**n.append(i)**

**return tuple(n)**

**lst = [2, 1, 3]**

**print(demo(lst))**



12、编写函数： 一年 365 天， 每周工作 5 天，休息 2 天，休息日水平下降 0.01，工

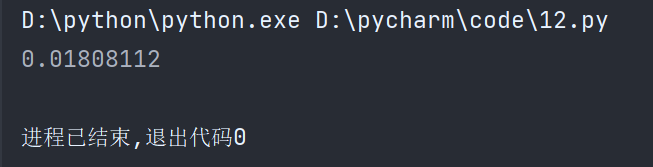
作日要努力到什么程度一年后的水平才与每天努力 1%所取得的效果（即 37.78倍）一样呢？

**import math**

**weekend = 365//7\*2**

**workday = 365-weekend**

**print('%.8f' % (math.pow(37.78/(0.99\*\*weekend), 1/workday)-1))**



13、查找两个字符串首尾交叉的最大子串长度，连接两个字符串，首尾交叉部分只保留

一份。例如，1234 和 2347 连接为 12347

要求：程序中使用 lambda 函数

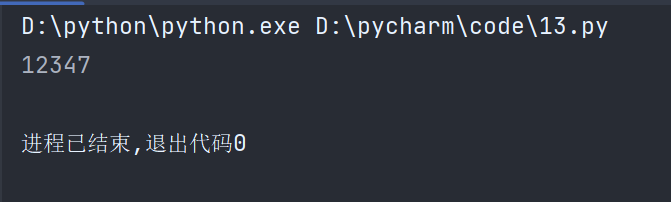
**def demo(x, y):**

**length = min(len(x), len(y))**

**stop = max(range(1, length+1), key=lambda i: i if x[len(x)-i:] == y[:i] else False) # range左包右不包 切片也是**

**return x+y[stop:]**

**print(demo('1234', '2347'))**



14，写一个删除列表中重复元素的函数，要求去重后元素相对位置保持不变。

**def func(x):**

**lst = []**

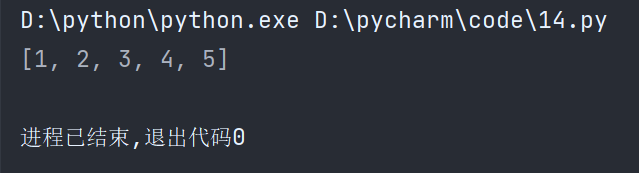
**for item in x:**

**if item not in lst:**

**lst.append(item)**

**return lst**

**print(func([1, 2, 3, 1, 1, 1, 2, 1, 3, 4, 5]))**



15、编写函数，计算传入函数的字符串中，数字、字母、空格以及其他字符的个数，

并返回结果。

**def select(x):**

**num, alpha, space, other = 0, 0, 0, 0**

**for i in x:**

**if i.isdigit():**

**num = num + 1**

**elif i.isalpha():**

**alpha = alpha + 1**

**elif i.isspace():**

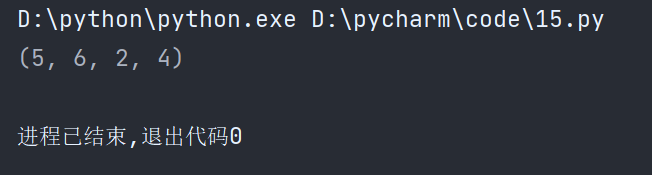
**space = space + 1**

**else:**

**other = other + 1**

**return num, alpha, space, other**

**print(select('ae123+dq+2 4() dq'))**



16、列表推导式与字典的应用

（1）.问题描述：编写程序，先生成包含 1000 个随机字符的字符串，然后统计每个字

符出现的次数。；

**import string**

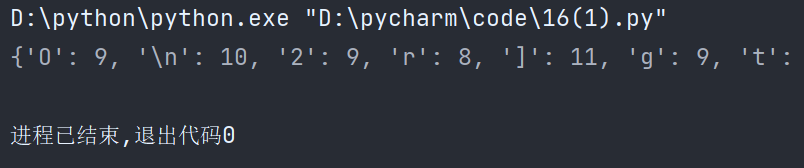
**import random**

**from collections import Counter**

**x = ''.join(random.choice(string.printable) for i in range(1000))**

**res = Counter(x)**

**print(dict(res))**



（2）.问题描述：编写一个程序，接受逗号分隔的单词序列作为输入，按字母顺序排序

后按逗号分隔的序列打印单词。假设向程序提供以下输入: without,hello,bag,world

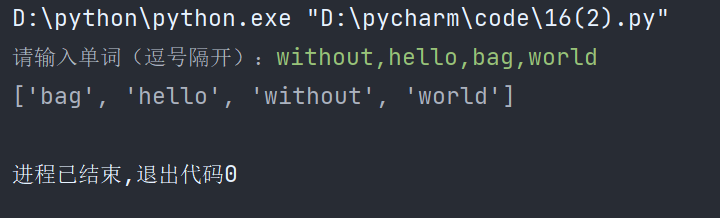
则输出为: bag,hello,without,world

**x = input('请输入单词（逗号隔开）：')**

**y = list(x.split(','))**

**y.sort()**

**print(y)**



17、集合的应用

（1）.问题描述：编写程序，输入两个集合 setA 与 setB，分别输出它们两个交集的交、

差、并。

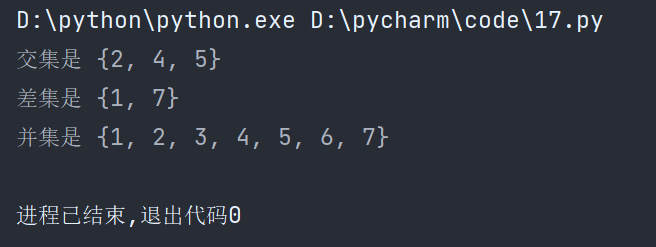
**A = {1, 2, 4, 5, 7}**

**B = {3, 4, 2, 5, 6}**

**print('交集是', A & B)**

**print('差集是', A - B)**

**print('并集是', A | B)**



（2）.编写一个程序，接受一系列空格分隔的单词作为输入，然后删除所有重复的单词

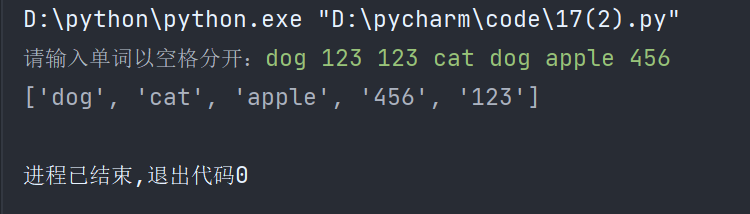
并按字母数字排序后打印这些单词。

**x = set(input('请输入单词以空格分开：').split(' '))**

**x = list(x)**

**x.sort(reverse=True)**

**print(x)**



18、字符串与列表推导式的应用

问题描述： 编写程序，生成含有 n 个元素的嵌套列表，即列表的每个元素仍是列表，

要求列表中的元素是长度不超过 m 的数字或字符组成的字符串，并按照字符串长度降序输

出结果

**m, n = 6, 6**

**y = []**

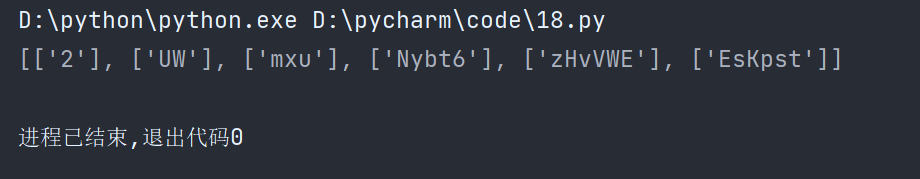
**for i in range(n):**

**a = random.randint(1, m)**

**x = [''.join(random.choice(string.digits + string.ascii\_letters) for \_ in range(a))]**

**y.append(x)**

**print(sorted(y, key=lambda b: len(b[0])))**



19、列表与切片的应用

问题描述：编写程序，生成一个整型列表，输出包含原列表中所有元素的新列表、包含

原列表中所有元素的逆序列表，以及输出具有偶数位置的元素列表。

**import random**

**lst = list(random.randint(1, 100) for i in range(10))**

**A = lst[::]**

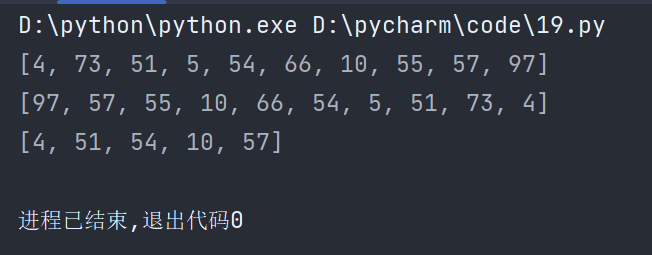
**B = lst[::-1]**

**C = lst[::2]**

**print(A)**

**print(B)**

**print(C)**



20、元组的应用

问题描述：编写程序，利用生成器推导式生成包含 n 个整数元素的元组，每个元素值不

大于 m，并过滤掉偶数整数，并输出。

**import random**

**n, m = 10, 8**

**x = tuple(random.randint(1, m) for i in range(n))**

**print(x)**

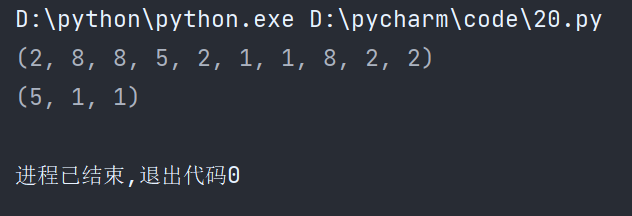
**x1, x2 = list(x), []**

**for i in x1:**

**if i % 2 == 1:**

**x2.append(i)**

**print(tuple(x2))**



21、字典的应用

问题描述：编写程序，输入一段英文，统计每个单词出现的次数并存储到字典进行输出。

例如：输入：“I love apple apple”，输出：I：1 love: 1 apple: 2

**x = list(input('请输入一段英文（以空格分开）:').split(' '))**

**y = list(set(x))**

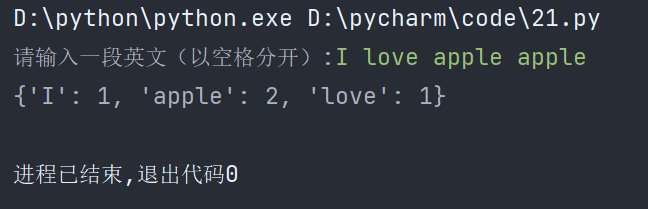
**z = []**

**for i in y:**

**a = x.count(i)**

**z.append(a)**

**print(dict(zip(y, z)))**



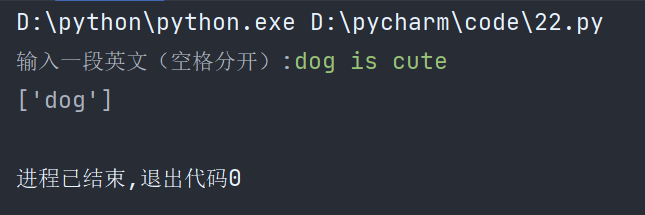
22、用户输入一段英文，使用正则表达式查找这段英文中所有长度为 3 个字母的单词。

（提示：可以调用 findall 函数，也可以先调用 split 函数将字符串进行分隔，再搜索长度为 3 的单词。）

**import re**

**s = input('输入一段英文（空格分开）:')**

**print(re.findall(r'\b[a-zA-Z]{3}\b', s))**



23、生成长度为 n 的字母与数字随机组成的字符串，使用正则表达式查找字符串中最

长的字母子串。

例如：3423ab45defgD 字符串中的最长字符串为 defgD

**import re**

**import random**

**import string**

**n = 20**

**s = ''.join(random.choice(string.digits+string.ascii\_letters) for \_ in range(n))**

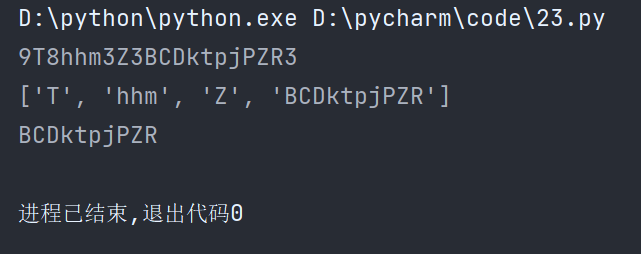
**print(s)**

**z = re.findall('[a-zA-Z]+', s)**

**print(z)**

**x = max(z, key=lambda i: len(i))**

**print(x)**



24、将字符串："a:1\*a1:2\*a2:3\*a3:4"，处理成 python 字典：{'a':'1', 'a1':'2',

'a2':'3','a3':'4' }

**s = "a:1\*a1:2\*a2:3\*a3:4"**

**z = s.split('\*')**

**a, b = [], []**

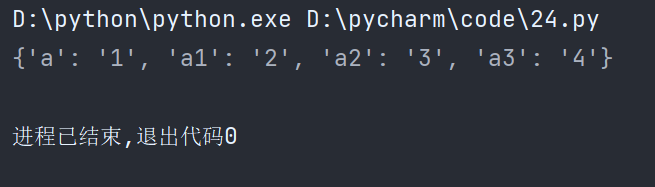
**for i in z:**

**c = i.split(':')**

**a.append(c[0])**

**b.append(c[1])**

**print(dict(zip(a, b)))**



25、用 Python 实现一个二分查找的函数

**for key in range(7, 13):**

**lst = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]**

**while len(lst) != 0:**

**mid = len(lst)//2**

**if lst[mid] == key:**

**print(key, 'True')**

**break**

**elif lst[mid] < key:**

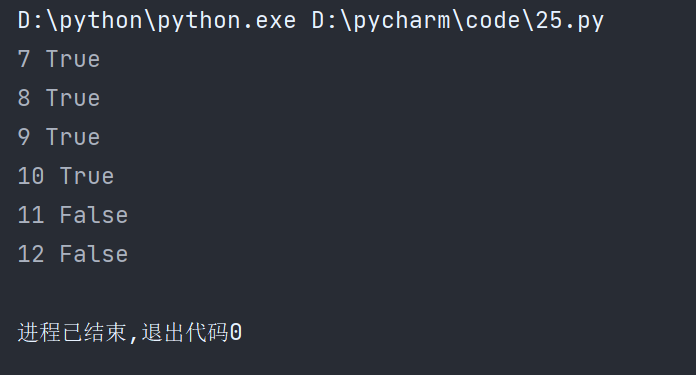
**lst = lst[mid+1:]**

**else:**

**lst = lst[:mid]**

**else:**

**print(key, 'False')**



26、一个列表中保存了一些整数，请输出列表中重复的数字，以及重复的次数

**import random**

**s = [random.randint(1, 10) for \_ in range(10)]**

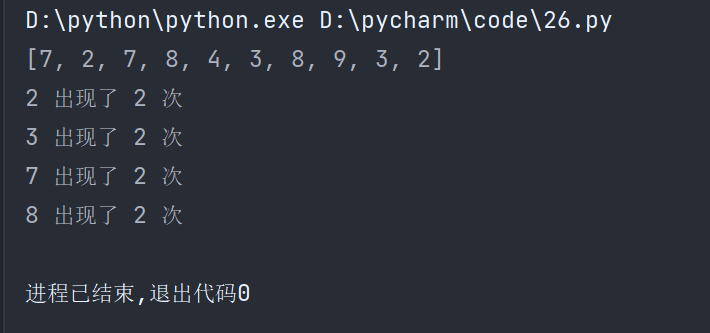
**print(s)**

**x = list(set(s))**

**for i in x:**

**if s.count(i) != 1:**

**print(i, '出现了', s.count(i), '次')**

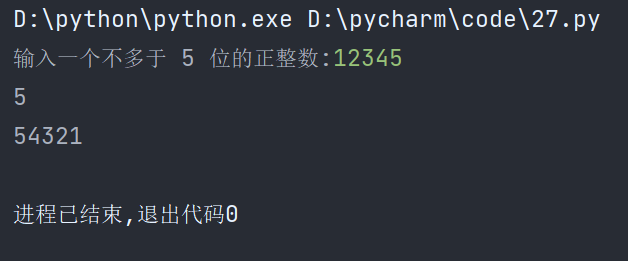


27、输入一个不多于 5 位的正整数，请输出它的位数，以及逆序输出各位数字。

**x = int(input('输入一个不多于 5 位的正整数:'))**

**print(len(str(x)))**

**print(str(x)[::-1])**



28、随机产生一个 10 以内的正整数，由用户去猜这个数字，程序根据用户猜测的数字

给出不同的提示，如果用户猜 3 次不能正确，退出程序。

**import random**

**x = random.randint(1, 10)**

**n = 3**

**print('你有3即机会')**

**while n > 0:**

**m = int(input('请输入你的猜测：'))**

**if m == x:**

**print('right!')**

**break**

**elif m > x:**

**print('too large!')**

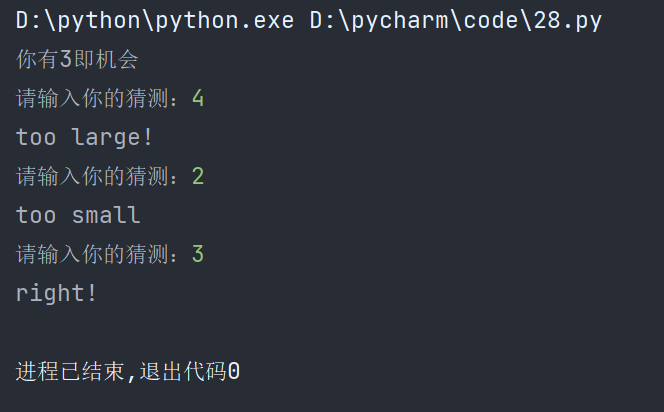
**else:**

**print('too small')**

**n = n - 1**

**else:**

**print('这个数是；', x)**



29. 问题描述： 编写关于学生信息的类，实例属性包括：学号、姓名、性别，班级、n 门

课程成绩，要求：

1) 利用文件读取，创建一个包含 n 个学生的班级;

文件存储格式以空格分隔例如： 202001 李四 1 班 高数:89 英语:28 计算机:77

2) n 门课程成绩利用字典存储，支持成绩录入、修改与删除;

3) 求解每个学生的 n 门成绩的平均值，及其平均值排名。并按照成绩平均成绩排名

正序输出：学号、姓名、性别、n 门课程成绩，n 门课程平均值，排名。

**class Student:**

**def \_\_init\_\_(self):**

**print('\*'\*10 + '学生信息管理' + '\*'\*10)**

**def show(self):**

**print('现有的学生信息如下......')**

**with open('29.txt', 'r', encoding='UTF8') as a:**

**lst = a.read().split()**

**lst2, average = [], 0**

**for i in range(len(lst) // 10):**

**self.num = lst[i\*10+0]**

**self.name = lst[i\*10+1]**

**self.sex = lst[i\*10+2]**

**self.room = lst[i\*10+3]**

**print('学号：', self.num, ' 姓名：', self.name, ' 性别：', self.sex, ' 班级：', self.room)**

**grade = dict(zip(['高数', '英语', '计算机'], [lst[i\*10+5], lst[i\*10+7], lst[i\*10+9]]))**

**lst1, he = list(grade.values()), 0**

**for i in lst1:**

**he += int(i)**

**average = he / len(lst1)**

**print('各科成绩为：', grade, '平均成绩是：', average)**

**lst2.append(average)**

**lst2.sort(reverse=True)**

**a.close()**

**with open('29.txt', 'r', encoding='UTF8') as b:**

**lst3 = b.read().split()**

**for i in range(len(lst3)//10):**

**rank = lst2.index((int(lst[i\*10+5]) + int(lst[i\*10+7]) + int(lst[i\*10+9]))/3) + 1**

**print(lst3[i\*10+1], '排名是：', rank)**

**b.close()**

**def logging(self): # 信息录入**

**with open('29.txt', 'a', encoding='UTF8') as b:**

**s = input('请输入学生信息（学号，姓名，性别，班级，各科成绩）：').split(' ')**

**b.write('\n' + s[0]+' '+s[1] + ' ' + s[2] + ' ' + s[3] + ' 高数： ' + s[4] + ' 英语： ' + s[5] + ' 计算机： ' + s[6])**

**b.close()**

**def delete(self): # 删除成绩**

**with open('29.txt', 'r', encoding='UTF8') as c:**

**num = input('请输入需要删除的同学的学号：')**

**lst = c.read().split()**

**x = lst.index(num) // 10**

**with open('29.txt', 'r', encoding='UTF8') as d:**

**line = d.readlines()**

**del line[x]**

**c.close()**

**with open('29.txt', 'w', encoding='UTF8') as e:**

**for item in line:**

**e.write(item)**

**e.close()**

**def alter(self): # 修改成绩**

**with open('29.txt', 'r', encoding='UTF8') as c:**

**num = input('请输入需要修改的同学的学号：')**

**lst = c.read().split()**

**x = lst.index(num) // 10**

**with open('29.txt', 'r', encoding='UTF8') as d:**

**line = d.readlines()**

**s = input('请输入修改后的各项信息：').split(' ')**

**line[x] = s[0]+' '+s[1] + ' ' + s[2] + ' ' + s[3] + ' 高数： ' + s[4] + ' 英语： ' + s[5] + ' 计算机： ' + s[6] + '\n'**

**c.close()**

**with open('29.txt', 'w', encoding='UTF8') as e:**

**for item in line:**

**e.write(item)**

**e.close()**

**m = Student()**

**m.show()**

**m.logging()**

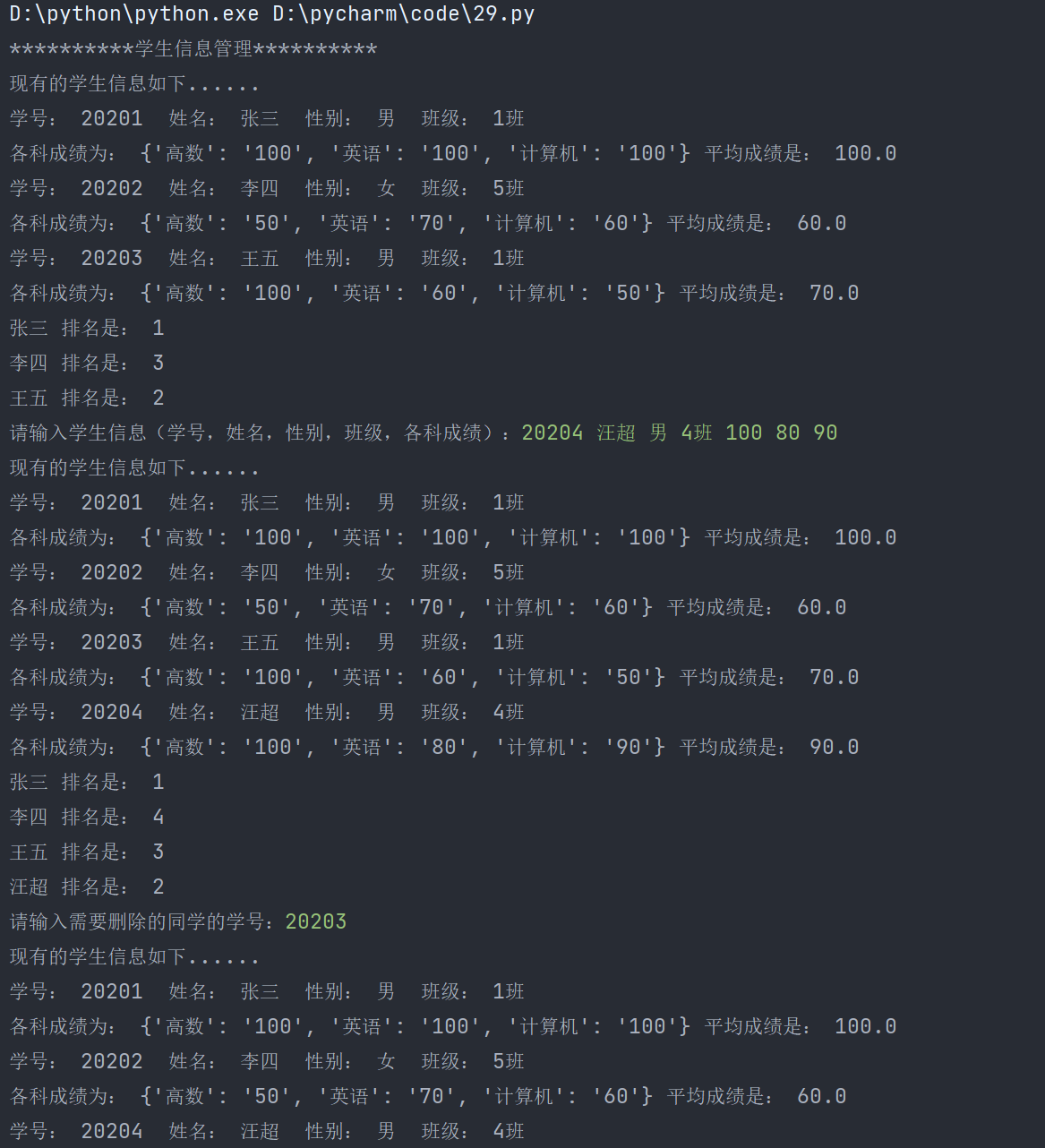
**m.show()**

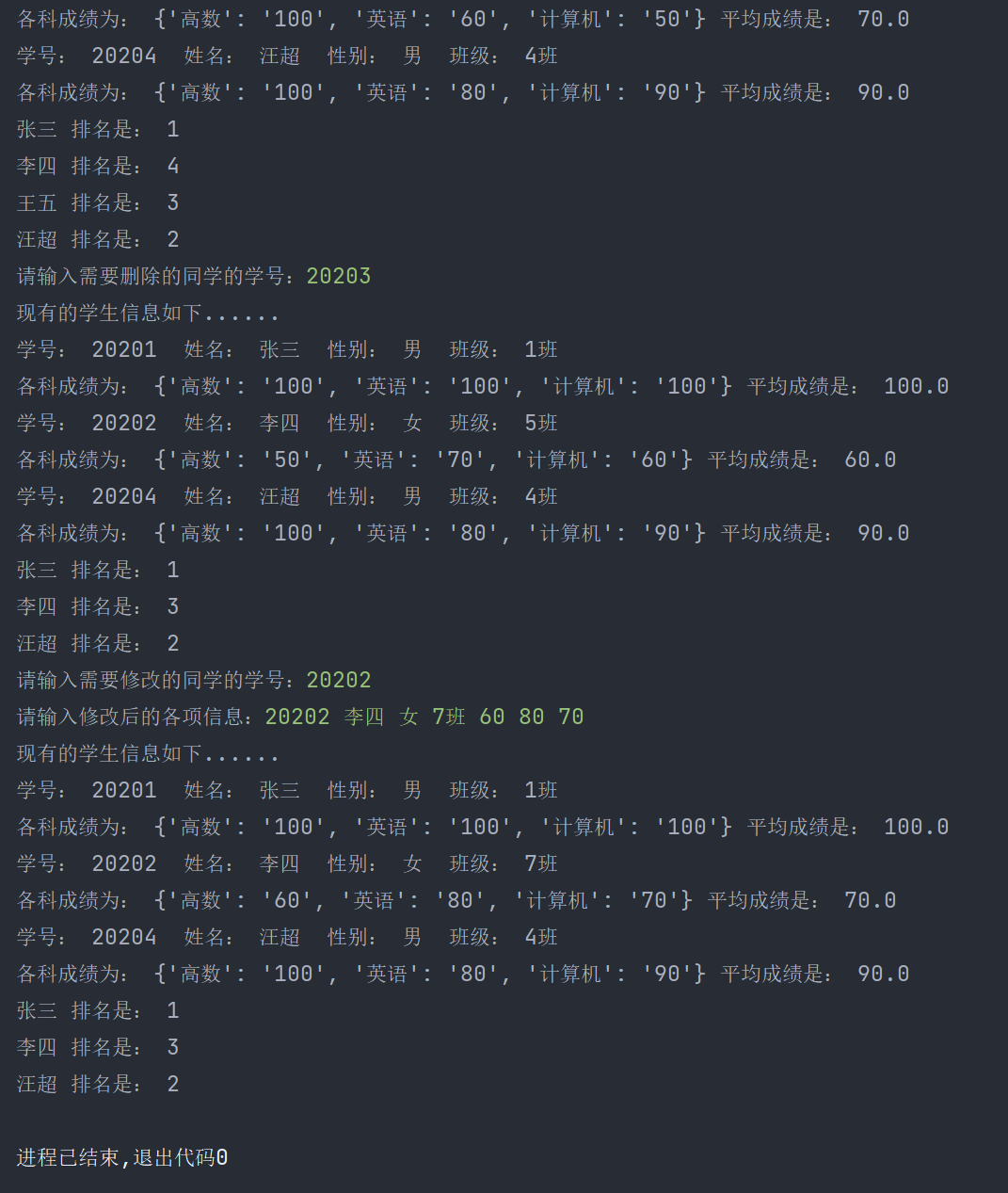
**m.delete()**

**m.show()**

**m.alter()**

**m.show()**





30. 编程设计一个雇员基类 Employee，包括姓名，编号，月薪三个实例属性，月薪计算

pay()和信息显示 show()两个函数成员；派生两个子类 Manager 类和 Salesman 类，重载相应的 2 个函数成员。

要求：根据以上描述设计类，并在主函数创建两个子类的实例化对象，分别调用其成员方法。

**class Employee:**

**def \_\_init\_\_(self, name, num, salary):**

**self.name = name**

**self.num = num**

**self.salary = salary**

**def pay(self):**

**pass**

**def show(self):**

**pass**

**class Manager(Employee):**

**def \_\_init\_\_(self, name, num, salary):**

**super().\_\_init\_\_(name, num, salary)**

**def pay(self):**

**print('调用了Manager子类中的pay')**

**def show(self):**

**print('Manger的姓名是：', self.name, ' 编号是：', self.num, ' 月薪是：', self.salary)**

**class Salesman(Employee):**

**def \_\_init\_\_(self, name, num, salary):**

**super().\_\_init\_\_(name, num, salary)**

**def pay(self):**

**print('调用了Salesman子类中的pay')**

**def show(self):**

**print('Salesman的姓名是：', self.name, ' 编号是：', self.num, ' 月薪是：', self.salary)**

**M = Manager('Tom', '2021211950', '10000')**

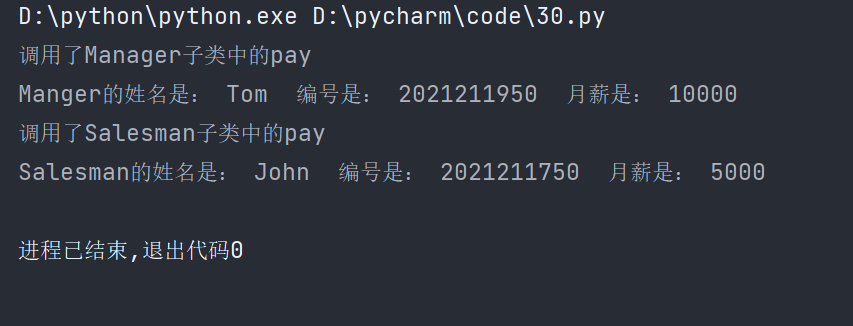
**M.pay()**

**M.show()**

**S = Salesman('John', '2021211750', '5000')**

**S.pay()**

**S.show()**



31.编程设计一个基类汽车类 Vehicle，包含最大速度 MaxSpeed，weight 两个实例私有 属性；设计一个派生子类自行车（Bicycle）类，增加 1 个实例私有属性高度（height）和 1 个成员函数 SetMaxSpeed 实现给父类的实例属性 MaxSpeed 的赋值。

要求：

1）根据以上设计类，并在主函数中创建子类的实例化对象，并设置对象的 MaxSpeed 值。

2）利用 property 将 height 设定为可读、可修改的属性。

**class Vehicle:**

**def \_\_init\_\_(self, MaxSpeed, weight):**

**self.MaxSpeed = MaxSpeed**

**self.weight = weight**

**def show(self):**

**print('汽车的最大速度是：', self.MaxSpeed)**

**print('汽车的重量是：', self.weight)**

**class Bicycle(Vehicle): # 增加 1 个实例私有属性高度（height）**

**def \_\_init\_\_(self, MaxSpeed, weight, height):**

**super().\_\_init\_\_(MaxSpeed, weight)**

**self.\_\_height = height**

**def SetMaxSpeed(self, news): # 1 个成员函数 SetMaxSpeed**

**self.MaxSpeed = news**

**def show(self):**

**print('自行车的最大速度是：', self.MaxSpeed)**

**print('自行车的重量是：', self.weight)**

**print('自行车的高度是：', self.\_\_height)**

**def \_\_get(self):**

**return self.\_\_height**

**def \_\_set(self, h):**

**self.\_\_height = h**

**height = property(\_\_get, \_\_set) # 利用 property 将 height 设定为可读、可修改的属性。**

**a = Vehicle(100, 1000)**

**a.show()**

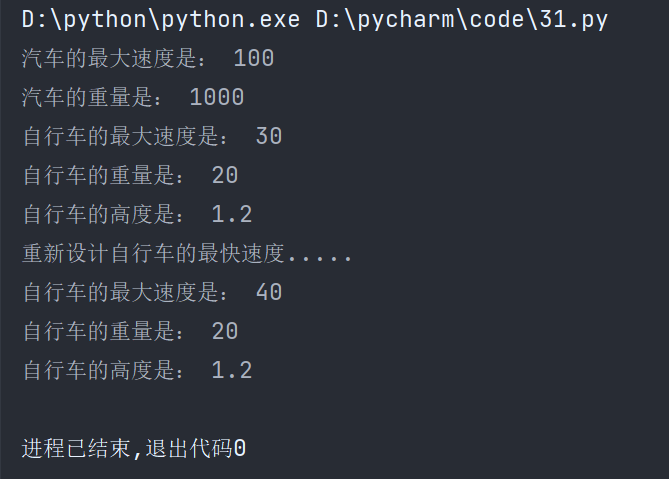
**b = Bicycle(30, 20, 1.2)**

**b.show()**

**print('重新设计自行车的最快速度.....')**

**b.SetMaxSpeed(40)**

**b.show()**



32. 编程设计一个队列类 Myqueue，主要的类成员包括： 3 个数据成员（队列的最大度

size，队列所有数据 data，队列的元素个数 current）和 6 个成员方法如下：

1) 初始化 ：设置队列为空;

2) 判断队列为空：若为空，则返回 TRUE，否则返回 FALSE.

3) 判断队列为满：若为满，则返回 TRUE，否则返回 FALSE.

4) 取队头元素：取出队头元素; 条件：队列不空。 否则，应能明确给出标识.

5) 入队：将元素入队，即放到队列的尾部

6) 出队：删除当前队头的元素

要求：根据以上描述设计类，并在主函数中创建类的实例化对象，构建一个长度为 N 的队

列，分别调用上述成员方法。

**class Myqueue:**

**def \_\_init\_\_(self, size): # 初始化 ：设置队列为空**

**self.size = size**

**self.current = 0**

**self.data = []**

**def full(self): # 判断队列为空：若为空，则返回 TRUE，否则返回 FALSE**

**return len(self.data) == self.size**

**def empty(self): # 判断队列为满：若为满，则返回 TRUE，否则返回 FALSE**

**return self.data == []**

**def top(self): # 取队头元素：取出队头元素 条件：队列不空。否则，应能明确给出标识，以便程序的处理**

**if not self.empty():**

**print('队头元素是：', self.data[0])**

**else:**

**print('队列为空')**

**def inline(self): # 入队：将元素入队，即放到队列的尾部**

**if not self.full():**

**self.data.append(input('请输入入队元素：'))**

**self.current += 1**

**else:**

**print('队列已满无法入队！')**

**def outline(self): # 出队：删除当前队头的元素**

**if not self.empty():**

**print('出队元素是：', self.data[0])**

**del self.data[0]**

**print('改变后的队列为：', self.data)**

**else:**

**print('队列已空无法出队！')**

**text = Myqueue(15)**

**print('队列最大长度是：', text.size)**

**print('给队列输入元素......')**

**for i in range(3):**

**text.inline()**

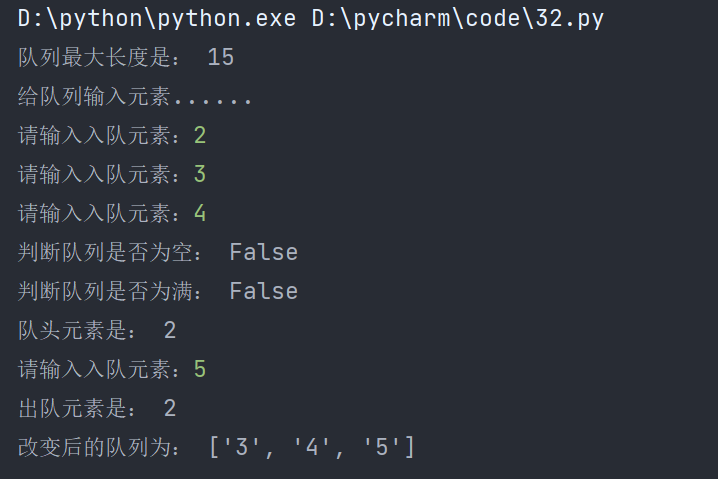
**print('判断队列是否为空：', text.empty())**

**print('判断队列是否为满：', text.full())**

**text.top()**

**text.inline()**

**text.outline()**



33.问题描述：编写程序，生成多个字符串，将字符串写入文件，同时读取当前文件，并输

出统计字符串的个数。

**import random**

**import string**

**with open('33.txt', 'w') as a:**

**for i in range(random.randint(1, 8)):**

**s = ''.join(random.choice(string.digits+string.ascii\_letters) for \_ in range(random.randint(3, 8)))**

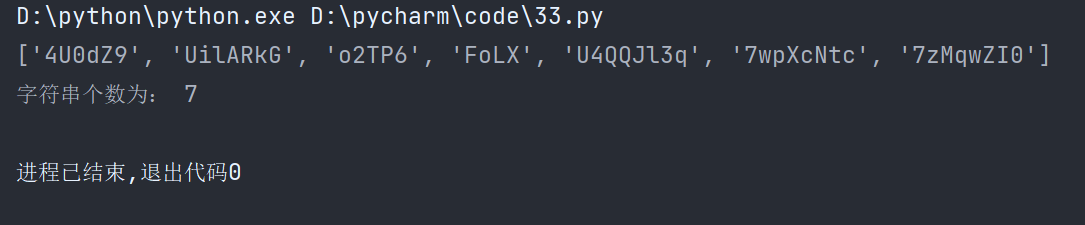
**a.write(s+'\n')**

**with open('33.txt', 'r') as b:**

**m = b.read().split()**

**print(m)**

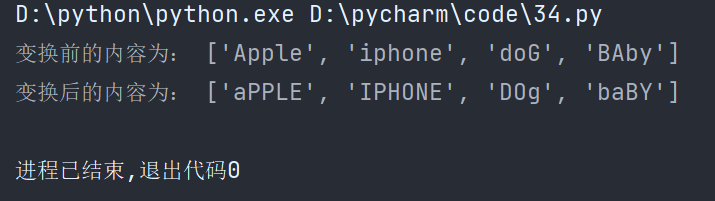
**print('字符串个数为：', len(m))**



34.假设有一个英文文本文件，编写程序读取其内容，并将其中的大写字母变为小写字母，

小写字母变为大写字母

**with open('34-english.txt', 'r') as a:  
 lst, lst1 = a.read().split(), []  
print('变换前的内容为：', lst)  
for i in lst:  
 lst1.append(i.swapcase())  
print('变换后的内容为：', lst1)**



35.假设有一个文本文件的编码为 GBK，编写程序读取其内容，将其转成 UTF-8 编码保存

到另一个文件中。

**with open('35-GBK.txt', 'r', encoding='GBK') as a:**

**a = a.read()**

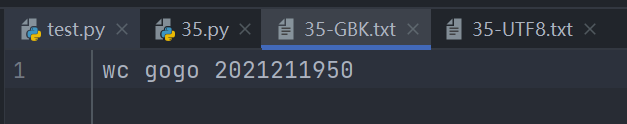
**print('GBK:', a)**

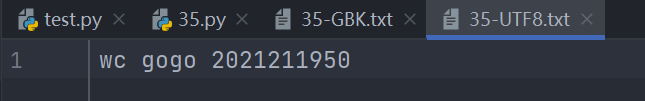
**with open('35-UTF8.txt', 'w', encoding='UTF8') as b:**

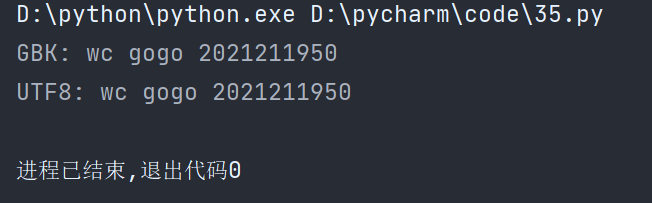
**b.write(str(a))**

**with open('35-UTF8.txt', 'r', encoding='UTF8') as c:**

**print('UTF8:', c.read())**







36.列出文件目录 d:\data 下的所有的文本文件（文件扩展名为.txt）

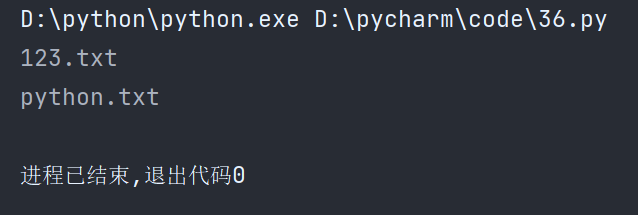
**import os**

**for i in os.listdir('D:\\data'):**

**if i[-3::] == 'txt':**

**print(i)**





37.假设当前目录下有一个文件名为 student.txt 的文本文件，存放着某班学生的学号（第 1 列）、数学成绩（第 2 列）和英语成绩（第 3 列），以空格分割各列数据。一行存放一个 学生的成绩。编写程序完成下列要求：

（1）分别求出这个班数学成绩和英语成绩的最高分（保留 1 位小数）并输出。

（2）找出这个班两门课的平均成绩≥90 分的学生，输出这些学生的学号、数学成绩和英语

成绩和平均成绩。

**with open('student.txt', 'r') as a:**

**lst = a.read().split()**

**a.close()**

**num = lst[::3]**

**math = lst[1::3]**

**english = lst[2::3]**

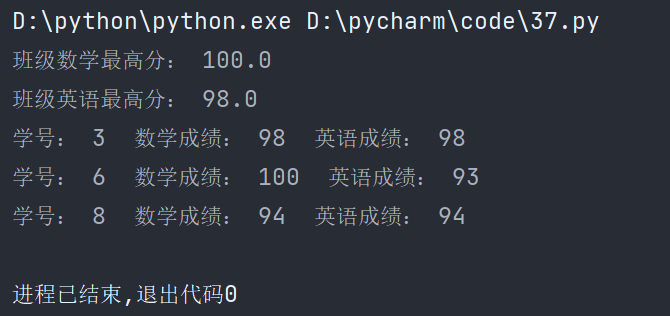
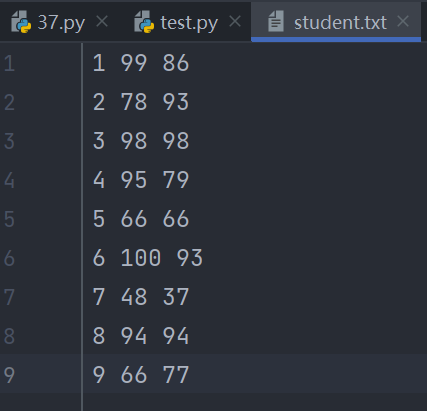
**print('班级数学最高分：', '%.1f' % max(int(i) for i in math))**

**print('班级英语最高分：', '%.1f' % max(int(i) for i in english))**

**for i in range(1, len(num)):**

**if int(math[i]) >= 90 and int(english[i]) >= 90:**

**print('学号：', num[i], ' 数学成绩：', math[i], ' 英语成绩：', english[i])**



38.有 n 个人围成一圈（n 的值由用户输入），顺序排号。从第一个人开始报数（从 1 到 3 报数），凡报到 3 的人退出圈子，问最后留下的是原来第几号的那位。

**n = int(input('请输入人数；'))**

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

**a = len(lst) % 3**

**del lst[2::3]**

**b = len(lst) % 3**

**while len(lst) >= 3:**

**del lst[2-a::3]**

**a = (a + b) % 3**

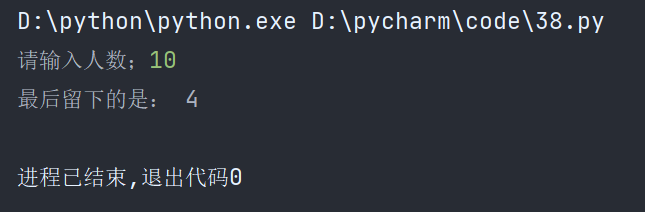
**b = len(lst) % 3**

**if a % 2 == 1:**

**print('最后留下的是：', lst[0])**

**else:**

**print('最后留下的是：', lst[1])**



39.编写程序以检查用户输入的密码的有效性。

检查密码的标准为：

1. [a-z]之间至少有 1 个字母

2. [0-9]之间至少有 1 个数字

1. [A-Z]之间至少有一个字母

3. [$＃@]中至少有 1 个字符

4.最短交易密码长度：6

5.交易密码的最大长度：12

程序接受一系列逗号分隔的密码，进行检查。再输出符合条件的密码，每个密码用逗号分隔。

例如：程序的输入：abcdEF12＃@,ccword12

程序的输出：abcdEF12＃@

**import re**

**lst = input('请输入密码（逗号隔开）：').split(',')**

**a = '[a-z]'**

**b = '[A-Z]'**

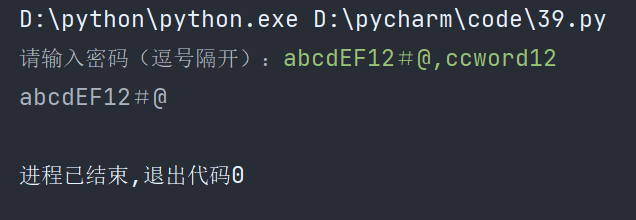
**c = '[0-9]'**

**d = '[$#@]'**

**for i in lst:**

**if re.findall(a, i) and re.findall(b, i) and re.findall(c, i) and re.findall(d, i) and 6 <= len(i) <= 12:**

**print(i)**



40. 编写程序，接受一系列逗号分隔的 4 位二进制数作为输入，检查它们是否可被 5 整除。 能被 5 整除的数字将以逗号分隔的顺序打印。 例如， 输入： 0100,0011,1010,1001

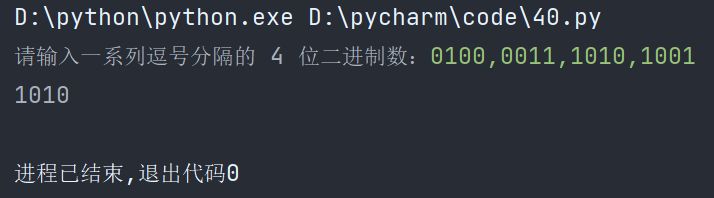
输出：1010

**lst = input('请输入一系列逗号分隔的 4 位二进制数：').split(',')**

**for item in lst:**

**if int(item, base=2) % 5 == 0:**

**print(item)**



41. 在一个坐标系中，绘出两个函数的图形，分别 y=sin(x)和 y=cos(x)，x 坐标取值范围为 (0, 2ￗ) 。

**import matplotlib.pyplot as plt**

**import numpy as np # 引入sin(x)和cos(x)**

**def plot\_sin\_cos():**

**sin\_x = np.sin(x)**

**cos\_x = np.cos(x)**

**plt.plot(x, sin\_x, color='red', linewidth=3, label='sin(x)')**

**plt.plot(x, cos\_x, color="black", linewidth=3, label='cos(x)')**

**plt.legend(loc='upper right')**

**def draw\_name():**

**plt.title('sin(x) cos(x)')**

**plt.xlabel('x')**

**plt.ylabel('y')**

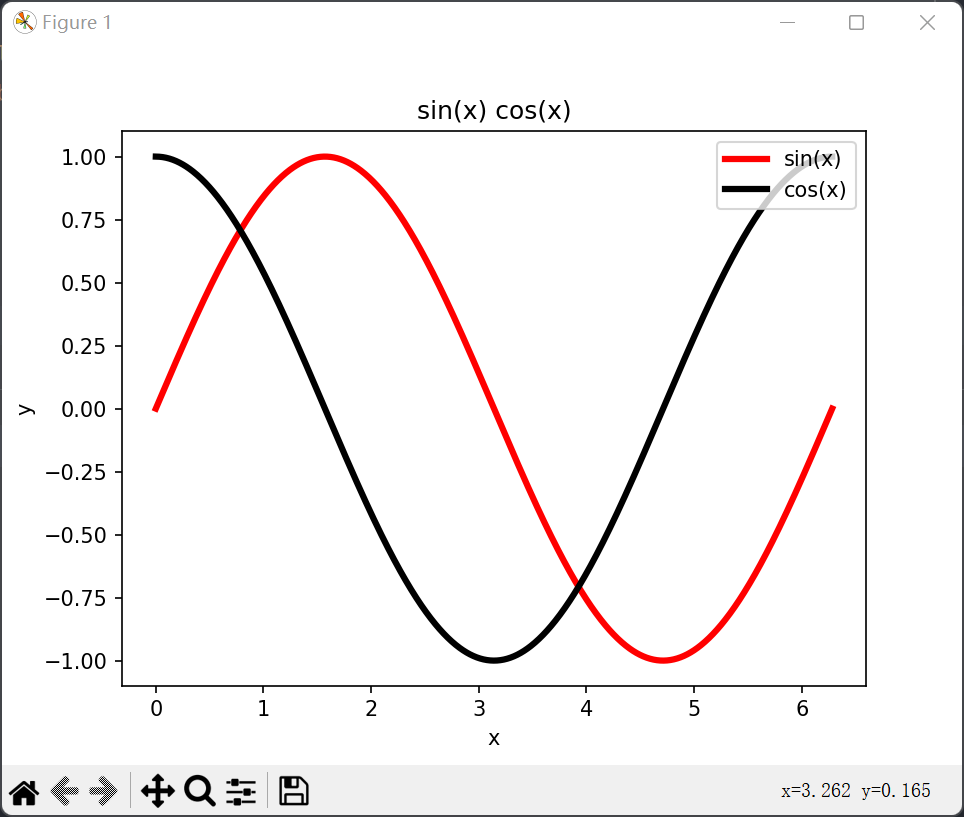
**if \_\_name\_\_ == '\_\_main\_\_':**

**x = np.linspace(0, 2 \* np.pi, 1000)**

**plot\_sin\_cos()**

**draw\_name()**

**plt.show()**



42. 编写多进程的程序，程序先输出主进程的 ID，然后创建一个子进程，子进程中输出自

己的 ID。(选做题)

**from multiprocessing import Process**

**import multiprocessing**

**import time**

**class My\_process(Process):**

**def \_\_init\_\_(self, name):**

**super().\_\_init\_\_()**

**self.name = name**

**def run(self):**

**print(self.name+'已经开始'+' id:'+str(self.pid))**

**time.sleep(3)**

**# for i in range(10000):**

**# print(i)**

**def test\_fun(name):**

**time.sleep(3)**

**print(f'我是{name}子进程')**

**if \_\_name\_\_ == '\_\_main\_\_':**

**curren\_time = time.time()**

**print('主程序id：', multiprocessing.current\_process().pid)**

**lst = []**

**p1 = My\_process('1进程')**

**p2 = My\_process('2进程')**

**p1.start()**

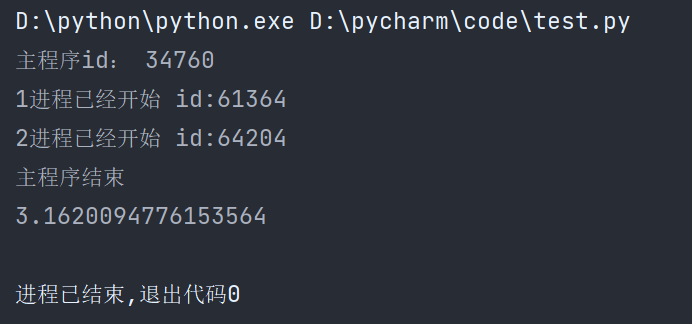
**p2.start()**

**p1.join()**

**p2.join()**

**print('主程序结束')**

**print(time.time()-curren\_time)**



**实验总结**

**1，巩固了python语法中最基础的方面，包括for循环，random随机数，while循环等基础的语法，对这些语法有了更加熟练的运用。**

**2，对于类的定义和内部成员的运用，是有很多技巧性的，尤其是PPT中的内容**

**3，匿名函数lambda与高级函数map(),filter(),max(),sorted()等连用很方便**

**4，写实验的过程中，遇到不会的知识点就回去看老师的ppt，边做题边复习也是很有效率的，很轻松的就记住了这些不熟悉的地方，所以学和做结合起来是很高效的。**

**5，对于一些我个人认为很复杂的代码，其实是会有很多的简便操作的，可以多在网上浏览一下，其实大神的代码，多学习他们的想法和算法，对自己的提升也是很显著的。**

**6，我拿38题举例，这道题我写了很久一开始的时候我就是想着用列表删除的方法来做的，其实最后写出来的代码行数并不多，但是个人认为对数学的思维要求很高。**

**7，以第10题举例，我看到题目的时候第一反映就是用正则表达式，自己也写了出来，差不多10行以内的代码，但是我在看ppt的时候凑巧看到了把单词全部小写的函数lower（），再对字符串中的apple进行计数，最后只用了2行代码就完成，我觉得是非常非常巧妙的！**

**8，因为我自己是转专业的同学，所以我需要补习C++的课程，在我自己学习的过程中，我很明显的感受到了python和c++的差别，python的能力实在是强大，代码很便捷，而且没有和c++一样需要头文件这样的繁琐。对比起来我就会一直选择python来进行以后的学习，对与那些大数据模型的训练，我以后就会选择python。**

**9，最重要的一点就是，实验里面的题目真的需要每个人都认认真真的做完，在我每一题都认认真真的做完了之后，觉得是很有帮助的，有些题目还是很费脑筋，但是都做出来之后，非常的有成就感**