



東南大學成賢學院

SOUTHEAST UNIVERSITY CHENGXIAN COLLEGE

Python 编程基础

实验报告

姓名： 邹惟一

学号： 10420629

成绩： _____

电子与计算机工程学院

School of Electronic & Computer Engineering

2022 年 10 月一实验二

【实验名称】

Python 基本数据类型练习

【实验目的】

- 1、编写程序，实现一个三位数的反序输出。从键盘上输入一个三位整数，对输入的整数进行处理和变换，输出这个三位数的反序数；
- 2、编写程序，实现月份数字向英文缩写的转换。从键盘上输入一个表示月份的数字（1~12），输出对应月份的英文缩写；
- 3、编写程序，实现从一段文本中提取电话号码及邮政编码。从键盘输入一段表示收件人信息的文本，利用 Python 正则表达式从文本中提取收件人的电话号码（固定电话或手机号）及邮政编码，并输出。

【实验内容】

1、程序清单

（1）反序输出

思路：python 没有直接反转 int 型整数的方法，而字符串有现成的方法，即 `reversed()`。注意：函数返回反向的迭代器对象；此方法并不会修改原来序列的元素顺序。

代码和注释：

```
num = input("请输入一个任意长度的整数")

newstr = ''.join(reversed(num)) # 将迭代器转为字符串
newnum = int(newstr) # 将字符串转为整数

print(f"这个整数的反序数为{newnum}") # 格式输出
```

（2）数字向英文缩写的转换

思路：采用列表存储英文月份，其索引对应数字月份。

代码：

```
list = ["Jan", "Feb", "Mar", "Apr", "May", "Jun",
        "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]

num = int(input("请输入数字月份 1~12: "))

print(f"{num}月对应的英文缩写是{list[num - 1]}")
```

(3) 提取电话号码及邮政编码

思路：固定电话号码和手机号码中很有可能含有邮编号码正则的正确匹配，这不是我们想要的，一个较好的解决办法是：在找到固定电话号码或手机号码后，使用字符串的 `replace` 方法去除它们，然后在检索邮政编码。

代码和注释：

```
import re

message = input()

# 固定电话号码或手机号码正则表达式
phone = "(0\d{2,3}-[1-9]\d{6,7}|1[3-9]\d{9})"

# 邮编正则表达式
post = "[0-8][0-7]\d{4}"

# 搜索电话号码或手机号码
res1 = re.search(phone, message)
if res1:
    print("找到的固定电话号码或手机号码为 %s" % res1.group())
    message = message.replace(res1.group(), "")
else:
    print("未找到固定电话号码或手机号码")

# 搜索邮政编码
res2 = re.search(post, message)
if res2:
    print("找到的邮编号码为 %s" % res2.group())
else:
    print("未找到邮编号码")
```

2、结果截图

(1) 反序输出

```
a1.py > ...  
1 num = input("请输入一个任意长度的整数")  
2  
3 newstr = ''.join(reversed(num)) # 将迭代器转为字符串  
4 newnum = int(newstr) # 将字符串转为整数  
5  
6 print(f"这个整数的反序数为{newnum}") # 格式输出  
7
```

问题 输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a1.py"  
请输入一个任意长度的整数87676876  
这个整数的反序数为67867678
```

(2) 数字向英文缩写的转换

```
1  
2 a1.py > ...  
1 list = ["Jan", "Feb", "Mar", "Apr", "May", "Jun",  
2         "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]  
3  
4 num = int(input("请输入数字月份1~12:"))  
5  
6 print(f"{num}月对应的英文缩写是{list[num - 1]}")  
7
```

问题 输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a1.py"  
请输入数字月份1~12: 12  
12月对应的英文缩写是Dec
```

(3) 提取电话号码及邮政编码

```
4
5 # 固定电话号码或手机号码正则表达式
6 phone = "(0\d{2,3}-[1-9]\d{6,7}|1[3-9]\d{9})"
7
8 # 邮编正则表达式
9 post = "[0-8][0-7]\d{4}"
10
11 # 搜索电话号码或手机号码
12 res1 = re.search(phone, message)
13 if res1:
14     print("找到的固定电话号码或手机号码为 %s" % res1.group())
15     message = message.replace(res1.group(), "")
16 else:
17     print("未找到固定电话号码或手机号码")
18
19 # 搜索邮政编码
20 res2 = re.search(post, message)
21 if res2:
22     print("找到的邮政编码为 %s" % res2.group())
23 else:
24     print("未找到邮政编码")
25
```

问题 输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a2.py"
成小贤, 江苏省南京市浦口区东大路6号, 025-58690736, 210088
找到的固定电话号码或手机号码为 025-58690736
找到的邮政编码为 210088
D:\vscode(python)>
```

【实验体会】

通过此次实验，第一题使我对字符串的使用更加熟悉了；第二题认识了列表存储数据的使用，列表是有序的，即放进去和取出来的顺序是一致的；第三题使我深入学习了正则表达式，会应用正则表达式在 python 字符串中寻找指定内容。`re.search()` 寻找整个字符串，寻找不成功返回 `none`，寻找成功返回第一个匹配成功的表达式，`re.match()` 尝试从字符串的起始位置匹配一个模式，匹配不成功返回 `none`，匹配成功返回一个匹配的对象。

2022 年 10 月一实验三

【实验名称】

Python 程序控制结构练习

【实验目的】

- 1、编写程序，从键盘输入一个 5 位数字，判断这个数字是不是回文数（设 n 是一任意自然数，如果 n 的各位数字反向排列所得自然数与 n 相等，则 n 被称为回文数）。
- 2、编写程序，根据用户输入的一个 18 位合法身份证号，输出用户的出生年月日、年龄和性别（第 7-10 位为出生年份，第 11-12 位为出生月份，第 13-14 位代表出生日期，第 17 位代表性别，奇数为男，偶数为女）。
- 3、编写程序，计算如下数列的值： $1-2+3-4+\cdots-966$ 。其中，所有数字为整数，从 1 开始递增，奇数为正，偶数为负。
- 4、在我国古代《算经》里有一个著名的不定方程问题：鸡翁一值钱五，鸡母一值钱三，鸡雏一值钱一。百钱买百鸡，问鸡翁、鸡母、鸡雏各几何？编写程序，求解此问题并输出解。

【实验内容】

1、程序清单

(1) 判断回文数

思路：python 没有直接反转 int 型整数的方法，而字符串有现成的方法，即 `reversed()`。注意：函数返回反向的迭代器对象；此方法并不会修改原来序列的元素顺序。

代码和注释：

```
num = input("请输入一个五位数字")

newstr = ''.join(reversed(num)) # 将迭代器转为字符串

# 两个字符串进行比较
if newstr == num:
    print("此数是回文数")
else:
    print("此数不是回文数")
```

(2) 输出出生年月日

思路：程序须根据身份证号计算年龄，为使程序更具一般性，考虑使用 `time`

类来获取当前的时间。记得将所有数字都转为整型，原因见注释。

代码和注释：

```
import time # 导入时间类

t = time.localtime() # 获取当前时间

id = input("请输入 18 位合法身份证号：")

# 此处转为整数的原因是和现在的年份相减
year = int(id[6] + id[7] + id[8] + id[9])
age = t.tm_year - year # 当前年份 - 出生年份

# 此处转为整数的原因是需要忽略开头的 0
m = int(id[10] + id[11]) # 出生月

# 此处转为整数的原因是需要忽略开头的 0
d = int(id[12] + id[13]) # 出生日

# 此处转为整数的原因是判断男女
sex = int(id[16])
if sex % 2:
    sex = "男"
else:
    sex = "女"

print(f"出生年月：{year}年{m}月{d}日")
print(f"年龄：{age}岁")
print(f"性别：{sex}")
```

(3) 计算数列

思路：方法一：使用 if 分支结构；方法二：使用 $(-1)^n$ 计算正负号。此处采用方法二。

代码和注释：

```
sum = 0

for i in range(1, 967):
    sum += (-1) ** (i - 1) * i # 采用  $(-1)^n$  计算正负号
print(sum)
```

(4) 不定方程

思路：列方程，利用二重循环求所有解：

设鸡翁 x 只、鸡母 y 只、鸡雏 z 只，

根据题意, $5x + 3y + z / 3 = 100$, $x + y + z = 100$, 两者联立, 得 $7x + 4y = 100$

代码和注释:

```
for x in range(0, 15):  
    for y in range(0, 26):  
        if (7 * x + 4 * y == 100):  
            print(f"鸡翁{x}只, 鸡母{y}只, 鸡雏{100 - x - y}只")
```

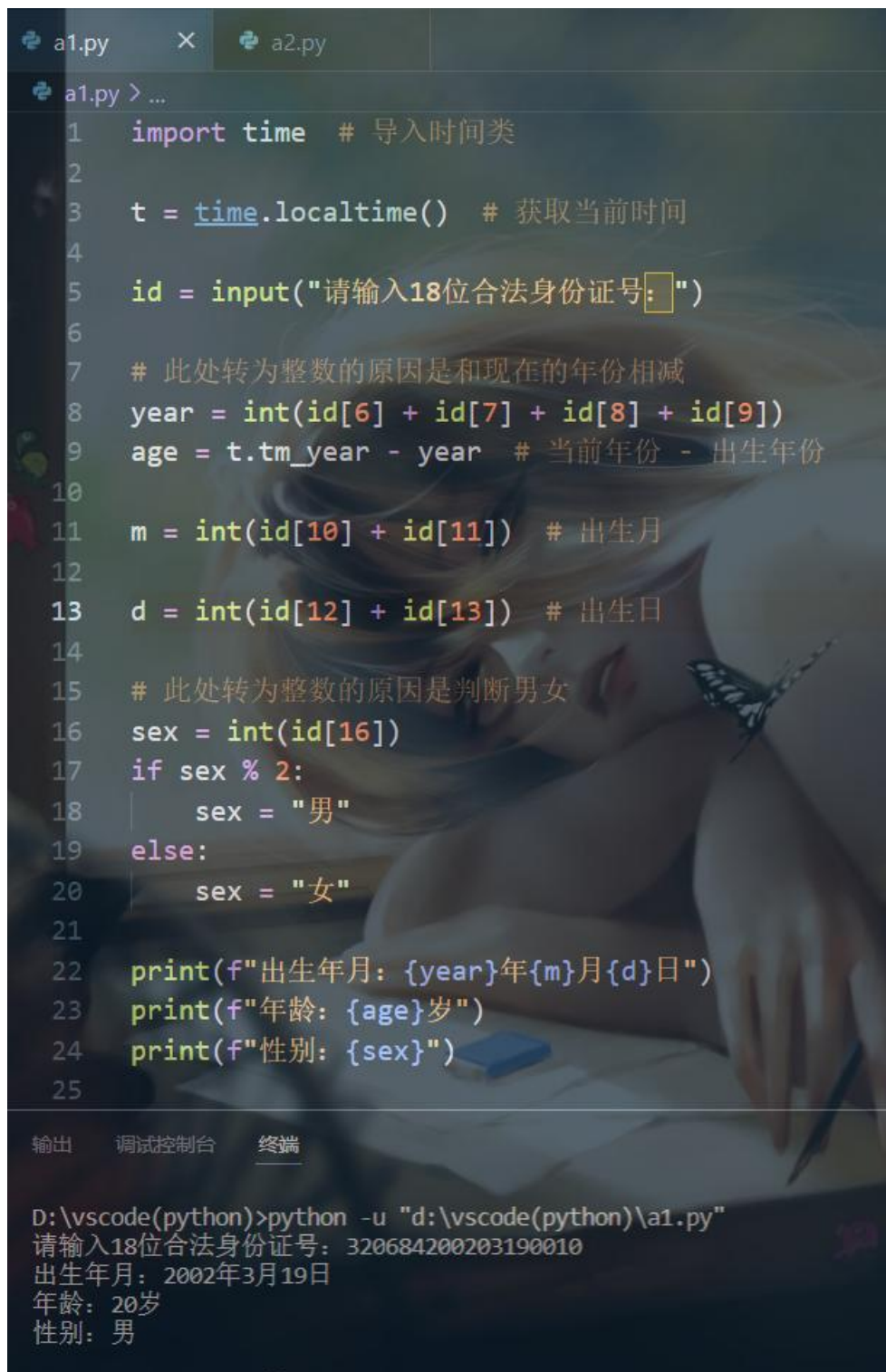
2、结果截图

(1) 判断回文数



```
a1.py  x  a2.py  
a1.py > ...  
1  num = input("请输入一个五位数字")  
2  
3  newstr = ''.join(reversed(num))  # 将迭代器转为字符串  
4  
5  # 两个字符串进行比较  
6  if newstr == num:  
7      print("此数是回文数")  
8  else:  
9      print("此数不是回文数")  
10  
输出  调试控制台  终端  
D:\vscode(python)>python -u "d:\vscode(python)\a1.py"  
请输入一个五位数字45654  
此数是回文数  
  
D:\vscode(python)>python -u "d:\vscode(python)\a1.py"  
请输入一个五位数字45678  
此数不是回文数  
  
D:\vscode(python)>|
```


(2) 输出出生年月日



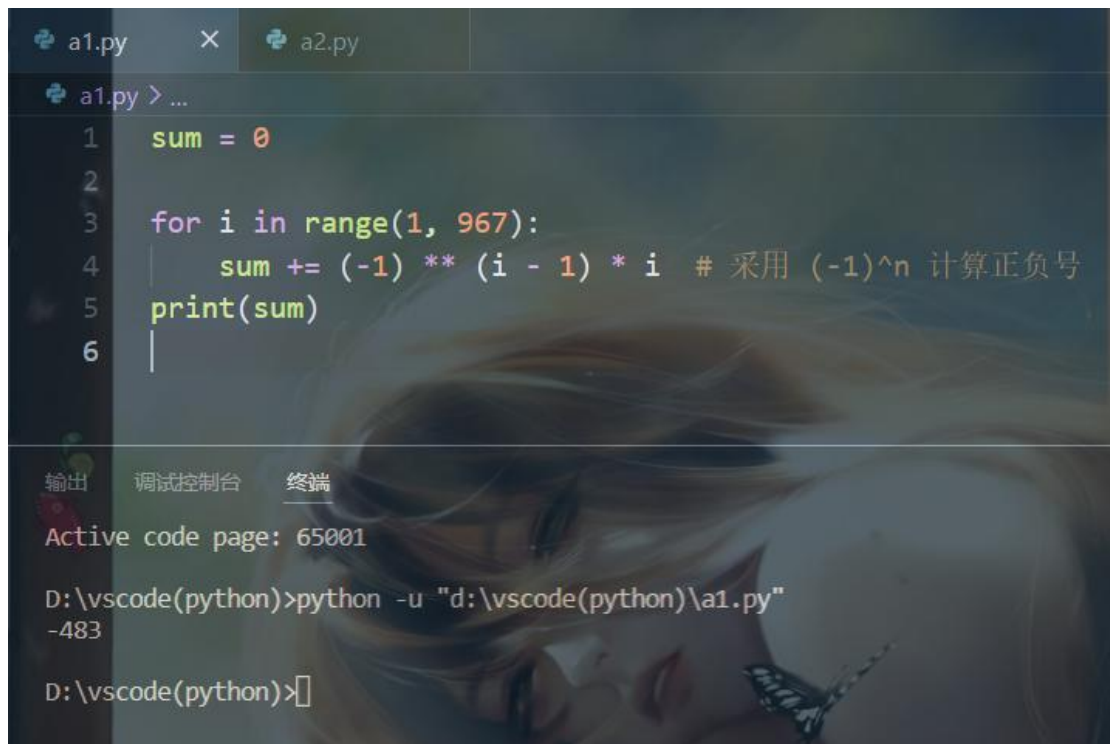
The image shows a VS Code editor window with two tabs: 'a1.py' and 'a2.py'. The 'a1.py' tab is active, displaying a Python script that calculates age and sex from an 18-digit ID card number. The script uses the `time` module to get the current local time. It prompts the user to enter an 18-digit ID card number. The script then extracts the year, month, and day from the ID card number and calculates the age by subtracting the birth year from the current year. It also determines the sex based on the 17th digit of the ID card number. Finally, it prints the birth date, age, and sex.

```
1 import time # 导入时间类
2
3 t = time.localtime() # 获取当前时间
4
5 id = input("请输入18位合法身份证号:")
6
7 # 此处转为整数的原因是和现在的年份相减
8 year = int(id[6] + id[7] + id[8] + id[9])
9 age = t.tm_year - year # 当前年份 - 出生年份
10
11 m = int(id[10] + id[11]) # 出生月
12
13 d = int(id[12] + id[13]) # 出生日
14
15 # 此处转为整数的原因是判断男女
16 sex = int(id[16])
17 if sex % 2:
18     sex = "男"
19 else:
20     sex = "女"
21
22 print(f"出生年月: {year}年{m}月{d}日")
23 print(f"年龄: {age}岁")
24 print(f"性别: {sex}")
25
```

The terminal output shows the execution of the script. The user enters the ID card number '320684200203190010'. The script outputs the birth date '2002年3月19日', the age '20岁', and the sex '男'.

```
D:\vscode(python)>python -u "d:\vscode(python)\a1.py"
请输入18位合法身份证号: 320684200203190010
出生年月: 2002年3月19日
年龄: 20岁
性别: 男
```

(3) 计算数列



```
a1.py  a2.py
a1.py > ...
1  sum = 0
2
3  for i in range(1, 967):
4      sum += (-1) ** (i - 1) * i # 采用 (-1)^n 计算正负号
5  print(sum)
6
```

输出 调试控制台 终端

Active code page: 65001

D:\vscode(python)>python -u "d:\vscode(python)\a1.py"

-483

D:\vscode(python)>

(4) 不定方程



```
a1.py  a2.py
a1.py > ...
1  for x in range(0, 15):
2      for y in range(0, 26):
3          if (7 * x + 4 * y == 100):
4              print(f"鸡翁{x}只, 鸡母{y}只, 鸡雏{100 - x - y}只")
5
```

输出 调试控制台 终端

D:\vscode(python)>python -u "d:\vscode(python)\a1.py"

鸡翁0只, 鸡母25只, 鸡雏75只
鸡翁4只, 鸡母18只, 鸡雏78只
鸡翁8只, 鸡母11只, 鸡雏81只
鸡翁12只, 鸡母4只, 鸡雏84只

D:\vscode(python)>

【实验体会】

第一题使我熟悉了对字符串的反转操作；第二题了解了 python 中 time 库的使用方法，并熟悉了对于字符串中单个字符的访问方式；第三题熟悉了循环和乘方的使用；第四题熟悉了二重循环的使用。

2022 年 11 月一实验四

【实验名称】

Python 列表与元组的应用练习

【实验目的】

- 1、编写程序，随机生成 10 个 100 以内的整数，随机数种子是 10，将这 10 个数添加到列表中。
- 2、编写程序，生成包含 20 个随机整数的元组，将前 10 个数按升序排列，后 10 个数按降序排列。
- 3、编写程序，从键盘输入一个列表，计算输出列表元素的平均值。
- 4、假设有三个列表：lst_who=['小马' , '小羊' , '小鹿'], lst_where=['草地上' , '电影院' , '家里'], lst_what=['看电影' , '听故事' , '吃晚饭']。试编写程序，随机生成三个 0-2 范围内的整数，将其作为索引分别访问三个列表中的对应元素，然后进行造句。例如，随机生成的三个整数分别为[1,0,2]，则输出句子”小羊在草地上吃晚饭”。

【实验内容】

1、程序清单

(1) 随机数放入列表

思路：python 使用随机数种子：random.seed(10)。

代码和注释：

```
import random

random.seed(10) # 设置随机数种子是 10

lista = []
for i in range(1, 11):
    lista.append(random.randint(0, 100))
    # 生成 10 个整数并添加到列表中
print(lista)
```

(2) 随机数放入元组并排序

思路：由于元组不可改变，我们首先采用列表存储 20 个整数，采用切片，分别对前 10 和后 10 个数排序，最后转为元组。

代码和注释：

```
import random
```

```

lista = []
for i in range(1, 21):
    lista.append(random.randint(0, 100))

pre = sorted(lista[0: 10]) # 截取列表中的前 10 个数并正向排序
post = sorted(lista[10: 20], reverse=True)
# 截取列表中的后 10 个数并逆向排序
tup1 = (pre + post) # 合并为元组

print(tup1)

```

(3) 计算列表元素的平均值

思路：依次输入 n 个数，采用序列的 sum() 方法计算总和。

代码和注释：

```

lista = []

n = int(input("请输入数字个数: "))
for i in range(0, n):
    lista.append(int(input(f"请输入第 {i + 1} 个数字: ")))

print(sum(lista) / n)

```

(4) 造句

思路：随机生成三个 0-2 范围内的整数，将其作为索引访问三个列表。

代码和注释：

```

import random

lst_who = ['小马', '小羊', '小鹿']
lst_where = ['草地上', '电影院', '家里']
lst_what = ['看电影', '听故事', '吃晚饭']

a = random.randint(0, 2)
b = random.randint(0, 2)
c = random.randint(0, 2)

print(f"{lst_who[a]}在{lst_where[b]}{lst_what[c]}")

```

2、结果截图

(1) 随机数放入列表

```
11.3(1).py > ...
1  import random
2
3  random.seed(10) # 设置随机数种子是 10
4
5  lista = []
6  for i in range(1, 11):
7      lista.append(random.randint(0, 100)) # 生成 10 个整数并添加到列表中
8  print(lista)
9
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\11.3(1).py"
[73, 4, 54, 61, 73, 1, 26, 59, 62, 35]

D:\vscode(python)>
```

(2) 随机数放入元组并排序

```
a1.py > ...
1  import random
2
3  lista = []
4  for i in range(1, 21):
5      lista.append(random.randint(0, 100))
6
7  pre = sorted(lista[0: 10]) # 截取列表中的前 10 个数并正向排序
8  post = sorted(lista[10: 20], reverse=True) # 截取列表中的后 10 个数并逆向排序
9  tup1 = (pre + post) # 合并为元组
10
11  print(tup1)
12
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a1.py"
[8, 18, 20, 35, 41, 43, 53, 62, 67, 78, 100, 85, 41, 39, 35, 31, 22, 21, 8, 2]

D:\vscode(python)>
```


(3) 计算列表元素的平均值

```
a2.py > ...
1  lista = []
2
3  n = int(input("请输入数字个数: "))
4  for i in range(0, n):
5      lista.append(int(input(f"请输入第 {i + 1} 个数字: ")))
6
7  print(sum(lista) / n)
8
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a2.py"
请输入数字个数: 5
请输入第 1 个数字: 1
请输入第 2 个数字: 2
请输入第 3 个数字: 3
请输入第 4 个数字: 4
请输入第 5 个数字: 5
3.0
```

(4) 造句

```
a3.py > ...
2
3  lst_who = ['小马', '小羊', '小鹿']
4  lst_where = ['草地上', '电影院', '家里']
5  lst_what = ['看电影', '听故事', '吃晚饭']
6
7  a = random.randint(0, 2)
8  b = random.randint(0, 2)
9  c = random.randint(0, 2)
10
11  print(f"{lst_who[a]}在{lst_where[b]}{lst_what[c]}")
12
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
小马在电影院看电影

D:\vscode(python)>s
```

【实验体会】

通过此次实验，使我更加熟悉了 python 中 random 库的使用，同时深入了解了内置方法 sorted 和 sort。熟悉了对于列表和元组的方法的使用。收获比较大。

2022 年 11 月一实验五

【实验名称】

Python 集合与字典的应用练习

【实验目的】

1、编写程序，获得用户输入的一个整数 N ，输出 N 中所出现的不同数字的和。
例如：用户输入 123123123，其中所出现的不同数字为 1、2、3，这几个数字的和为 6。

2、编写程序，使用字典保存用户姓名和对应密码，输出所有用户姓名，并找出某个用户的密码。

3、编写程序，输出字典 `fruits` 中键值最大的键值对。

```
fruits = { "apple" :10, "mango" :12, "durian" :20, "banana" :5}
```

4、假设字典 `dic_city` 中存放了每个人旅游过的城市，内容为{“张三凤”:[“北京”, “成都”], “李莱绸”:[“上海”, “广州”, “兰州”], “慕容福”:[“太原”, “济南”, “上海”, “西安”]}。试编写程序，实现以下功能。

统计每个人旅游过的城市的数目。输出结果如下：

张三凤去过 2 个城市

李莱绸去过 3 个城市

慕容福去过 4 个城市

统计去过上海的人数以及名单。输出结果如下：

去过上海的有 2 人，他们是李莱绸、慕容福

【实验内容】

1、程序清单

(1) 输出 N 中所出现的不同数字的和

思路：使用集合来存储 N 个数字，因为集合的特性为不含有重复元素；将输入的数字字符串放入集合后，集合存储的是 n 个单个字符串，所以要把每个字符转化为数字，再使用 `sum()` 方法求和。误区：在输入时就把字符串转为了数字，这时将 N 位的数字放入集合不会将 N 个数字分割为单个。

代码和注释：

```
s1 = set(input('请输入一个整数: ')) # 存储的是单个数字字符
s2 = set()
```

```
for i in s1:
```

```
s2.add(int(i)) # 将每个数字字符转化为单个数字

print(f"所出现的不同数字的和为{sum(s2)}")
```

(2) 使用字典保存用户姓名和对应密码

思路：字典的基本使用，键值对间使用英文冒号分隔。使用 `keys()`、`values()` 和 `items()` 方法访问字典的键、值、键值对

代码和注释：

```
dict1 = {"张三": 11, "李四": 22, "王五": 332}

print(dict1.keys()) # 获取字典的所有值

print(dict1['李四']) # 用键访问值
```

(3) 输出字典 `fruits` 中值最大的键值对

思路：我们可以使用 `values()` 方法获取所有值，并使用 `max` 获取值的最大值，但无法获取相应的键，考虑到列表 `list` 中有 `index()` 可确定索引，使用 `list()` 将原来的元组转为列表。

代码和注释：

```
fruits = {'apple': 10, 'mango': 12, 'durian': 20, 'banana': 5}

l1 = list(fruits.keys()) # 获取所有键
l2 = list(fruits.values()) # 获取所有值

i = l2.index(max(l2)) # 找出值最大的键值对的索引

print({l1[i]: l2[i]})
```

(4) 统计每个人旅游过的城市的数目

思路：使用 `items()` 遍历字典，使用 `len()` 方法获取列表的长度；遍历字典，如果在列表中找到“上海”，计数器加 1，并且将相应的人名加入新列表。

代码和注释：

```
dict1 = {'张三凤': ['北京', '成都'], '李莱绸': ['上海', '广州', '兰州'],
        '慕容福': ['太原', '济南', '上海', '西安']}

for i in dict1.items():
    print(f"{i[0]}去过{len(i[1])}个城市")
```

```
cnt = 0
list1 = []
for i in dict1.items():
    if "上海" in i[1]:
        cnt += 1
        list1.append(i[0])

print(f"去过上海的有{cnt}人，他们是", end="")
for i in list1:
    print(f"{i}", end=" ")
```

2、结果截图

(1) 输出 N 中所出现的不同数字的和



The screenshot shows a VS Code editor with a Python file named 'a3.py'. The code defines two sets, 's1' and 's2'. 's1' is created from the input '123123123'. 's2' is then populated with the unique digits from 's1' by converting each character to an integer and adding it to the set. Finally, the sum of the elements in 's2' is printed. The output window at the bottom shows the command 'python -u "d:\vscode(python)\a3.py"' being executed, followed by the input '123123123' and the output '所出现的不同数字的和为6'.

```
a3.py > ...
1  s1 = set(input('请输入一个整数: ')) # 存储的是单个数字字符
2  s2 = set()
3
4  for i in s1:
5      s2.add(int(i)) # 将每个数字字符转化为单个数字
6
7
8  print(f"所出现的不同数字的和为{sum(s2)}")
9
```

输出 调试控制台 终端

D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
请输入一个整数: 123123123
所出现的不同数字的和为6

(2) 使用字典保存用户姓名和对应密码

```
a2.py > ...
1 dict1 = {"张三": 11, "李四": 22, "王五": 332}
2
3 print(dict1.keys()) # 获取字典的所有值
4
5 print(dict1['李四']) # 用键访问值
6
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a2.py"
dict_keys(['张三', '李四', '王五'])
22
```

(3) 输出字典 **fruits** 中值最大的键值对

```
a1.py > ...
1
2 fruits = {'apple': 10, 'mango': 12, 'durian': 20, 'banana': 5}
3
4 l1 = list(fruits.keys())
5 l2 = list(fruits.values())
6
7 i = l2.index(max(l2))
8
9 print({l1[i]: l2[i]})
10
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a1.py"
{'durian': 20}
```

(4) 统计每个人旅游过的城市的数目

```
11.3(1).py > ...
1 dict1 = {'张三凤': ['北京', '成都'], '李茉绸': ['上海', '广州', '兰州'],
2         '慕容福': ['太原', '济南', '上海', '西安']}
3
4 for i in dict1.items():
5     print(f"{i[0]}去过{len(i[1])}个城市")
6
7 cnt = 0
8 list1 = []
9 for i in dict1.items():
10     if "上海" in i[1]:
11         cnt += 1
12         list1.append(i[0])
13
14 print(f"去过上海的有{cnt}人，他们是", end="")
15 for i in list1:
16     print(f"{i}", end=" ")
17
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\11.3(1).py"
张三凤去过2个城市
李茉绸去过3个城市
慕容福去过4个城市
去过上海的有2人，他们是李茉绸 慕容福
D:\vscode(python)>]
```

【实验体会】

通过此次实验，使我更加熟悉了 python 中的序列：集合、字典、列表各自的特性以及它们的公共方法和独有的方法，也掌握了各种序列间的关系，会将它们灵活转化，根据实际情况解题。

2022 年 11 月一实验六

【实验名称】

函数的应用练习

【实验目的】

1. 编写一个函数，输入不超过 5 位的正整数，输出该数为几位数，并逆序打印出各位数字。例如，输入 456，则输出（3， 654）。
2. 利用函数编写程序，生成 50 个随机数据，模拟一个班的考试成绩（要求分数在 40-100 之间），计算这批数据的平均分、最高分和最低分，并由高到低输出排序值。
3. 编写程序，以整数 17 为随机数种子，以用户输入的整数 N 为长度，产生 3 个长度为 N 位的密码，密码的每位是一个数字，每个密码单独一行输出。
4. 定义一个函数，函数参数为一个用户输入的小于 10000 的正整数，分解它的各位数字，并以一个元组的形式返回，在主程序中调用该函数。
5. 随机输入若干个不超过 2 位的正整数（输入-1 表示输入结束），找出其中所有同构数并排序输出。（正整数 n 若是它的平方数的尾部，则称 n 为同构数。如 5 的平方数是 25，且 5 是 25 的尾部，则 5 就是一个同构数。同理，25 的平方为 625，25 也是同构数。）

【实验内容】

1、程序清单

（1）逆序打印正整数

思路：对输入的正整数字符串应用 `reversed()` 方法完成逆序。注意：函数返回反向的迭代器对象；此方法生成新的字符串，并不会修改原来序列的元素顺序。

代码和注释：

```
def func(num):  
    newstr = ''.join(reversed(num))  
  
    return (len(newstr), int(newstr))  
  
num = input("请输入一个不超过 5 位的正整数：")  
print(func(num))
```

（2）计算数据的平均分、最高分和最低分

思路：调用列表的 `sort()` 函数对数据排序

代码和注释：

```
import random

def func():
    list1 = []
    for i in range(50):
        list1.append(random.randint(40, 100)) # 使用列表存储随机生成的数据
    list1.sort(reverse=True) # 对数据由高到低排序
    print(f"平均分: {sum(list1) / 50}") # 平均分
    print(f"最高: {list1[0]}")
    print(f"最低: {list1[49]}")
    print(f"原始数据为{list1}")

func()
```

(3) 产生 3 个长度为 N 位的密码

思路：假设输入整数 4，生成数的范围为 1000-9999，可推算对于 N 位整数的范围。

代码和注释：

```
import random

random.seed(17) # 种子为 17

N = int(input("请输入整数: "))

low = eval("1" + "0" * (N - 1)) # 下限
high = eval("1" + "0" * N) # 上限

for i in range(0, 3):
    print(random.randint(low, high - 1))
```

(4) 分解整数的各位数字

思路：使用 list() 方法将整数字符串转为由单个字符组成的列表，再遍历列表，将单个字符转为整数放入另一个列表中。

代码和注释：

```
def func(num):
    l1 = list(str(num))
    l2 = []
    for i in l1:
        l2.append(int(i))
    return tuple(l2)
```

```
num = int(input("请输入整数: "))
print(func(num))
```

(5) 找出同构数并排序输出

思路：输入的正整数是一位或两位数。对于一位数，比较末位和原数；对于两位数，比较末两位和原数。

代码和注释：

```
list1 = []
```

```
while True:
    n = int(input("请输入不超过 2 位的正整数，用-1 结束: "))
    if n == -1:
        break
    if (n < 10):
        temp = n * n
        if n == temp % 10:
            list1.append(n)
    else:
        temp = n * n
        first = temp % 10
        second = int(temp % 100 / 10)
        if second and n == eval(str(second) + str(first)):
            list1.append(n)

print(list1)
```

2、结果截图

(1) 逆序打印正整数



```
a3.py > ...
1 def func(num):
2     newstr = ''.join(reversed(num))
3
4     return (len(newstr), int(newstr))
5
6
7 num = input("请输入一个不超过 5 位的正整数: ")
8 print(func(num))
9

输出 调试控制台 终端
D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
请输入一个不超过 5 位的正整数: 23411
(5, 11432)
```


(2) 计算数据的平均分、最高分和最低分

```
1 import random
2
3
4 def func():
5     list1 = []
6     for i in range(50):
7         list1.append(random.randint(40, 100)) # 使用列表存储随机生成的数据
8     list1.sort(reverse=True) # 对数据由高到低排序
9     print(f"平均分: {sum(list1) / 50}") # 平均分
10    print(f"最高: {list1[0]}")
11    print(f"最低: {list1[49]}")
12    print(f"原始数据为{list1}")
13
14 |
15    func()
16
```

输出 调试控制台 终端

D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
平均分: 68.74
最高: 99
最低: 40
原始数据为[99, 97, 96, 95, 94, 94, 92, 89, 88, 88, 84, 84, 83, 82, 79, 77, 77, 76, 75, 74, 73, 73, 69, 68, 68, 67, 67, 65, 64, 64, 63, 60, 59, 57, 57, 56, 55, 55, 54, 53, 52, 52, 50, 49, 47, 47, 44, 44, 42, 40]

(3) 产生 3 个长度为 N 位的密码

```
1 import random
2
3 random.seed(17)
4
5 N = int(input("请输入整数: "))
6
7 low = eval("1" + "0" * (N - 1))
8 high = eval("1" + "0" * N)
9
10 for i in range(0, 3):
11     print(random.randint(low, high - 1))
12
```

输出 调试控制台 终端

D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
请输入整数: 5
78417
64285
49770

(4) 分解整数的各位数字

```
a3.py > ...
1  def func(num):
2      l1 = list(str(num))
3      l2 = []
4      for i in l1:
5          l2.append(int(i))
6      return tuple(l2)
7
8
9  num = int(input("请输入整数: "))
10 print(func(num))
11
```

输出 调试控制台 终端

D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
请输入整数: 28789
(2, 8, 7, 8, 9)

(5) 找出同构数并排序输出

```
a3.py > ...
1  list1 = []
2
3
4  while True:
5      n = int(input("请输入不超过 2 位的正整数, 用-1结束: "))
6      if n == -1:
7          break
8      if (n < 10):
9          temp = n * n
10         if n == temp % 10:
11             list1.append(n)
12     else:
13         temp = n * n
14         first = temp % 10
15         second = int(temp % 100 / 10)
16         if second and n == eval(str(second) + str(first)):
17             list1.append(n)
18
19  print(list1)
20
```

输出 调试控制台 终端

D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
请输入不超过 2 位的正整数, 用-1结束: 5
请输入不超过 2 位的正整数, 用-1结束: 25
请输入不超过 2 位的正整数, 用-1结束: 45
请输入不超过 2 位的正整数, 用-1结束: 3
请输入不超过 2 位的正整数, 用-1结束: -1
[5, 25]

【实验体会】

通过此次实验，使我更加熟悉了 `python` 中函数的各种调用，也巩固了对于集合、字典、列表各自的特性以及它们的公共方法和独有的方法，也回顾了随机数的使用，收获较大。

2022 年 12 月一实验七

【实验名称】

数据及文件操作练习

【实验目的】

1. 统计 hamlet.txt 中每个单词的出现频次，输出频次最高的 10 个单词。
 - 1) 保留介词、冠词、连词的情况下统计词频，并输出结果
 - 2) 去除介词、冠词、连词的情况下统计词频，并输出结果
2. 将整数 12345 分别写入文本文件 test.txt 和二进制文件 test.dat, 并比较两个文件的不同输出。

【实验内容】

1、程序清单

(1) 统计每个单词的出现频次

1) 保留介词、冠词、连词的情况下统计词频

```
import re

res = {}
# 只读打开文件
with open('./hamlet.txt', 'r') as f:
    txt = f.read()

for line in txt.splitlines():
    line = re.sub(r'[+=$#!]', ' ', line) # 去除所有标点符号
    for word in line.split():
        flag = False
        if word[-1] == '-':
            up = word[:-1]
            flag = True
            break
        if flag:
            word = up + word # 拼接末位单词
            flag = False
        res.setdefault(word.lower(), 0)
        res[word.lower()] += 1
values = sorted(res.values())

sortedres = sorted(res.items(), key=lambda d: d[1], reverse=True)
for i in range(0, 10):
    print(sortedres[i][0])
```

2) 去除介词、冠词、连词的情况下统计词频

```
import re

res = {}
# 只读打开文件
with open('./hamlet.txt', 'r') as f:
    txt = f.read()

for line in txt.splitlines():
    line = re.sub(r'[+=$#!]', ' ', line) # 去除所有标点符号
    for word in line.split():
        flag = False
        if word[-1] == '-':
            up = word[:-1]
            flag = True
            break
        if flag:
            word = up + word # 拼接末位单词
            flag = False
        res.setdefault(word.lower(), 0)
        res[word.lower()] += 1
values = sorted(res.values())

sortedres = sorted(res.items(), key=lambda d: d[1], reverse=True)

lista = ['the', 'a', 'an', 'at', 'on', 'behind', 'during', 'from',
        'into', 'and', 'but', 'or', 'so', 'however', 'although']
count = 0
for i in range(0, 20):
    if sortedres[i][0] not in lista:
        print(sortedres[i][0])
        count += 1
    if count == 10:
        break
```

(2) 将整数 12345 分别写入文本文件 test.txt 和二进制文件 test.dat

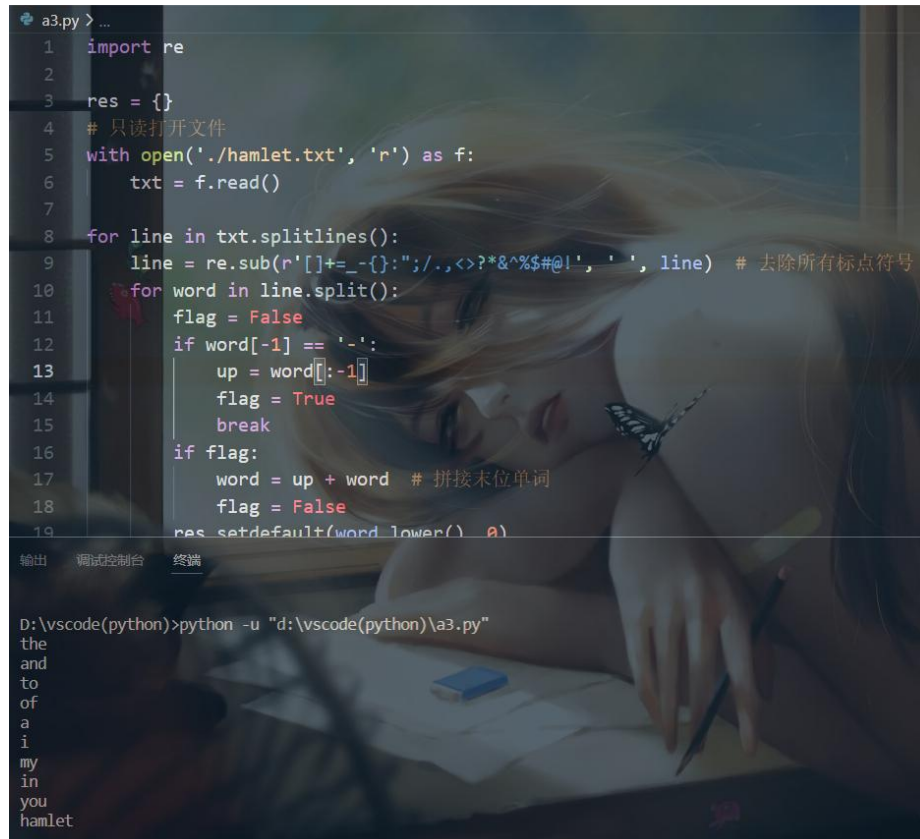
```
a = '12345'

with open('./test.txt', 'w') as f:
    f.write(a)

with open('./test.dat', 'w') as f:
    f.write(a)
```

```
with open('./test.dat', 'r') as f:
    print(f.read())
```

1) 保留介词、冠词、连词的情况下统计词频



```

a3.py > ...
1  import re
2
3  res = {}
4  # 只读打开文件
5  with open('./hamlet.txt', 'r') as f:
6      txt = f.read()
7
8  for line in txt.splitlines():
9      line = re.sub(r'[\s+=-{}:":;/.,<>?*&^%$#@!]', ' ', line) # 去除所有标点符号
10     for word in line.split():
11         flag = False
12         if word[-1] == '-':
13             up = word[:-1]
14             flag = True
15             break
16         if flag:
17             word = up + word # 拼接末位单词
18             flag = False
19     res.setdefault(word.lower(), 0)

```

输出 调试控制台 终端

```

D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
the
and
to
of
a
i
my
in
you
hamlet

```


2) 去除介词、冠词、连词的情况下统计词频

```
a3.py > ...
1 import re
2
3 res = {}
4 # 只读打开文件
5 with open('./hamlet.txt', 'r') as f:
6     txt = f.read()
7
8 for line in txt.splitlines():
9     line (variable) word: str ' ', line) # 去除所有标点符号
10    for word in line.split():
11        flag = False
12        if word[-1] == '-':
13            up = word[:-1]
14            flag = True
15            break
16        if flag:
17            word = up + word # 拼接末位单词
18            flag = False
19        res.setdefault(word.lower(), 0)
20        res[word.lower()] += 1
21 values = sorted(res.values())
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\a3.py"
to
of
i
my
in
hamlet
you
that
it
is
```

(2) 将整数 12345 分别写入文本文件 test.txt 和二进制文件 test.dat

```
1 a = '12345'
2
3 with open('./test.txt', 'w') as f:
4     f.write(a)
5
6 with open('./test.dat', 'w') as f:
7     f.write(a)
8
9 with open('./test.txt', 'r') as f:
10    print(f.read())
11
12 with open('./test.dat', 'r') as f:
13    print(f.read())
14
```

输出 调试控制台 终端

```
D:\vscode(python)>python -u "d:\vscode(python)\11.3(1).py"
12345
12345
```

【实验体会】

通过此次实验，使我更加熟悉了 `python` 中对于文件的操作和对于数据的处理，在数据的处理中，回顾了 `re` 库——正则表达式的使用，收获较大。