

```

30     elif player_life <= 0 and enemy_life > 0:
31         enemy_victory += 1
32         print('悲催，敌人把你干掉了！')
33     else:
34         print('哎呀，你和敌人同归于尽了！')
35     if player_victory > enemy_victory:
36         time.sleep(1)
37         print('\n【最终结果：你赢了！】')
38     elif enemy_victory > player_victory:
39         print('\n【最终结果：你输了！】')
40     else:
41         print('\n【最终结果：平局！】')
42     a1 = input('要继续游戏吗，请输入n退出，输入其他继续：')
43     if a1 == 'n':
44         break

```

```

1 #方法二
2 import time
3 import random
4
5 player_victory = 0
6 enemy_victory = 0
7 again = True
8 while again:
9     for i in range(1,4):
10         time.sleep(1.5)
11         print(' \n——现在是第 %s 局——' % i)
12         player_life = random.randint(100,150)
13         player_attack = random.randint(30,50)
14         enemy_life = random.randint(100,150)
15         enemy_attack = random.randint(30,50)
16
17         print('【玩家】\n血量: %s\n攻击: %s' % (player_life,player_attack))
18         print('-----')
19         time.sleep(1)
20         print('【敌人】\n血量: %s\n攻击: %s' % (enemy_life,enemy_attack))
21         print('-----')
22         time.sleep(1)
23
24         while player_life > 0 and enemy_life > 0:
25             player_life = player_life - enemy_attack
26             enemy_life = enemy_life - player_attack

```

```

27         print('你发起了攻击, 【玩家】剩余血量%s' % player_life)
28         print('敌人向你发起了攻击, 【敌人】的血量剩余%s' % enemy_life)
29         print('-----')
30         time.sleep(1.2)
31
32     if player_life > 0 and enemy_life <= 0:
33         player_victory += 1
34         print('敌人死翘翘了, 你赢了! ')
35     elif player_life <= 0 and enemy_life > 0:
36         enemy_victory += 1
37         print('悲催, 敌人把你干掉了! ')
38     else:
39         print('哎呀, 你和敌人同归于尽了! ')
40
41     if player_victory > enemy_victory :
42         time.sleep(1)
43         print('\n【最终结果: 你赢了! 】')
44     elif enemy_victory > player_victory:
45         print('\n【最终结果: 你输了! 】')
46     else:
47         print('\n【最终结果: 平局! 】')
48
49     a1 = input('要继续游戏吗, 请输入n退出, 输入其他继续: ') # 在 while True 循环中设置跳出条件。
50     if a1 == 'n':
51         again = False
52     else:
53         again = True

```

## 课后练习2：再来一种

- 题目解析：
  - 我们可以使用str.format()函数来取代之前学过的字符串格式化方法。
  - format()函数中的参数需要和之前字符串中的占位符{}匹配，当{}里没有数字，则按照默认的位置顺序匹配，如果{}中有数字，则按照数字去进行匹配。
- 参考代码：

```

1 import time
2 import random
3
4 player_victory = 0

```

```
5 enemy_victory = 0
6
7 while True:
8     for i in range(1, 4):
9         time.sleep(1.5)
10        print(' \n——现在是第 {} 局——'.format(i))
11        player_life = random.randint(100, 150)
12        player_attack = random.randint(30, 50)
13        enemy_life = random.randint(100, 150)
14        enemy_attack = random.randint(30, 50)
15
16        print(' 【玩家】 \n血量: {} \n攻击: {}'.format(player_life, player_attack))
17        print('-----')
18        time.sleep(1)
19        print(' 【敌人】 \n血量: {} \n攻击: {}'.format(enemy_life, enemy_attack))
20        print('-----')
21        time.sleep(1)
22
23        while player_life > 0 and enemy_life > 0:
24            player_life = player_life - enemy_attack
25            enemy_life = enemy_life - player_attack
26            print('敌人发起了攻击, 【玩家】剩余血量{}'.format(player_life))
27            print('你发起了攻击, 【敌人】的血量剩余{}'.format(enemy_life))
28            print('-----')
29            time.sleep(1.2)
30
31        if player_life > 0 and enemy_life <= 0:
32            player_victory += 1
33            print('敌人死翘翘了, 你赢了! ')
34        elif player_life <= 0 and enemy_life > 0:
35            enemy_victory += 1
36            print('悲催, 敌人把你干掉了! ')
37        else:
38            print('哎呀, 你和敌人同归于尽了! ')
39
40    if player_victory > enemy_victory :
41        time.sleep(1)
42        print('\n【最终结果: 你赢了! 】 ')
43    elif enemy_victory > player_victory:
44        print('\n【最终结果: 你输了! 】 ')
45    else:
46        print('\n【最终结果: 平局! 】 ')
```

```
47
48     a1 = input('要继续游戏吗, 请输入n退出, 输入其他继续: ')
49     if a1 == 'n':
50         break
```

## 第8关 编程思维：如何解决问题

### 课后练习1：老师我帮你

- 题目解析：
  - 合并列表，可用append()函数或者extend()函数将两个列表的合并。
  - 列表排序，可用sort()函数或者sorted()函数将列表的内容排序。
- 参考代码：

```
1 # 合并列表
2 # 使用append()函数
3 list1 = [91, 95, 97, 99]
4 list2 = [92, 93, 96, 98]
5 for i in list2:
6     list1.append(i)
7 print(list1)
8
9 # 使用extend()函数
10 list1 = [91, 95, 97, 99]
11 list2 = [92, 93, 96, 98]
12 list3 = list1
13 list3.extend(list2)
14 print(list3)
```

### 课后练习2：老师我又来了

- 题目解析1：
  - 使用for循环叠加成绩，然后除以总人数或直接使用numpy模块中的mean()函数，求出平均值。
  - 由于学生的成绩已经被集中到一个列表里，所以可以用for循环遍历这个列表来取出成绩。取出的成绩与平均值进行判断，将小于平均值的成绩存放到新的列表里。
- 参考代码：

```

1 # 方法一：通过for循环求出平均值并判断
2 scores1 = [91, 95, 97, 99, 92, 93, 96, 98]
3 sum = 0
4 scores2 = []
5
6 for score in scores1:
7     sum = sum + score # 叠加成绩
8     average = sum/len(scores1) # 用len()函数计算出总人数，即列表的元素数量
9 print('平均成绩是: {}'.format(average))
10
11 for score in scores1:
12     if score < average: # 判断是否小于平均值
13         scores2.append(score)
14     continue
15 print('低于平均成绩的有: {}'.format(scores2))

```

```

1 # 方法二，通过mean()函数，直接求出平均值
2 import numpy as np
3
4 scores1 = [91, 95, 97, 99, 92, 93, 96, 98]
5 scores2 = []
6
7 average = np.mean(scores1) # 一行解决。
8 print('平均成绩是: {}'.format(average))
9
10 for score in scores1:
11     if score < average:
12         scores2.append(score)
13     continue
14 print('低于平均成绩的有: {}'.format(scores2))

```

## 第9关 函数

### 课后练习1：年终奖有多少

- 题目解析：
  - 定义两个函数，一个函数的功能是计算奖金，另一个函数的功能是打印员工的姓名，入职时长和所获奖金的

相关信息。

- 使用input()函数，输入员工姓名与入职时长，调用计算奖金的函数，计算出对应的奖金金额。
  - 将奖金金额传递到打印功能的函数，将相关信息用print()函数打印出来。
- 参考代码：

```
1 # 计算奖金金额
2 def bonus(month):
3     if month < 6:
4         money = 500
5         return money
6     elif 6 <= month <= 12:
7         money = 120 * month
8         return money
9     else:
10        money = 180 * month
11        return money
12
13 # 打印相关信息
14 def info(name, month):
15     gain = bonus(month) # 调用bonus()函数，并将奖金金额存放到gain变量
16     print('%s来了%s个月，获得奖金%s元' % (name, month, gain))
17
18 info('大聪', 14)
```

## 课后练习2：hellokikty抽奖器

- 题目解析：
  - 通过随机抽取luckylist列表中的元素，来作为中奖的人员，所以luckylist列表中的元素，为可变变量，可以作为函数的参数名。
  - 将抽奖代码封装成函数，将luckylist列表中的元素作为函数的参数名。
  - 代码封装完函数后，调用函数，给参数名进行赋值。

- 参考代码：

```
1 import random
2 import time
3
4 # 将抽奖程序封装成函数
5 def choujiang(q,w,e): # 定义一个抽奖函数，带有3个参数，也就是3位候选人
6     luckylist = [q,w,e] # 定义一个中奖名单的列表
7     a = random.choice(luckylist) # 在中奖名单里面随机选择
```

```

8     print('开奖倒计时', 3)
9     time.sleep(1)
10    print('开奖倒计时', 2)
11    time.sleep(1)
12    print('开奖倒计时', 1)
13    time.sleep(1)
14    image = '''
15    /\_)o<
16    |       \\
17    | o . o|
18    \____/
19    ...
20
21    print(image)
22    print('恭喜' + a + '中奖! ')
23 choujiang('虚竹', '萧峰', '段誉')  # 调用函数，给参数名q,w,e赋值

```

## 第10关 工作计量器

### 课后练习1：剪刀石头布

- 题目解析：
  - 使用random.choice()函数，实现电脑随机亮券；使用input()函数，实现用户亮券。
  - 使用while循环，判断用户输入的是否是石头、剪刀或布，若不是，则重新输入。
  - 通过穷举赢的情况，使用if...elif...else语句判断出哪方胜出。
- 参考代码：

```

1 punches = ['石头', '剪刀', '布']
2 computer_choice = random.choice(punches) # 电脑亮券
3 user_choice = input('请出拳：(石头、剪刀、布) ') # 请用户亮券
4 while user_choice not in punches: # 当用户输入错误，提示错误，重新输入
5     print('输入有误，请重新出拳')
6     user_choice = input() # 用户重新输入
7
8 # 双方亮拳信息
9 print('——战斗过程——')
10 print('电脑出了: %s' %(computer_choice))
11 print('你出了: %s' %(user_choice))

```

```
12
13 # 胜负结果
14 print('——结果——')
15 if user_choice == computer_choice:
16     print('平局! ')
17 elif (user_choice == '石头' and computer_choice == '剪刀') or (user_choice == '剪刀' and
18     computer_choice == '布') or (user_choice == '布' and computer_choice == '石头'):
19     print('你赢了! ')
20 else:
21     print('你输了! ')
```

## 课后练习2：让代码更简洁

- 题目解析：
  - 使用index()函数，优化用户赢的判断条件。
  - 可以先确定电脑随机选到的选项在列表punches中的索引位置，将其和玩家的选项联系起来。（如电脑出布，那么玩家的选项要出剪刀才能赢，通过元素的索引位置，观察规律）
  - 用户必赢规律：当用户亮券所对应的位置索引值是电脑所亮券所对应的位置索引值-1时，用户必赢。
- 参考代码：

```
1 import random
2 punches = ['石头', '剪刀', '布']
3 computer_choice = random.choice(punches)
4 user_choice = input('请出拳：(石头、剪刀、布) ')
5 while user_choice not in punches:
6     print('输入有误，请重新出拳')
7     user_choice = input()
8
9 print('——战斗过程——')
10 print('电脑出了: %s' % computer_choice)
11 print('你出了: %s' % user_choice)
12
13 print('——结果——')
14 if user_choice == computer_choice:
15     print('平局! ')
16 elif user_choice == punches[punches.index(computer_choice)-1]: # 判断电脑位置索引值-1的元素
17     print('你赢了! ')
18 else:
19     print('你输了! ')
```

## 第11关 编程思维：如何debug

### 课后练习1：一起来抓虫

- 题目解析：
  - 代码一：主要考察全局变量与局部变量的作用域，属于知识点debug。
  - 代码二：主要考察代码的运行逻辑，属于思维不清debug。
  - 代码三：主要考察代码的特殊性，属于被动掉坑debug。

#### ● 参考代码：

```
1 # 代码一
2 scores = {'语文':89, '数学':95, '英语':80}
3 def get_average(scores):
4     sum_score = 0 # sum_score 作为函数内部的局部变量，从而可以为函数所用。
5     for subject, score in scores.items():
6         sum_score += score
7         print('现在的总分是%d'%sum_score)
8     ave_score = sum_score/len(scores)
9     print('平均分是%d'%ave_score)
10 get_average(scores)
```

```
1 # 代码二
2 not_bad_word = True
3 while not_bad_word:
4     x = input('请给旺财取个外号：')
5     if x == '小狗' or x == '汪汪': # 只要外号是两个中的一个，就会生气。
6         not_bad_word = False
7         print('我生气了，不想理你了！')
8
9 print('对不起，以后我不会这么叫你了')
```

```
1 # 代码三
2 deposit = [100,300,900,2000,5000,0,2000,4500]
3 for i in range(1, len(deposit)):
4     if deposit[i-1] == 0: # 判断除数等于0时，特殊处理。
5         print('你上次存款为 0 哟！')
```

```
6     else:  
7         times = deposit[i]/deposit[i-1]  
8         print('你的存款涨了%f倍' %times)
```

## 课后练习2：贴心的除法计算器

- 题目解析：
  - 使用input()函数，输入被除数，除数，并将数据类型转换为浮点型。
  - 通过try...except语句，排除掉输入了非数值（即不属于整数和浮点数）或被除数为零的情况。
- 参考代码：

```
1  print('\n欢迎使用除法计算器! \n')  
2  while True:  
3      try:  
4          x = input('请你输入被除数: ')  
5          y = input('请你输入除数: ')  
6          z = float(x)/float(y)  
7      except ZeroDivisionError: # 排除掉除数不为0的情况  
8          print('0是不能做除数的! ')  
9      except ValueError: # 排除掉非数值的情况  
10         print('除数和被除数都应该是整值或浮点数! ')  
11     else:  
12         print(x, '/', y, '=', z)  
13         break
```

## 第12关 类与对象1

### 课后练习1：一次性说完

- 题目解析：一次性“说完”居住地和出生地。
  - 新建一个方法，在这个方法中调用实例方法born和live，调用实例方法时需要加上self，即 self.方法名()。
  - 创建完之后，需要在类外创建一个实例，然后调用新建的方法。创建实例之后self就指向新建的实例，所以可以使用实例名.方法名()的方式进行调用。
- 参考代码：

```
1  class Chinese:
```

```

2     def __init__(self,hometown,region):
3         self.hometown = hometown
4         self.region = region
5         print('程序持续更新中.....')
6
7     def born(self):
8         print('我生在%s。'%self.hometown)
9
10    def live(self):
11        print('我在%s。'%self.region)
12
13    # 新建方法，调用上面的两个方法（注：方法名可自定义）。
14    def citys(self):
15        self.born()
16        self.live()
17
18 wufeng = Chinese('广东', '深圳')
19 wufeng.citys()  # 调用新建实例方法

```

## 课后练习2：重要的事情说三遍

- 题目解析：
  - 第一题：打招呼
    - 先创建一个Robot类，然后定义一个初始化方法。由于需要输入名字和称呼，所以需要使用input()语句来获取这两个值。
    - 然后要使得代码能够运行，需要进行实例化调用。
  - 第二题：打印三遍愿望
    - 定义一个名为say\_wish()的实例方法之后，需要获取愿望，故要使用功能input()函数获取愿望。
    - 然后将愿望打印三遍，打印多遍用循环。
    - 最后还是实例化调用类，创建类对象后调用实例方法say\_wish()。
- 参考代码：

```

1  # 第一题代码
2  class Robot:
3      def __init__(self):
4          self.name = input('我还没有名字，请给我起个名字吧：')
5          self.user = input('你叫什么名字：')
6          print('你好%s, 我叫%s, 很高兴认识你!' % (self.user, self.name))
7  robot1 = Robot()

```

```

8
9  # 第二题代码
10 class Robot:
11     def __init__(self):
12         self.name = input('我还没有名字, 请给我起个名字吧: ')
13         self.user = input('你叫什么名字: ')
14         print('你好%s, 我叫%s, 很高兴认识你!' %(self.user, self.name))
15     def say_wish(self):
16         wish = input('告诉我一个你的愿望吧: ')
17         print('%s的愿望是: %s' %(self.user, wish))
18         for i in range(3):
19             print(wish)
20
21 robot2 = Robot()
22 robot2.say_wish()

```

## 第13关 类与对象2

### 课后练习1：老师和父亲

- 题目解析：
  - 创建两个子类，同时继承已有的两个类，一个是(Teacher ,Father )，另一个是(Father ,Teacher )。
  - 在其中任选一个子类进行定制：将 face 属性的值改变为'gentle'。
  - 最后创建实例time3 和time4 分别调用两个子类，再用实例对象调用face 属性并打印。
- 参考代码：

```

1  # 第一题代码
2  class Teacher:
3      face = 'serious'
4      job = 'teacher'
5
6  class Father:
7      face = 'sweet'
8      parenthood = 'dad'
9
10 class TeacherMore(Teacher, Father):
11     pass
12
13 # 定义一个新的类或函数，如果其内部无须其他代码，也必须加上pass使得定义成立。

```

```
14 class FatherMore(Father, Teacher):  
15     face = 'gentle'  
16  
17 time3 = TeacherMore()  
18 time4 = FatherMore()  
19 print(time3.face)  
20 print(time4.face)
```

## 课后练习2：学习时间记录

- 题目解析：
  - 如果编程开发人员学 Python 的话，学习效率默认为1。则在继承之后，给rate 添加一个默认参数即可。
  - job 的属性为 programmer，即给job 添加要给默认参数。
  - 因为学习效率不同的人员可能不同，可以通过外部传值进行修改，所以默认参数添加到子类对应的方法中，而工作是固定的都是programmer，所以在调用父类的初始函数的时候，给job 传入一个参数，这样子就不会受到外部传入的参数的影响。
- 参考代码：

```
1 class Student:  
2     def __init__(self, name, job=None, time=0.00, time_effective=0.00):  
3         self.name = name  
4         self.job = job  
5         self.time = time  
6         self.time_effective = time_effective  
7  
8     def count_time(self, hour, rate):  
9         self.time += hour # 在类外进行多次调用时，会累计时间  
10        self.time_effective += hour * rate  
11  
12 class Programmer(Student):  
13     def __init__(self, name): # 因为job、time、time_effective都有了默认值，所以这里可以不用  
14         # 写  
15         Student.__init__(self, name, job='programmer', time=0.00, time_effective=0.00)  
16  
17     def count_time(self, hour, rate=1):  
18         Student.count_time(self, hour, rate)  
19         # 这里也是为了简写代码而直接调用类Student中的方法count_time。  
20  
21 student1 = Student('韩梅梅')  
22 student2 = Programmer('李雷')
```

```
22 print(student1.job)
23 print(student2.job)
24 student1.count_time(10, 0.8)
25 student2.count_time(10)
26 print(student1.time_effective)
27 print(student2.time_effective)
```

## 第14关 流浪图书馆

### 课后练习1：书的分类

- 题目解析：
  - 创建一个子类FictionBook，继承父类Book。
  - 在子类改造父类的初始化方法，让程序能够打印出实例book的相关信息。由于初始函数不能使用return 返回字符串，而要打印相关的信息，所以需要加入本课的新知识点\_\_str\_\_()方法对信息以字符串形式进行返回。
- 参考代码：

```
1 class Book:
2     def __init__(self, name, author, comment, state = 0):
3         self.name = name
4         self.author = author
5         self.comment = comment
6         self.state = state
7
8 # 创建一个Book类的子类 FictionBook
9 class FictionBook(Book):
10    def __init__(self, name, author, comment, state = 0, type = '虚构类'):
11        Book.__init__(self, name, author, comment, state = 0)
12        self.type = type
13 # 这里是类的继承和改写，在初始化方法中增加了默认参数type = '虚构类'，而为了简写代码，直接调用Book类中的初始化方法，因为Book类中初始化方法本身不含参数type，故增加15行代码self.type = type。
14    def __str__(self):
15        status = '未借出'
16        if self.state == 1:
17            status = '已借出'
18        return '类型: %s 名称: 《%s》 作者: %s 推荐语: %s\n状态: %s' % (self.type,
19                           self.name, self.author, self.comment, status)
```

```
20 book = FictionBook('囚鸟', '冯内古特', '我们都是受困于时代的囚鸟')
21 print(book)
```

## 课后练习2：我想看TA的书

- 题目解析：
  - 要求补全show\_author\_book()方法，先用条件判断语句判断该作者在不在列表authors里，需要使用if 条件句，直接判断输入的作者是否在作者列表中。
    - 如果不在就打印“很可惜，我们暂时没有收录这位作者的作品”。
    - 如果在，则遍历列表books的每个实例，当实例属性author与输入的作者名相等，就打印该实例。

- 参考代码：

```
1 class Book:
2     def __init__(self, name, author, comment, state = 0):
3         self.name = name
4         self.author = author
5         self.comment = comment
6         self.state = state
7     def __str__(self):
8         status = '未借出'
9         if self.state == 1:
10             status = '已借出'
11     #这里表示当state=1时，书籍状态为"已借出"
12     return '名称:《%s》作者:%s 推荐语: %s\n状态: %s' % (self.name, self.author,
13     self.comment, status)
14
15 class BookManager:
16     authors = []
17     def __init__(self):
18         book1 = Book('撒哈拉的故事', '三毛', '我每想你一次，天上便落下一粒沙，从此便有了撒哈拉。')
19         book2 = Book('梦里花落知多少', '三毛', '人人都曾拥有荷西，虽然他终会离去。')
20         book3 = Book('月亮与六便士', '毛姆', '满地都是六便士，他却抬头看见了月亮。')
21     #这里是实例化Book类并赋值给book1、book2、book3
22     self.books = [book1, book2, book3]
23     #将三个实例化类装进books列表中
24     self.authors.append(book1.author)
25     self.authors.append(book2.author)
26     self.authors.append(book3.author)
27     #将每个实例中的author装进authors列表
```

```

27
28     def menu(self):
29         while True:
30             print('1.查询书籍')
31             choice = int(input('请输入数字选择对应的功能: '))
32             if choice == 1:
33                 self.show_author_book()
34             else:
35                 print('感谢使用!')
36                 break
37
38     def show_author_book(self):
39         author = input('请输入想查询作家的名称: ')
40         if author in self.authors:
41             print(author + '的作品有: ')
42             for book in self.books:
43 #从books列表中遍历出每一个实例类
44                 if book.author == author:
45 #book.author表示调用实例的author属性
46                     print(book)
47             else:
48                 print('很可惜, 我们暂时没有收录这位作者的作品')
49
50 manager = BookManager()
51 manager.menu()

```

## 第15关 编码和文件读写课后

### 课后练习1：数据转移中的变化

- 题目解析：
  - 编码与解码：使用encode()函数和decode()函数可以实现编码与解码。
  - 通过文件读写复制图片：可以通过open()函数使用rb权限来打开一个图片文件，并把文件赋值给一个变量。再用open()函数使用wb权限打开一个新的图片文件，并把之前的变量写入，这样就完成了复制图片的操作。
- 参考代码：

```

1  # 1. 分别使用gbk和utf-8编码自己的名字，并打印。（以吴枫为例）
2  print('吴枫'.encode('gbk'))

```

```
3 print('吴枫'.encode('utf-8'))
4 # 结果为:
5 # b'\xe4\xce\xe2\xb7\xe3'
6 # b'\xe4\xec\x90\xb4\xe6\x9e\xab'
7
8 # 2. 复制上一步得到的结果, 进行解码, 分别打印。
9 print(b'\xe4\xce\xe2\xb7\xe3'.decode('gbk'))
10 print(b'\xe4\xec\x90\xb4\xe6\x9e\xab'.decode('utf-8'))
```

```
1 #练习2
2 with open('photo2.png', 'rb') as file: # 以"rb"模式打开图片
3     data = file.read()
4     with open('photo3.png', 'wb') as newfile: # 以"wb"模式写入
5         newfile.write(data)
```

```
1 #练习3-1
2 file1 = open('scores.txt', 'r', encoding='utf-8')
3 file_lines = file1.readlines()
4 file1.close()
5
6 final_scores = []
7 for i in file_lines:
8     data = i.split()
9     sum = 0 # 先把总成绩设为0
10    for score in data[1:]: # 遍历列表中第1个数据和之后的数据
11        sum = sum + int(score) # 然后依次加起来, 但分数是字符串, 所以要转换
12    result = data[0]+str(sum)+'\n' # 结果就是学生姓名和总分
13    print(result)
14    final_scores.append(result)
15 print(final_scores)
16
17 sum1 = open('winner.txt', 'w', encoding='utf-8')
18 sum1.writelines(final_scores)
19 sum1.close()
```

```
1 #练习3-2
2 file1 = open('winner.txt', 'r', encoding='utf-8')
3 file_lines = file1.readlines()
4 file1.close()
5
```

```

6  dict_scores = {}
7  list_scores = []
8  final_scores = []
9
10 for i in file_lines: # i是字符串。
11     print(i)
12     name = i[:-4] # 取出名字 (注：字符串和列表一样，是通过偏移量来获取内部数据。)
13     score = int(i[-4:-1]) # 取出成绩
14     dict_scores[score] = name # 将名字和成绩对应存为字典的键值对 (注意：这里的成绩是键)
15     list_scores.append(score)
16
17 list_scores.sort(reverse=True) # reverse，逆行，所以这时列表降序排列，分数从高到低。
18
19 for i in list_scores:
20     result = dict_scores[i] + str(i) + '\n'
21     final_scores.append(result)
22
23 print(final_scores) # 最终结果
24
25 winner_new = open('winner_new.txt', 'w', encoding='utf-8')
26 winner_new.writelines(final_scores)
27 winner_new.close()

```

## 课后练习2：古诗默写

- 题目解析：
  - 创建一个列表来存放需要默写的内容，列表中的每一个元素即是需要默写的诗句。
  - 使用open()函数打开诗词文件后，使用readlines()函数读取内容，得到一个列表。列表中的每一个元素是诗句的每一行内容。
  - 使用open()函数打开诗词文件，并用循环来比对原始诗词和我们需要默写的内容，如果一致就把原始诗词中相应的诗句替换成下划线，如果不一致则原封不动。

- 参考代码：

```

1 list_test = ['一弦一柱思华年。\\n', '只是当时已惘然。\\n'] # 将要默写的诗句放在列表里。
2 with open ('poem2.txt', 'r') as f:
3     lines = f.readlines()
4     print(lines)
5 with open('poem2.txt', 'w') as new:
6     for line in lines:
7         if line in list_test: # 属于默写列表中的句子，将其替换成横线。

```

```
8         new.write('_____。 \n')
9     else:
10        new.write(line)
```

## 第16关 模块

### 课后练习1：时间管理器

- 题目解析：
  - 设定一个无限循环，方便任务管理器的程序可以被反复执行。
  - 设定一个目标任务时间并赋值给变量，使用time.time()方法记录任务开始的时间。
  - 使用for ... in range():的程序结构来倒计时任务，range()函数中设置步长为-1，即可起到倒计时的效果。
  - 任务结束后继续使用time.time()函数记录任务结束的时间，并用任务结束的时间减去任务开始的时间，将结果记为实际任务时间，将实际任务时间记入txt文档中。
- 参考代码：

```
1 import time
2
3 input("欢迎使用\"时间管理器\"！请按回车继续。")
4 while True:
5     task_name = input('请输入任务名：')
6     task_time = int(input('觉得自己至少可以专注这个任务多少分钟？输入 N 分钟'))
7     input('此次任务信息：\n我要完成的任务：%s\n我至少要专注：%d分钟\n按回车开始计时：' %
8           (task_name, task_time))
9     start = time.time() # 开始计时
10    start_time = time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(time.time())) # 格式化日期
11    # for t in range(task_time*60, 0, -1): # 实际代码：分钟转成秒要乘60，用-1来倒计时。
12    for t in range(task_time, 0, -1):
13        info = '请专注任务，还要保持专注 ' + str(t) + ' 秒哦！'
14        print(info, end="")
15        print("\b" * (len(info) * 2), end="", flush=True)
16    #\b是退格的意思，将原有的的内容从终端退出去。
17    #len()是计算字符串长度，在这里是计算info字符串的长度，因为汉字占2个字节，所以乘以2。
18    #"\b"*(len(info)*2)的意思是将终端里的内容退出。
19    #end=""是不换行的意思，flush=True是刷新的意思，即刷新终端内容后不换行。
```

```

20         time.sleep(1)
21     print('你已经专注了 %d 分钟，很棒~再加把劲，完成任务!' % task_time) # 倒计时后，才继续运行之后
22     的代码。
23
24     # 询问任务是否完成
25     task_status = input('请在任务完成后按输入y:')
26     if task_status == 'y':
27         end = time.time() # 一定用户按了 y，就记下结束时间。
28         end_time = time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(time.time())) # 日
29         期格式化
30         actual_time = int((end - start) / 60) # 始末时间相减，从秒换算到分，除以60。
31         start_end = start_time + '—' + end_time + '\n'
32         with open('timelog3.txt', 'a', encoding = 'utf-8') as f:
33             f.write(task_name + ' 的预计时长为: ' + str(task_time) + '分钟\n')
34             f.write(task_name + ' 的实际时长为: ' + str(actual_time) + '分钟,具体时间为: ' +
35             start_end)
36             again = input('建立一个新任务请按 y, 退出时间日志记录器请按 q: ')
37             if again == 'q':
38                 break
39         else:
40             print('抱歉，你的输入有误。请重启时间记录器。')
41
42     print('愿被你善待的时光，予你美好的回赠。')

```

## 课后练习2：再出古诗默写题

- 题目解析：
  - 替换诗句的逻辑同第15关的古诗默写练习题
  - 使用os.replace(src,dst)函数来修改文件名，src是原文件名，dst是目标文件名。
- 参考代码：

```

1 import os
2 list_test = ['一弦一柱思华年。\\n', '只是当时已惘然。\\n']
3
4 with open ('poem3.txt', 'r') as f:
5     lines = f.readlines()
6
7 with open('poem_new.txt', 'w') as new:
8     for line in lines:
9         if line in list_test:
10             new.write('_____。\\n')

```

```
11     else:  
12         new.write(line)  
13  
14 os.replace('poem_new.txt', 'poem3.txt')
```

## 第17关 邮件发送