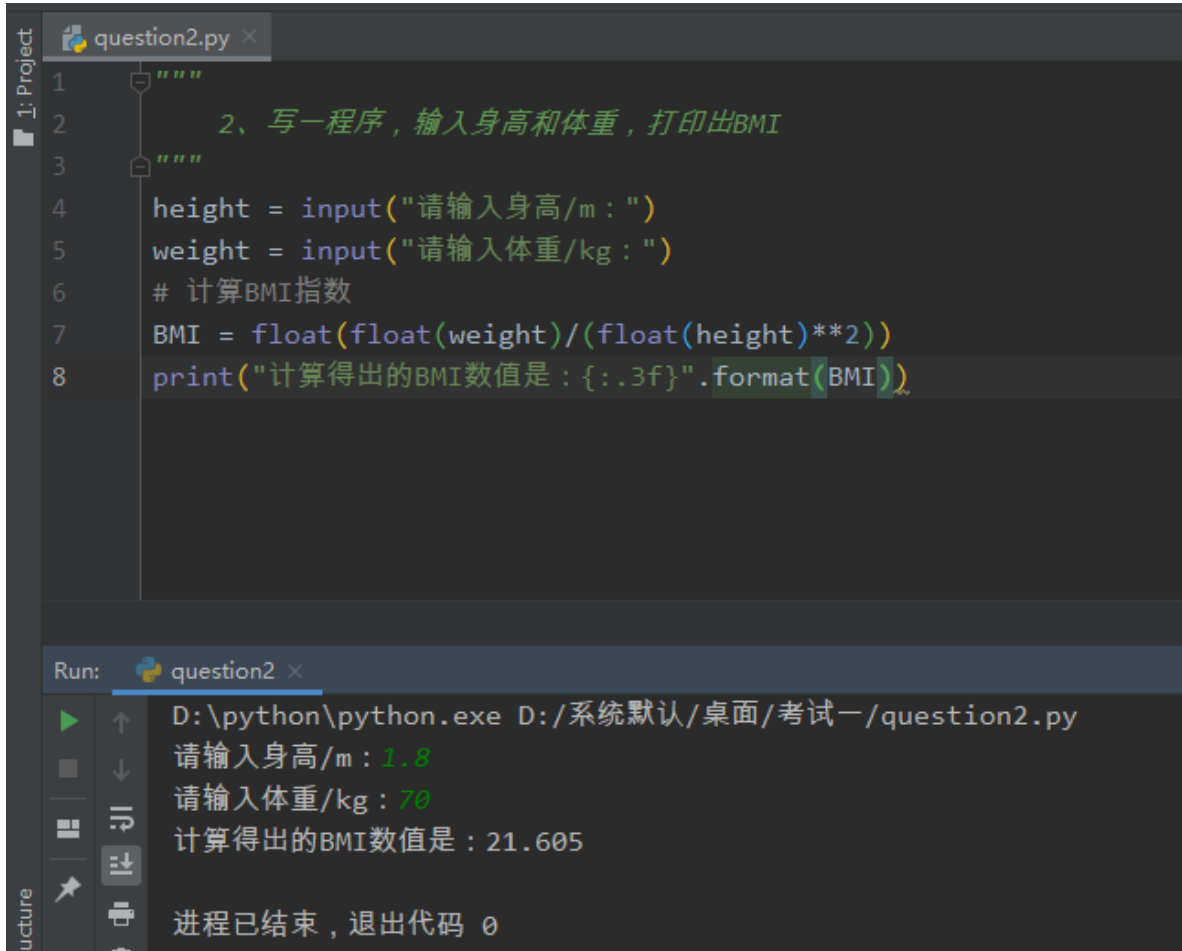


2、写一程序，输入身高和体重，打印出BMI

代码如下：

```
height = input("请输入身高/m: ")
weight = input("请输入体重/kg: ")
# 计算BMI指数
BMI = float(float(weight)/(float(height)**2))
print("计算得出的BMI数值是: {:.3f}".format(BMI))
```

运行截图：



The screenshot shows a Python IDE with a file named 'question2.py'. The code in the editor is as follows:

```
1 """
2     2、写一程序，输入身高和体重，打印出BMI
3 """
4 height = input("请输入身高/m: ")
5 weight = input("请输入体重/kg: ")
6 # 计算BMI指数
7 BMI = float(float(weight)/(float(height)**2))
8 print("计算得出的BMI数值是: {:.3f}".format(BMI))
```

Below the editor, the 'Run' window shows the execution of 'question2.py'. The output is:

```
D:\python\python.exe D:/系统默认/桌面/考试一/question2.py
请输入身高/m: 1.8
请输入体重/kg: 70
计算得出的BMI数值是: 21.605
进程已结束，退出代码 0
```

算法描述：

首先，用两个input()语句分别来接收身高和体重，然后用公式体重除以身高的平方计算得出BMI，因为input()语句是字符串，所以需要输入的数据变成浮点数，最后打印用格式化语句保留小数点后三位有效数字！

3、某人第一天存款10元，第二天存款20元，第三天存款40元，每天的存款为前一天的两倍
请用递归程序来计算第十天时应该存入多少元，10天共存了多少元？

代码如下：

```
def save_each_day(i):
    if i == 1:
        money = 10
```

```

else:
    # i天存款是i-1天的2倍
    money = 2 * save_each_day(i-1)
return money

def save_total(i):
    count = 0
    for i in range(1,i+1):
        count += save_each_day(i)
    return count

if __name__ == '__main__':
    x = save_each_day(10)
    y = save_total(10)
    print("第10天存了{}元，共存了{}元".format(x,y))

```

运行截图：

```

5  def save_each_day(i):
6      if i == 1:
7          money = 10
8      else:
9          # i天存款是i-1天的2倍
10         money = 2 * save_each_day(i-1)
11     return money
12
13  def save_total(i):
14      count = 0
15      for i in range(1,i+1):
16          count += save_each_day(i)
17      return count
18
19
20  if __name__ == '__main__':
21      x = save_each_day(10)
22      y = save_total(10)
23      print("第10天存了{}元，共存了{}元".format(x,y))

```

Run: question3 ×

D:\python\python.exe D:/系统默认/桌面/考试一/question3.py

第10天存了5120元，共存了10230元

进程已结束，退出代码 0

算法描述：

使用两个函数，分别代表每天存钱数以及累计存钱数，唯一参数都是当前天 i ，其中每天存钱的函数 $\text{save_each_day}(i)$ 判断 i 为1的话就给出值10，否则就是前一天的两倍（调用本函数）

累计存钱 $\text{save_total}(i)$ 表示 i 天累计存钱数，可以用 for 循环从1遍历到 $i+1$ ，然后累加，返回值主函数部分给出初值，然后分别调用两个子函数，最后格式化输出！

4、

写一个程序，输入班上同学（10个）参加TOEIC考试（范围0–120）所考分数，求：

- （a）全班最高分
- （b）105分以上人数
- （c）介于80–95（含95）的人数
- （d）全班总平均分（当输入140这个数字时，代表输入结束，直接跳出循环）

输入案例：105 110 90 95 87 78 100 83 67 80 140

代码如下：

```
# 根据题意知道是1位同学1位同学的录入
score = int(input("请输入1位同学的成绩: "))
# 创建列表，存储各位同学的成绩
scores_all = []
# 大于105分的列表
scores_bigger = []
# 介于80-95分的列表
scores_80_95 = []
# 判断当前输入的成绩是否跳出循环
while score != 140:
    # 只有在输入不是140的时候有效
    scores_all.append(score)
    if score > 105:
        scores_bigger.append(score)
    if score > 80 and score <= 105:
        scores_80_95.append(score)
    score = int(input("请继续输入一位同学的成绩: "))
# 循环结束后
print("全班最高分是: {}".format(max(scores_all)))
print("105分以上有{}人".format(len(scores_bigger)))
print("介于80-95分有{}人".format(len(scores_80_95)))
print("全班总平均分是: {}分".format(sum(scores_all) / len(scores_all)))
```

运行截图：

```
question2.py x question3.py x question4.py x
1 """
2     写一个程序，输入班上同学（10个）参加TOEIC考试（范围0-120）所考分数，
3     (a) 全班最高分
4     (b) 105分以上人数
5     (c) 介于80—95（含95）的人数
6     (d) 全班总平均分（当输入140这个数字时，代表输入结束，直接跳出循环）
"""
while score != 140
Run: question4 x
D:\python\python.exe D:/系统默认/桌面/考试一/question4.py
请输入1位同学的成绩：105
请继续输入一位同学的成绩：110
请继续输入一位同学的成绩：90
请继续输入一位同学的成绩：95
请继续输入一位同学的成绩：87
请继续输入一位同学的成绩：78
请继续输入一位同学的成绩：100
请继续输入一位同学的成绩：83
请继续输入一位同学的成绩：67
请继续输入一位同学的成绩：80
请继续输入一位同学的成绩：140
全班最高分是：110
105分以上有1人
介于80-95分有6人
全班总平均分是：89.5分
进程已结束，退出代码 0
```

算法描述：

首先，最开始输入一位同学的成绩，并转换为整型，然后创建几个列表分别来存储全班同学的成绩，大于105的成绩，介于80-95的成绩，然后再while循环里，继续输入9位同学的成绩，使用if分支语句进行判断当前分数所在的段，从而存在不同的列表！
最后，输入140中断循环，分别打印出以上几个要求！

5、某人第一天存款1+3+5元，第二天存款1+3+5+7+9元，第三天存款1+3+5+7+9+11+13元，第四天存款1+3+5+7+9+11+13+15+17元，依此类推，请计算第10天存多少元？10共存多少元？

代码如下：

```
"""
(1) 可以看出是 第i天存款是1+3+.....+ (4i+1)元，那么i天总共存款便可以计算出来了！
"""
# 计算每天存款多少元
def save_every_day(i):
    # 用于计数
    count = 0
    # 步长设置为2，将奇数保存下来
    for m in range(1, 4*i+2, 2):
        count += m
    return count
print("第10天存%s元"%save_every_day(10))
```

```

# 计算目标天数内存了多少元
def save_total(i):
    # 用于计数
    count = 0
    for m in range(1,i+1):
        count += save_every_day(m)
    return count
print("10天共存{}元".format(save_total(10)))

```

运行截图：

```

1 """
2     (1) 可以看出是 第i天存款是1+3+.....+ (4i+1) 元，那么i天总共存款便可以计算出:
3 """
4 # 计算每天存款多少元
5 def save_every_day(i):
6     # 用于计数
7     count = 0
8     # 步长设置为2，将奇数保存下来
9     for m in range(1,4*i+2,2):
10         count += m
11     return count
12 print("第10天存%5元"%save_every_day(10))
13 # 计算目标天数内存了多少元
14 def save_total(i):
15     # 用于计数
16     count = 0
17     for m in range(1,i+1):
18         count += save_every_day(m)
19     return count
20 save_total(10)

```

Run: question5 ×

D:\python\python.exe D:/系统默认/桌面/考试一/question5.py

第10天存441元

10天共存1770元

进程已结束，退出代码 0

算法描述：

可以看出是 第 i 天存款是 $1+3+.....+(4i+1)$ 元，那么 i 天总共存款便可以计算出来了！

构造两个函数，唯一参数都是当前天 i ，首先计算每天存款，这个很明显是所有的加数都是奇数项，然后项的个数满足 $4*i+1$ 个，所以可以用循环以及range加上步长来计算，当前天累计这个可以调用第一个函数，用for循环从1遍历到 $i+1$ ，计算得出的便是 i 天总存款！

6、使用列表，输入10位同学的学号（学号为1，2.....）和成绩（0—100），根据成绩高低，由高到低印出每位同学的学号和成绩！

代码如下：

```

"""
    输入案例：
        学号： [1,2,3,4,5,6,7,8,9,10]
        成绩： [90,87,100,66,80,71,68,60,54,81]
"""

scores = eval(input("输入10位同学的成绩： "))
numbers = eval(input("输入对应的1-位同学的学号： "))
# 排序后列表
scores_sorted = sorted(scores,reverse=True)
for i in range(len(scores_sorted)):
    for j in range(len(scores)):
        if scores[j] == scores_sorted[i]:
            print("{} {}".format(numbers[j],scores[j]))

```

运行截图：

```

question6.py
2      输入案例：
3      学号： [1,2,3,4,5,6,7,8,9,10]
4      成绩： [90,87,100,66,80,71,68,60,54,81]
5      """
6      scores = eval(input("输入10位同学的成绩： "))
7      numbers = eval(input("输入对应的1-位同学的学号： "))
8      # 排序后列表
9      scores_sorted = sorted(scores,reverse=True)
10     for i in range(len(scores_sorted)):
11         for j in range(len(scores)):
12             if scores[j] == scores_sorted[i]:
13                 print("{} {}".format(numbers[j],scores[j]))

```

Run: question6

```

输入对应的1-位同学的学号： [1,2,3,4,5,6,7,8,9,10]
3,100
1,90
2,87
10,81
5,80
6,71
7,68
4,66
8,60
9,54

进程已结束，退出代码 0

```

算法描述：

首先将输入的列表数据用eval函数转换为其原来的格式即list，然后创建一个变量，用于给scores列表进行从高到低的排序，加上了reverse = True参数，然后可以用双层for循环，将成绩以及对应的学号输出！

7、写一程序，输入一周7天的空气品质AQI值（即一个整数），若AQI值低于50，则“空气良好”，值较高但低于100，则“空气普通”，高于100低于150，则“敏感族不适合”；其他值，则“全部族群不适合”，求当输入特定AQI值，打印出对应的文字结果！

代码如下：

```
# 输入一周七天的AQI值（整数）
i = 1
# 创建星期字典
week_dict = {
    1: "周一",
    2: "周二",
    3: "周三",
    4: "周四",
    5: "周五",
    6: "周六",
    7: "周日"
}
while i <= 7:
    AQI = int(input("请输入当天的空气品质AQI值: "))
    if AQI < 50:
        print("{} {}".format(week_dict[i], "空气良好"))
    elif AQI < 100:
        print("{} {}".format(week_dict[i], "空气普通"))
    elif AQI < 150:
        print("{} {}".format(week_dict[i], "敏感族不适合"))
    else:
        print("{} {}".format(week_dict[i], "全部族群不适合"))
    i += 1
```

运行截图：

```
1 """
2     输入案例：
3         40 50 60 70 80 100 160
4 """
5 # 输入一周七天的AQI值（整数）
6 i = 1
7 # 创建星期字典
8 week_dict = {
9     1: "周一",
10    2: "周二",
11    3: "周三",
12    4: "周四",
13    5: "周五",
14    6: "周六",
15    7: "周日"
16 }
17
18 # 循环输入7天的AQI值
19 while i <= 7:
20     aqi = int(input("请输入当天的空气品质AQI值："))
21     # 根据AQI值判断空气质量
22     if aqi <= 50:
23         print("空气良好")
24     elif 51 <= aqi <= 100:
25         print("空气轻度污染")
26     elif 101 <= aqi <= 150:
27         print("空气中度污染")
28     elif 151 <= aqi <= 200:
29         print("空气重度污染")
30     else:
31         print("空气污染严重")
32     i += 1
33
34 # 输出结果
35 for i, day in week_dict.items():
36     print(day, "空气", "普通" if i < 6 else "不适合")
37
38 # 进程已结束，退出代码 0
```

算法描述：

首先，需要的是一周7天的空气指标，所以用while循环来解决，另外定义一个字典，通过键值对匹配能输出当前的日期，循环内部是多分支循环结构，根据输入的AQI指标来决定其对应的段，最后打印输出！

8、共有25根牙签，依次最多取3根（也可以取1根或者2根），取到最后一根（第25根）牙签就算输，写一程序，让使用者和PC比赛，使用者先开始去取牙签，然后让PC永远胜利！

（ps：PC赢得键：取到第4根—第8根—第12根—第16根—第20根—第24根）

代码如下：

运行截图：

算法描述：

