实验五 Python 模块应用

一、实验目的

1 理解模块与包的概念及用法。

2 掌握 Python 内置模块的基本使用方法。

3 掌握第三方库的使用。

二、实验内容

1、上课笔记（拍照，签名）

2、本实验主要包含以下验证内容。

（1）模块的导入方法。

（2）random 标准库的使用。

（3）time 库的使用。

（4）第三方库的安装 pip3 常用命令

（5）pyinstaller 第三方库的使用。

3、编程实现下列功能

(1)拼手气红包，发红包时用户输入一个红包总金额 total 和待发红包总数 num，发布红包后，其它用户抢红包时可以随机得到不定金额的红包（前 num 个人中每个人都能抢到红包），RP 好的可能抢到几块，RP 不好时可能只会抢到几毛，甚至几分钱。请编程实现上面这一个过程，并输出每个人的红包金额数。

```
import random
def hongbao(total, num):
    #total表示拟发红包总金额
    #num表示拟发红包数量
    each = []
    #已发红包总金额
    already = 0
    for i in range(1, num):
        #为当前抢红包的人随机分配金额
        #至少给剩下的人每人留一分钱
        请实现这部分代码
        each.append(t)
        already = already+t
    #剩余所有钱发给最后一个人
    请实现这部分代码
    return each

def test():
    for i in range(20):    #测试20次
        each = hongbao(20, 7)
        print('第{}种随机分配方案：  '.format(i+1),end='')
        print(*each,sep=',  ')

if __name__ =="__main__":
    test()
```

(2) 将 1 题中的代码保存为模块 hongbao.py，练习模块的导入与使用，并使用 pyinstaller 将 hongbao.py 打包 exe。

```
>>> import hongbao
>>> hongbao.hongbao(100,10)
[83.55, 12.06, 1.69, 1.48, 0.71, 0.07, 0.18, 0.08, 0.04, 0.14]
>>> hongbao.hongbao(1,100)
[0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01,
0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01,
0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01,
0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01,
0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01,
0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01, 0.01]
```

(3) 猜数字游戏。在程序中预设一个 0-9 之间的整数，让用户通过键盘输入所猜的数，如果大于预设的数，显示“你猜的数字大于正确答案”；小于预设的数，显示“你猜的数字小于正确答案”，如此循环，直至猜中该数，显示“你猜了 N 次，猜对了，真厉害”,其中 N 是用户输入数字的次数。

```
guess = 0      #用户输入的数字
secret = 7     #预设的数字
times = 1      #猜数字的次数

print('--------------欢迎参加猜数字游戏，请开始--------------')
while guess !=secret:
    guess = int(input("请输入0~9之前的数字： "))
    print("你输入数字是:", guess)
    if guess ==secret:
        print('你猜了{}次，猜对了，真厉害'.format(times))
    elif guess<secret:
        print('你猜的数字小于正确答案')
    else:
        print('你猜的数字大于正确答案')
    times+=1
print('game over')
```

(4) 猜数字游戏续。改编(3)中的猜数字游戏，让计算机能够随机产生一个预设数字，范围在 0-100 之间，其他游戏规则不变。

(5) 猜数字游戏再续。用变量 maxtimes 设置允许猜数字的最大次数(比如最多只允许猜 6 次(maxtimes=6)，并在猜错后提示还有几次机会。**用 for 循环改写整个程序**。(提示，猜对后可使用 break 跳出循环)

输出效果如下所示：

```
----------欢迎参加猜数字游戏，请开始----------
请输入0~100之前的数字：80
你输入数字是: 80
你猜的数字大于正确答案，还有5次机会
请输入0~100之前的数字：40
你输入数字是: 40
你猜的数字小于正确答案，还有4次机会
请输入0~100之前的数字：60
你输入数字是: 60
你猜的数字大于正确答案，还有3次机会
请输入0~100之前的数字：50
你输入数字是: 50
你猜的数字大于正确答案，还有2次机会
请输入0~100之前的数字：45
你输入数字是: 45
你猜了5次，猜对了，真厉害
game over
```

(6) 猜数字游戏之续了又续。为了增加代码的复用性，将猜数字游戏封装为函数 GuessSecret(maxtimes)，将允许猜数字的最大次数 maxtimes 作为参数。在调用 GuessSecret 时允许用户自己设置 maxtimes，美化程序的输出界面。

```
import random

def  GuessSecret(maxtimes):
        函数的实现代码

maxts=eval(input("@请输入猜数字的最大次数:"))
GuessSecret(maxts)  <-- 此处调用GuessSecret函数
```

执行效果如下图所示。

```
请输入猜数字的最大次数：7
   ---------------------------------------------------------
   ---                                                   ---
   ---    欢迎参加猜数字游戏，请开始                      ---
   ---                                                   ---
   ---------------------------------------------------------
请输入0~100之前的数字：30
你输入数字是: 30
你猜的数字大于正确答案，还有6次机会
请输入0~100之前的数字：15
你输入数字是: 15
你猜的数字大于正确答案，还有5次机会
请输入0~100之前的数字：7
你输入数字是: 7
你猜的数字小于正确答案，还有4次机会
请输入0~100之前的数字：10
你输入数字是: 10
你猜了4次，猜对了，真厉害
game over
```

思考题：

从发红包的输出结果，可以看到每次发红包的，抢红包的人越靠前，抢到红包的金额越大，试着修改代码，尽量使大红包分布均匀。

```
第1种随机分配方案：15.92，2.45，1.35，0.07，0.04，0.11，0.06
第2种随机分配方案：10.91，7.83，1.04，0.1，0.09，0.02，0.01
第3种随机分配方案：17.1，2.62，0.18，0.02，0.03，0.02，0.03
第4种随机分配方案：8.93，0.47，4.65，5.11，0.33，0.43，0.08
第5种随机分配方案：2.74，2.5，7.62，4.19，2.56，0.1，0.29
第6种随机分配方案：14.57，5.07，0.06，0.03，0.13，0.03，0.11
第7种随机分配方案：19.82，0.1，0.02，0.02，0.02，0.01，0.01
第8种随机分配方案：16.34，0.54，2.11，0.18，0.2，0.59，0.04
第9种随机分配方案：4.61，10.06，0.48，1.89，2.06，0.63，0.27
第10种随机分配方案：15.22，1.28，1.79，0.49，0.96，0.06，0.2
第11种随机分配方案：9.27，7.88，2.27，0.31，0.04，0.06，0.17
第12种随机分配方案：13.1，2.46，1.36，2.14，0.13，0.51，0.3
第13种随机分配方案：4.52，5.87，4.64，0.13，0.92，1.26，2.66
第14种随机分配方案：13.54，3.24，0.28，0.27，1.76，0.55，0.36
第15种随机分配方案：0.2，3.56，0.44，12.01，2.05，0.7，1.04
第16种随机分配方案：8.95，6.45，0.4，3.36，0.09，0.71，0.04
第17种随机分配方案：4.61，5.86，2.12，0.52，4.7，1.3，0.89
第18种随机分配方案：17.88，0.61，0.16，0.19，0.35，0.06，0.75
第19种随机分配方案：8.18，5.69，2.81，1.6，0.47，1.15，0.1
第20种随机分配方案：4.39，6.79，6.68，0.92，0.37，0.52，0.33
```