山东大学网络空间安全学院

Python高级程序设计课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202100460065 | 姓名：李昕 | | 班级：21级密码2班 |
| 实验题目：实验3-函数的定义与使用 | | | |
| 实验学时：2 | | 实验日期：2022/10/13 | |
| 实验目的：熟悉函数的定义与使用，编写代码，完成对自定义函数的编写与调用 | | | |
| 硬件环境：  AMD Ryzen 7 5800H with Radeon Graphics     3.20 GHz  机带 RAM 16.0 GB (13.9 GB 可用) | | | |
| 软件环境：  操作系统：windows 11  编译器：IDLE | | | |
| 实验步骤与内容：    1.熟悉默认值参数、关键参数和可变长度参数的用法;  2.一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.编写程序找出1000以内的所有完数，要求结合使用lambda表达式和filter、sum函数。  3.假设一段楼梯共n个台阶，小明一步最多能上3 个台阶，编写递归函数计算小明上这段楼梯一共有多少种方法。  4.假设一段楼梯共n个台阶，小明一步最多能上3 个台阶，编写程序列出小明上这段楼梯的所有方法，如n=5时，得到 [(1, 1, 1, 1, 1), (1, 1, 1, 2), (1, 1, 2, 1), (1, 1, 3), (1, 2, 1, 1), (1, 2, 2), (1, 3, 1), (2, 1, 1, 1), (2, 1, 2), (2, 2, 1), (2, 3), (3, 1, 1), (3, 2)]  其中(2, 1, 2)表示小明可以先上2级台阶，再上1级台阶，然后在上2级台阶即可上去，以此类推；  【1000以内的完数】  该问题有三步：  1.利用取余方法求得待定数的虽有因子并将其放入预先开辟的数组中  2.定义 lambda匿名函数，求得列表中所有元素的和的值，并判断该值是否与指定数相等，如果相等则返回True，否则，返回False  3.使用filter函数，将函数返回值为True的数过滤输出  代码实现：  def perfect(a): #输出1000以内的完数  b=[]  for i in range(1,a):  if a%i==0:  b.append(i)  c=lambda b:sum(b)  if c(b)==a:  return True  else:  return False  print('1000以内的完数：',list(filter(perfect,range(1,1000))))  运行结果：    【计算可能的走台阶的种数】  由分析可知，每一步可能的情况有一个台阶，两个台阶，三个台阶，将台阶总数设为n，方法数设为step(n)，可以得到，迈一个台阶后方法数为step(n-1)，两个台阶后为step(n-2)，三个台阶为step(n-3)，则可得step(n)=step(n-1)+step(n-2)+step(n-3)，发现这是一个往下递归的过程，最终当n=3,2,1时分别可直接数出有4,2,1种方法。  代码实现**：**  def step(n):  if n==1:  return 1;  if n==2:  return 2;  if n==3:  return 4;  sum=step(n-1)+step(n-2)+step(n-3);  return sum;  n=int(input('请输入台阶数：'))  print(step(n))  运行结果：    【写出走台阶具体的情况】  同样按照上一个问题的思路，对于n个台阶，则在原有的步数列表基础上分别产生三个子列表，子列表分别添加步数1,2,3，然后将n-1,n-2,n-3送入下一级函数进行递归，最终进行到n=1或2或3的情况，穷举出其可能的步数添加，即可解决问题。只不过关键在于代码的细节问题，即子列表的生成：分别计算n,n-1,n-2,n-3个台阶方法数（这里第一个问题的函数则可派上用场），同样记为step(x)，将n-1送入下一级函数之前，在原有列表基础上对对应的step(n-1)个列表在后面添加“1”，n-2,n-3同理。  代码实现：  def step(n):  if n==1:  return 1;  if n==2:  return 2;  if n==3:  return 4;  sum=step(n-1)+step(n-2)+step(n-3);  return sum;    def way(n,list,i=0):  if n==1:  list[i].append(1);  return;  if n==2:  list[i].append(2);  list[i+1].extend([1,1]);  return;  if n==3:  list[i].append(3);  list[i+1].extend([1,1,1]);  list[i+2].extend([1,2]);  list[i+3].extend([2,1]);  return;  for j in range(i,step(n-1)):  list[j].append(1);  way(n-1,list,i);  i=i+step(n-1);  for j in range(i,i+step(n-2)):  list[j].append(2);  way(n-2,list,i);  i=i+step(n-2);  for j in range(i,i+step(n-3)):  list[j].append(3);  way(n-3,list,i);  def way\_achieve(n,list):  #list=[];  for j in range(step(n)):  list.append([]);  way(n,list);  k=int(input('请输入台阶数：'))  p=[]  way\_achieve(k,p)  print(p)  运行结果： | | | |
| 结论分析与体会：  通过本次练习，我熟悉了解了Python的循环与条件控制语句，对函数的定义与使用有了一定的了解。通过对上台阶应用算法进行分析逐步掌握了递归函数的编写，同时也学习了lambda匿名函数的书写方式。 | | | |