#### 菜鸟教程python100例

本章用于巩固学习的python基础语法，题目普遍比较老，我做完了然后选择出比较适合的题目，并给出解答，考察知识点。做完这些题目，基础语法应该是达到比较扎实的层度，对以后学习算法，机器学习等都有很大的帮助。在菜鸟教程100例中，有一些题目重复或者质量不高，时间不足或者基础比较好，可以选择下面的重点题目进行作答。

**重点题目**

| 题目序号 | 知识点 | 备注 |
| --- | --- | --- |
| 1，2，3，8，15，17，29，31，39，44，49，61，62，68，74，77 | 基础语法 | 涉及较少算法，必须熟练掌握 |
| 4，10，16，50，92，93，95 | 模块 | 主要涉及datetime模块 |
| 6，11，12，13，14，19，24，27，30，36，80，89 | 基础算法 | 部分使用模拟即可得到结果，部分涉及到递归可以先不要求掌握 |
| 51，52，53，54，55 | 二进制运算 | 主要是二进制的使用，基础知识 |
| 56，57，58，63，64，65 | 画图 | 使用matplotlib画图，可以日后学习 |
| 69，72，73 | 数据结构与算法 | 涉及对链表的操作，数据结构与算法的基础，日后学习 |
| 97，98，99 | 文件读写 | 熟练掌握 |

本章题目全部来源于[菜鸟教程](https://www.runoob.com/python/python-100-examples.html)，从刚开始学习就接触到了菜鸟教程，开源免费，真的是非常伟大的一个网站。网站上的python100例的python解释器是2.7，本章我会用最新的解释器3.12把题目做一遍。本章对于每一道题我会给出解答和考察知识点，方便查漏补缺。

##### 练习题1

有四个数字：1、2、3、4，能组成多少个互不相同且无重复数字的三位数？各是多少？

分析：使用嵌套循环的方式来找出所有满足条件的三位数

count = 0 # 计数器，用于记录满足条件的三位数的个数  
  
for i in range(1, 5): # 百位数字的取值范围为 1 到 4  
 for j in range(1, 5): # 十位数字的取值范围为 1 到 4  
 for k in range(1, 5): # 个位数字的取值范围为 1 到 4  
 if i != j and j != k and i != k: # 确保三个数字互不相同  
 num = i \* 100 + j \* 10 + k # 计算三位数的值  
 print(num) # 输出满足条件的三位数  
 count += 1  
  
print("共有", count, "个满足条件的三位数")

知识点：循环语句

##### 练习题2

企业发放的奖金根据利润提成。利润(I)低于或等于10万元时，奖金可提10%；利润高于10万元，低于20万元时，低于10万元的部分按10%提成，高于10万元的部分，可提成7.5%；20万到40万之间时，高于20万元的部分，可提成5%；40万到60万之间时高于40万元的部分，可提成3%；60万到100万之间时，高于60万元的部分，可提成1.5%，高于100万元时，超过100万元的部分按1%提成，从键盘输入当月利润I，求应发放奖金总数？

分析：利用input()函数获取利润，根据if else来判断获得的奖金

profit = float(input("请输入当月利润（单位：万元）："))  
  
bonus = 0 # 初始化奖金总额为0  
  
if profit <= 10:  
 bonus = profit \* 0.1  
elif profit <= 20:  
 bonus = 10 \* 0.1 + (profit - 10) \* 0.075  
elif profit <= 40:  
 bonus = 10 \* 0.1 + 10 \* 0.075 + (profit - 20) \* 0.05  
elif profit <= 60:  
 bonus = 10 \* 0.1 + 10 \* 0.075 + 20 \* 0.05 + (profit - 40) \* 0.03  
elif profit <= 100:  
 bonus = 10 \* 0.1 + 10 \* 0.075 + 20 \* 0.05 + 20 \* 0.03 + (profit - 60) \* 0.015  
else:  
 bonus = 10 \* 0.1 + 10 \* 0.075 + 20 \* 0.05 + 20 \* 0.03 + 40 \* 0.015 + (profit - 100) \* 0.01  
  
print("应发放奖金总数为：", bonus, "万元")

知识点：从控制台或者输入；if elif .... else表达式；算术运算符

##### 练习题3

一个整数，它加上100后是一个完全平方数，再加上168又是一个完全平方数，请问该数是多少？

分析：总体思路构建一个等式，满足上面完全平方数的规定；设整数为i，加上100后是一个完全平方数则可知i+100>0，所以遍历i从-99开始；随便举一个数字200，200的平方是40000，201的平放是40401，之间相差401，后面的数不可能满足题目中要求的一个完全平方数再加上168后又是一个完全平方数，所以i的结束遍历可以是400.

import math  
  
for i in range(-100, 10000):  
 x = math.isqrt(i + 100) # 计算 i + 100 的平方根  
 y = math.isqrt(i + 268) # 计算 i + 268 的平方根  
 if x \* x == i + 100 and y \* y == i + 268: # 判断是否满足完全平方数的条件  
 print("符合条件的整数是：", i)

知识点：算术表达式；循环遍历；算法基础

##### 练习题4

输入某年某月某日，判断这一天是这一年的第几天？

分析：input()函数获取控制台的输入信息；使用python中的datatime模块处理日期信息

from datetime import datetime  
  
date\_str = input("请输入日期（格式为yyyy-mm-dd）：")  
date = datetime.strptime(date\_str, "%Y-%m-%d") # 将输入的日期字符串转换为 datetime 对象  
  
day\_of\_year = date.timetuple().tm\_yday # 获取该日期在一年中的天数  
  
print("这一天是这一年的第", day\_of\_year, "天")

知识点：datetime模块，查阅资料

##### 练习题5

输入三个整数x,y,z，请把这三个数由小到大输出。

分析：使用列表来保存3个整数信息；使用sort函数来对列表进行排序

x = int(input("请输入第一个整数："))  
y = int(input("请输入第二个整数："))  
z = int(input("请输入第三个整数："))  
  
numbers = [x, y, z] # 将输入的三个整数存储在列表中  
numbers.sort() # 对列表进行排序，默认是升序排序  
  
print("从小到大排序后的结果为：", numbers)

知识点：列表；排序函数

##### 练习题6

斐波那契数列

分析：构造斐波那契数列的等式，用代码表示出来

def fibonacci(n):  
 fib\_sequence = [0, 1] # 初始化斐波那契数列的前两个数字  
 if n <= 1:  
 return fib\_sequence[:n] # 返回前 n+1 个数字  
 else:  
 for i in range(2, n + 1):  
 fib\_sequence.append(fib\_sequence[i - 1] + fib\_sequence[i - 2]) # 计算并添加下一个数字  
 return fib\_sequence  
  
n = int(input("请输入斐波那契数列的长度："))  
fib\_numbers = fibonacci(n)  
print("斐波那契数列的前", n, "个数字为：", fib\_numbers)

知识点：函数；递推；循环

##### 练习题7

将一个列表的数据复制到另一个列表中。

分析：使用切片，使用copy函数

list1 = [1, 2, 3, 4, 5]  
list2 = list1[:] # 使用切片操作将 list1 的数据复制到 list2  
list3 = list1.copy() # 使用 copy() 方法将 list1 的数据复制到 list3  
  
print("复制前的列表 list1:", list1)  
print("复制后的列表 list2:", list2)  
print("复制后的列表 list3:", list3)

知识点：数组的操作

##### 练习题8

输出 9\*9 乘法口诀表

分析：使用嵌套循环和标准输出

for i in range(1, 10):  
 for j in range(1, 10):  
 result = i \* j  
 print(i, "x", j, "=", result)  
 print() # 输出空行，用于分隔每一行

知识点：嵌套循环；print()

##### 练习题9

暂停一秒输出。

分析：time模块的sleep()函数

import time  
  
print("开始输出")  
time.sleep(1) # 暂停一秒  
print("暂停一秒后输出")

知识点：time模块

##### 练习题10

暂停一秒输出，并格式化当前时间。

分析：datetime模块；sleep()函数和strftime()函数

import time  
from datetime import datetime  
  
print("开始输出")  
time.sleep(1) # 暂停一秒  
  
current\_time = datetime.now().strftime("%Y-%m-%d %H:%M:%S")  
print("暂停一秒后的当前时间：", current\_time)

知识点：time模块，格式化时间

##### 练习题11

古典问题：有一对兔子，从出生后第3个月起每个月都生一对兔子，小兔子长到第三个月后每个月又生一对兔子，假如兔子都不死，问每个月的兔子总数为多少？

分析：第 1 个月：1 对兔子（初始兔子对数）；第 2 个月：1 对兔子；第 3 个月：2 对兔子（初始兔子对数 + 上个月的兔子对数）；第 4 个月：3 对兔子（上个月的兔子对数 + 上上个月的兔子对数）；第 5 个月：5 对兔子（上个月的兔子对数 + 上上个月的兔子对数）；第 6 个月：8 对兔子（上个月的兔子对数 + 上上个月的兔子对数）；就是变形的斐波那契数列

def fibonacci(n):  
 fib\_sequence = [1, 1] # 前两个月的兔子数  
 if n <= 2:  
 return fib\_sequence[:n] # 返回前 n 个月的兔子数  
 else:  
 for i in range(2, n):  
 fib\_sequence.append(fib\_sequence[i - 1] + fib\_sequence[i - 2]) # 计算并添加下一个月的兔子数  
 return fib\_sequence  
  
n = int(input("请输入月份："))  
rabbit\_counts = fibonacci(n)  
print("第", n, "个月的兔子总数为：", rabbit\_counts[-1])

知识点：斐波那契数列；函数；列表；循环

##### 练习题12

101-200之间有多少个素数，并输出所有素数。

分析：循环101到200，使用函数判断循环中的素数

def is\_prime(n):  
 if n < 2:  
 return False  
 for i in range(2, int(n\*\*0.5) + 1):  
 if n % i == 0:  
 return False  
 return True  
  
count = 0  
primes = []  
  
for num in range(101, 201):  
 if is\_prime(num):  
 count += 1  
 primes.append(num)  
  
print("101到200之间的素数数量为:", count)  
print("所有素数为:", primes)

知识点：循环；函数；素数判断

##### 练习题13

打印出所有的"水仙花数"，所谓"水仙花数"是指一个三位数，其各位数字立方和等于该数本身。例如：153是一个"水仙花数"，因为153=1的三次方＋5的三次方＋3的三次方

分析：将数字转换为字符串获得每一个字符，然后将字符转换为数字获得个位，十位和百位，最后使用等式进行判断

def is\_armstrong\_number(n):  
 num\_str = str(n)  
 sum\_of\_cubes = sum(int(digit)\*\*3 for digit in num\_str)  
 return n == sum\_of\_cubes  
  
armstrong\_numbers = []  
  
for num in range(100, 1000):  
 if is\_armstrong\_number(num):  
 armstrong\_numbers.append(num)  
  
print("所有的水仙花数为:", armstrong\_numbers)

知识点：函数；字符串和数字的转换

##### 练习题14

将一个正整数分解质因数。例如：输入90,打印出90=2\*3\*3\*5。

分析：因为合数总能分解为质数，所以将正整数除以从2开始的数即可，如果能除尽则一直除，直到得到的结果为1.

def prime\_factors(n):  
 factors = []  
 i = 2  
 while i <= n:  
 if n % i == 0:  
 factors.append(i)  
 n //= i  
 else:  
 i += 1  
 return factors  
  
num = int(input("请输入一个正整数："))  
factors = prime\_factors(num)  
  
print(num, "=", "\*".join(map(str, factors)))

知识点：**算法基础**；函数；字符串拼接

##### 练习题15

利用条件运算符的嵌套来完成此题：学习成绩>=90分的同学用A表示，60-89分之间的用B表示，60分以下的用C表示。

分析：基础题目：if elif .....else进行判断即可

score = int(input("请输入学习成绩："))  
  
grade = 'A' if score >= 90 else ('B' if score >= 60 else 'C')  
  
print("成绩等级：", grade)

知识点：判断语句

##### 练习题16

输出指定格式的日期。

分析：利用datatime模块，获取当前日期，然后格式化日期

from datetime import datetime  
  
# 获取当前日期和时间  
now = datetime.now()  
  
# 指定日期格式  
date\_format = "%Y-%m-%d" # 例如：2022-01-01  
# date\_format = "%d/%m/%Y" # 例如：01/01/2022  
# date\_format = "%B %d, %Y" # 例如：January 01, 2022  
  
# 格式化日期  
formatted\_date = now.strftime(date\_format)  
  
# 输出格式化后的日期  
print("当前日期:", formatted\_date)

知识点：datatime模块，格式化日期

##### 练习题17

输入一行字符，分别统计出其中英文字母、空格、数字和其它字符的个数

分析：得到输入字符，然后使用计数器，如果出现一个英文字母，则英文字母计数器加1，其他字符一样

line = input("请输入一行字符：")  
  
letter\_count = 0  
space\_count = 0  
digit\_count = 0  
other\_count = 0  
  
for char in line:  
 if char.isalpha():  
 letter\_count += 1  
 elif char.isspace():  
 space\_count += 1  
 elif char.isdigit():  
 digit\_count += 1  
 else:  
 other\_count += 1  
  
print("英文字母个数：", letter\_count)  
print("空格个数：", space\_count)  
print("数字个数：", digit\_count)  
print("其他字符个数：", other\_count)

知识点：字符函数的使用；循环

##### 练习题18

求s=a+aa+aaa+aaaa+aa...a的值，其中a是一个数字。例如2+22+222+2222+22222(此时共有5个数相加)，几个数相加由键盘控制

分析：从键盘中得到a值和项数两个参数，然后使用计算得到结果即可

a = int(input("请输入一个数字 a："))  
n = int(input("请输入相加的项数 n："))  
  
result = 0  
term = a  
  
for i in range(n):  
 result += term  
 term = term \* 10 + a  
  
print("s =", result)

知识点：函数；input()；基础算法

##### 练习题19

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.编程找出1000以内的所有完数。

分析：首先找到一个数的所有因子，然后求出因子之和，相等则判断这个数是完数

def find\_perfect\_numbers(limit):  
 perfect\_numbers = []  
  
 for num in range(1, limit + 1):  
 factors = []  
 for i in range(1, num):  
 if num % i == 0:  
 factors.append(i)  
   
 if sum(factors) == num:  
 perfect\_numbers.append(num)  
   
 return perfect\_numbers  
  
limit = 1000  
perfect\_numbers = find\_perfect\_numbers(limit)  
  
print("1000以内的完数：", perfect\_numbers)

知识点：求数的因子；函数

##### 练习题20

一球从100米高度自由落下，每次落地后反跳回原高度的一半；再落下，求它在第10次落地时，共经过多少米？第10次反弹多高？

分析：根据题意，直接模拟

height = 100 # 初始高度  
total\_distance = 100 # 总共经过的距离  
  
for \_ in range(1, 10):  
 height /= 2 # 反弹高度减半  
 total\_distance += height \* 2 # 每次落地会经过反弹高度两次  
  
# 第10次落地时，球经过的距离  
print("第10次落地时，共经过的距离：", total\_distance)  
  
# 第10次反弹的高度  
print("第10次反弹的高度：", height)

知识点：函数编写；算术表达式

##### 练习题21

猴子吃桃问题：猴子第一天摘下若干个桃子，当即吃了一半，还不瘾，又多吃了一个第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第10天早上想再吃时，见只剩下一个桃子了。求第一天共摘了多少。

分析：使用逆推法，前一天桃子是后一天桃子+1的两倍

peach = 1 # 第10天剩下的桃子数  
  
for day in range(9, 0, -1):  
 peach = (peach + 1) \* 2  
  
# 第一天摘的桃子数  
print("第一天共摘了", peach, "个桃子")

知识点：函数；算法基础

##### 练习题22

两个乒乓球队进行比赛，各出三人。甲队为a,b,c三人，乙队为x,y,z三人。已抽签决定比赛名单。有人向队员打听比赛的名单。a说他不和x比，c说他不和x,z比，请编程序找出三队赛手的名单

分析：首先，获得所有可能的比赛对手，然后选择出正确的比赛顺序

import itertools  
  
team\_a = ['a', 'b', 'c']  
team\_b = ['x', 'y', 'z']  
  
# 生成所有可能的比赛名单  
matches = list(itertools.permutations(team\_b, 3))  
  
# 遍历比赛名单，找到满足条件的名单  
for match in matches:  
 if match[0] != 'x' and match[2] != 'x' and match[2] != 'z':  
 team\_a\_match = list(team\_a)  
 team\_a\_match.extend(match)  
 print(f"比赛名单：{team\_a\_match[:3]} vs {team\_a\_match[3:]}")  
 break

知识点：算法基础；循环遍历

##### 练习题23

打印出如下图案（菱形）:

\*  
 \*\*\*  
 \*\*\*\*\*  
\*\*\*\*\*\*\*  
 \*\*\*\*\*  
 \*\*\*  
 \*

分析：使用循环结构和条件判断来实现

def print\_diamond\_pattern(n):  
 for i in range(n):  
 # 打印空格  
 for j in range(n - i - 1):  
 print(" ", end="")  
 # 打印星号  
 for j in range(2 \* i + 1):  
 print("\*", end="")  
 print()  
  
 for i in range(n - 2, -1, -1):  
 # 打印空格  
 for j in range(n - i - 1):  
 print(" ", end="")  
 # 打印星号  
 for j in range(2 \* i + 1):  
 print("\*", end="")  
 print()  
  
# 调用函数打印菱形图案  
print\_diamond\_pattern(4)

知识点：循环；判断；函数；算法基础

##### 练习题24

有一分数序列：2/1，3/2，5/3，8/5，13/8，21/13...求出这个数列的前20项之和。

分析：找规律，分子和分母分别满足斐波那契数列

def fibonacci\_fraction\_sum(n):  
 numerator1, numerator2 = 2, 3  
 denominator1, denominator2 = 1, 2  
 fraction\_sum = numerator1 / denominator1 + numerator2 / denominator2  
  
 for \_ in range(3, n + 1):  
 numerator = numerator1 + numerator2  
 denominator = denominator1 + denominator2  
 fraction\_sum += numerator / denominator  
  
 numerator1, numerator2 = numerator2, numerator  
 denominator1, denominator2 = denominator2, denominator  
  
 return fraction\_sum  
  
# 计算前20项之和  
sum\_20 = fibonacci\_fraction\_sum(20)  
print("前20项之和:", sum\_20)

知识点：算法基础；循环

##### 练习题25

求1+2!+3!+...+20!的和。

分析：循环20个数即可，在python中阶乘函数已经实现

import math  
  
def factorial\_sum(n):  
 sum = 0  
 for i in range(1, n+1):  
 factorial = math.factorial(i)  
 sum += factorial  
 return sum  
  
# 计算 1 + 2! + 3! + ... + 20! 的和  
result = factorial\_sum(20)  
print("1 + 2! + 3! + ... + 20! 的和:", result)

知识点：函数；循环

##### 练习题26

利用递归方法求5!

分析：递归是一种调用自身解决问题的方法；基本情况：0! 和 1! 的值都为 1；递归情况：对于大于 1 的数 n，n! 可以表示为 n \* (n-1)!。

def factorial(n):  
 if n == 0 or n == 1:  
 return 1  
 else:  
 return n \* factorial(n-1)  
  
# 计算 5!  
result = factorial(5)  
print("5! =", result)

知识点：递归算法；函数

##### 练习题27

利用递归函数调用方式，将所输入的5个字符，以相反顺序打印出来。

分析：基本情况：当只有一个字符时，直接打印该字符；递归情况：将字符串分为第一个字符和其余部分，然后递归地打印其余部分，并在之后打印第一个字符。

def reverse\_print(string):  
 if len(string) == 1:  
 print(string)  
 else:  
 reverse\_print(string[1:])  
 print(string[0])  
  
# 输入字符  
input\_string = input("请输入一个字符串：")  
  
# 调用函数以相反顺序打印字符  
reverse\_print(input\_string)

知识点：递归算法；函数

##### 练习题28

有5个人坐在一起，问第五个人多少岁？他说比第4个人大2岁。问第4个人岁数，他说比第3个人大2岁。问第三个人，又说比第2人大两岁。问第2个人，说比第一个人大两岁。最后问第一个人，他说是10岁。请问第五个人多大？

分析：利用递归函数来求解第五个人的年龄

def find\_age(n):  
 if n == 1:  
 return 10  
 else:  
 return find\_age(n-1) + 2  
  
# 调用函数求解第五个人年龄  
age\_of\_fifth\_person = find\_age(5)  
print("第五个人年龄:", age\_of\_fifth\_person)

知识点：算法基础；递归

##### 练习题29

给一个不多于5位的正整数，要求：一、求它是几位数，二、逆序打印出各位数字。

分析：先将整数转换为字符串，然后使用字符串的函数求出字符串的长度以及逆序打印数字

def analyze\_number(num):  
 # 转换为字符串，以便进行长度和逆序操作  
 num\_str = str(num)  
  
 # 求出位数  
 num\_digits = len(num\_str)  
  
 # 逆序打印各位数字  
 reversed\_num\_str = num\_str[::-1]  
  
 return num\_digits, reversed\_num\_str  
  
# 输入正整数  
num = int(input("请输入一个不多于5位的正整数: "))  
  
# 调用函数进行分析  
digits, reversed\_num = analyze\_number(num)  
  
# 输出结果  
print("位数:", digits)  
print("逆序打印各位数字:", reversed\_num)

知识点：函数调用；字符串操作

##### 练习题30

一个5位数，判断它是不是回文数。即12321是回文数，个位与万位相同，十位与千位相同

分析：将数字转换为字符串，然后使用循环遍历判断对应位置上的数字字符是否相等

def is\_palindrome(num):  
 # 将数字转换为字符串  
 num\_str = str(num)  
  
 # 比较对应位置上的数字是否相同  
 if num\_str[0] == num\_str[4] and num\_str[1] == num\_str[3]:  
 return True  
 else:  
 return False  
  
# 输入一个5位数  
num = int(input("请输入一个5位数: "))  
  
# 判断是否为回文数  
if is\_palindrome(num):  
 print("是回文数")  
else:  
 print("不是回文数")

知识点：字符串操作；函数编写调用

##### 练习题31

请输入星期几的第一个字母来判断一下是星期几，如果第一个字母一样，则继续判断第二个字母。

分析：if elif .... else判断语句的使用

def determine\_weekday(day):  
 if day.lower() == 'm':  
 return "星期一"  
 elif day.lower() == 't':  
 second\_letter = input("请输入第二个字母：")  
 if second\_letter.lower() == 'u':  
 return "星期二"  
 elif second\_letter.lower() == 'h':  
 return "星期四"  
 else:  
 return "输入无效"  
 elif day.lower() == 'w':  
 return "星期三"  
 elif day.lower() == 'f':  
 return "星期五"  
 elif day.lower() == 's':  
 second\_letter = input("请输入第二个字母：")  
 if second\_letter.lower() == 'a':  
 return "星期六"  
 elif second\_letter.lower() == 'u':  
 return "星期日"  
 else:  
 return "输入无效"  
 else:  
 return "输入无效"  
  
# 输入星期几的首字母  
first\_letter = input("请输入星期几的第一个字母：")  
  
# 调用函数判断星期几  
weekday = determine\_weekday(first\_letter)  
  
# 输出结果  
print(weekday)

知识点：if elif else的嵌套使用

##### 练习题32

按相反的顺序输出列表的值。

分析：列表的使用，切片

def reverse\_print\_list(lst):  
 reversed\_lst = lst[::-1]  
 for item in reversed\_lst:  
 print(item)  
  
# 输入列表  
input\_list = input("请输入一个列表，各元素之间用空格分隔：").split()  
  
# 调用函数以相反顺序打印列表值  
reverse\_print\_list(input\_list)

知识点：列表操作；字符串切分

##### 练习题33

按逗号分隔列表。

分析：使用join的方式将列表中的元素转换为逗号分隔的字符串

def comma\_separated\_list(lst):  
 separated\_str = ', '.join(lst)  
 print(separated\_str)  
  
# 输入列表  
input\_list = input("请输入一个列表，各元素之间用空格分隔：").split()  
  
# 调用函数以逗号分隔列表值  
comma\_separated\_list(input\_list)

##### 练习题34

练习函数调用。

def hello\_runoob():  
 print ('RUNOOB')  
   
def hello\_runoobs():  
 for i in range(3):  
 hello\_runoob()  
if \_\_name\_\_ == '\_\_main\_\_':  
 hello\_runoobs()

##### 练习题35

文本颜色设置。

分析：定义字符串，使用颜色的转义序列来表示颜色

# 示例：设置文本颜色为红色  
def set\_red\_text(text):  
 RED = '\033[91m' # 红色的转义序列  
 RESET = '\033[0m' # 重置颜色的转义序列  
 colored\_text = RED + text + RESET  
 return colored\_text  
  
# 调用示例函数  
message = "Hello, World!"  
colored\_message = set\_red\_text(message)  
print(colored\_message)

知识点：颜色的转义序列

##### 练习题36

求100之内的素数。

分析：定义判断素数的函数，从2循环到100，求出素数

def is\_prime(num):  
 if num < 2:  
 return False  
 for i in range(2, int(num\*\*0.5) + 1):  
 if num % i == 0:  
 return False  
 return True  
  
primes = []  
for num in range(2, 101):  
 if is\_prime(num):  
 primes.append(num)  
  
print(primes)

知识点：函数调用；循环；算法基础

##### 练习题37

对10个数进行排序

分析：从控制台得到10个数，然后使用sort()函数对数组进行排列

# 输入10个数  
numbers = []  
for i in range(10):  
 num = float(input("请输入第{}个数：".format(i+1)))  
 numbers.append(num)  
  
numbers.sort()  
  
# 打印排序后的结果  
print("排序后的结果：", numbers)

知识点：从控制台得到输入；列表排序

##### 练习题38

求一个3\*3矩阵主对角线元素之和。

分析：利用矩阵的索引获得主对角线的元素，然后加在一起

def diagonal\_sum(matrix):  
 n = len(matrix)  
 diagonal\_sum = 0  
 for i in range(n):  
 diagonal\_sum += matrix[i][i]  
 return diagonal\_sum  
  
# 输入3x3矩阵  
matrix = []  
print("请输入3x3矩阵的元素：")  
for i in range(3):  
 row = []  
 for j in range(3):  
 element = int(input("请输入第{}行第{}列的元素：".format(i + 1, j + 1)))  
 row.append(element)  
 matrix.append(row)  
  
# 调用函数计算主对角线元素之和  
sum\_of\_diagonal = diagonal\_sum(matrix)  
  
# 输出结果  
print("主对角线元素之和：", sum\_of\_diagonal)

知识点：矩阵遍历，矩阵索引，函数调用

##### 练习题39

有一个已经排好序的数组。现输入一个数，要求按原来的规律将它插入数组中。

分析：根据大小顺序，找到数字需要插入的数组索引位置，然后使用insert()函数将数字插入到数组中

def insert\_into\_sorted\_array(arr, num):  
 index = 0  
 while index < len(arr) and num > arr[index]:  
 index += 1  
 arr.insert(index, num)  
  
# 输入已排序的数组  
sorted\_array = [1, 3, 5, 7, 9]  
print("已排序的数组：", sorted\_array)  
  
# 输入要插入的数  
number = int(input("请输入要插入的数："))  
  
# 调用函数插入数到数组  
insert\_into\_sorted\_array(sorted\_array, number)  
  
# 输出插入后的数组  
print("插入后的数组：", sorted\_array)

知识点：数组的使用；算法基础

##### 练习题40

将一个数组逆序输出

分析：使用切片将数组逆序

def reverse\_array(arr):  
 reversed\_arr = arr[::-1]  
 return reversed\_arr  
  
# 输入数组  
array = input("请输入数组元素，以空格分隔：").split()  
  
# 调用函数逆序输出数组  
reversed\_array = reverse\_array(array)  
  
# 输出逆序后的数组  
print("逆序输出的数组：", reversed\_array)

知识点：列表切片；函数调用

##### 练习题41

模仿静态变量的用法。

分析：静态变量可以被修改，使用函数进行模仿

def my\_function():  
 if not hasattr(my\_function, 'counter'):  
 my\_function.counter = 0  
 my\_function.counter += 1  
 print("Counter:", my\_function.counter)  
  
# 调用函数多次  
my\_function()  
my\_function()  
my\_function()

知识点：静态变量

##### 练习题42

学习使用auto定义变量的用法

分析：不理解。

##### 练习题43

模仿静态变量(static)另一案例。

分析：应该是区别静态变量和全局变量的使用

class Num:  
 nNum = 1  
 def inc(self):  
 self.nNum += 1  
 print ('nNum = %d' % self.nNum)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 nNum = 2  
 inst = Num()  
 for i in range(3):  
 nNum += 1  
 print ('The num = %d' % nNum)  
 inst.inc()

知识点：静态变量和全局变量的作用域

##### 练习题44

两个 3 行 3 列的矩阵，实现其对应位置的数据相加，并返回一个新矩阵：

分析：双重for循环，遍历两个矩阵，对应位置相加存到一个新的矩阵里

X = [[12, 7, 3],  
 [4, 5, 6],  
 [7, 8, 9]]  
  
Y = [[5, 8, 1],  
 [6, 7, 3],  
 [4, 5, 9]]  
  
result = [[0, 0, 0],  
 [0, 0, 0],  
 [0, 0, 0]]  
  
# 遍历矩阵的行  
for i in range(len(X)):  
 # 遍历矩阵的列  
 for j in range(len(X[0])):  
 result[i][j] = X[i][j] + Y[i][j]  
  
# 打印结果矩阵  
for row in result:  
 print(row)

知识点：双重for循环，矩阵索引

##### 练习题45

统计 1 到 100 之和。

分析：从1遍历到100，将值加在一起

total = 0  
  
for num in range(1, 101):  
 total += num  
  
print("总和:", total)

知识点：for循环

##### 练习题46

求输入数字的平方，如果平方运算后小于 50 则退出

分析：使用input()接受输入，if语句用于判断，break语句用于退出循环

while True:  
 num = int(input("请输入一个数字："))  
 square = num \*\* 2  
  
 print("平方结果:", square)  
  
 if square < 50:  
 break

知识点：判断语句；

##### 练习题47

两个变量值互换。

分析：使用临时变量存储中间值，完成两个变量值的互换

# 初始变量值  
x = 10  
y = 20  
  
print("交换前：")  
print("x =", x)  
print("y =", y)  
  
# 交换变量值  
temp = x  
x = y  
y = temp  
  
print("交换后：")  
print("x =", x)  
print("y =", y)

知识点：基础语法

##### 练习题48

数字比较

分析：利用判断语句和比较运算符完成数字的比较

num1 = 10  
num2 = 5  
  
if num1 > num2:  
 print("num1 大于 num2")  
elif num1 < num2:  
 print("num1 小于 num2")  
else:  
 print("num1 等于 num2")

知识点：判断语句，比较运算符

##### 练习题49

使用lambda来创建匿名函数。

分析：lambad的使用，创建小函数

# 使用 lambda 创建一个接受两个参数并返回它们之和的匿名函数  
add = lambda x, y: x + y  
  
# 调用匿名函数  
result = add(3, 4)  
print(result) # 输出：7

知识点：lambda的使用

##### 练习题50

输出一个随机数。

分析：random模块的使用

import random  
  
random\_number = random.randint(1, 100)  
print(random\_number)

知识点：random模块的使用

##### 练习题51

学习使用按位与 &

分析：二进制与按位与

a = 10 # 二进制表示为 1010  
b = 6 # 二进制表示为 0110  
  
result = a & b  
print(result) # 输出：2

知识点：二进制的运算

##### 练习题52

学习使用按位或 |

分析：二进制与按位或

a = 10 # 二进制表示为 1010  
b = 6 # 二进制表示为 0110  
  
result = a | b  
print(result) # 输出：14

知识点：二进制的运算

##### 练习题53

学习使用按位异或 ^

分析：二进制与按位异或

a = 10 # 二进制表示为 1010  
b = 6 # 二进制表示为 0110  
  
result = a ^ b  
print(result) # 输出：12

知识点：二进制与按位异或

##### 练习题54

取一个整数a从右端开始的4〜7位。

分析：根据题目意思应该是二进制位移，先进行移位将4~7位移动到最右侧，然后与0b1111进行按位与

a = 12345678  
mask = 0b1111 # 用来提取4位的掩码，二进制表示为 1111  
  
result = (a >> 4) & mask  
print(result) # 输出：6

知识点：二进制移位

##### 练习题55

学习使用按位取反~

分析：二进制数在内存中以补码的形式进行存储

a = 10 # 二进制表示为 00001010  
  
result = ~a  
print(result) # 输出：-11

知识点：二进制原码，补码，反码

##### 练习题56

画图，学用circle画圆形

分析：学习调用工具画图

import matplotlib.pyplot as plt  
  
def draw\_circle(radius):  
 circle = plt.Circle((0, 0), radius, fill=False)  
 fig, ax = plt.subplots()  
 ax.set\_aspect('equal')  
 ax.add\_artist(circle)  
 plt.xlim(-radius-1, radius+1)  
 plt.ylim(-radius-1, radius+1)  
 plt.gca().set\_aspect('equal', adjustable='box')  
 plt.grid(True)  
 plt.show()  
  
# 画一个半径为5的圆形  
draw\_circle(5)

知识点：matplotlib画图库

##### 练习题57

画图，学用line画直线。

分析：学习调用工具画图

import matplotlib.pyplot as plt  
  
def draw\_line(x1, y1, x2, y2):  
 plt.plot([x1, x2], [y1, y2], 'b-')  
 plt.xlabel('X')  
 plt.ylabel('Y')  
 plt.title('Line')  
 plt.grid(True)  
 plt.show()  
  
# 绘制直线从点(1, 2)到点(5, 7)  
draw\_line(1, 2, 5, 7)

知识点：matplotlib画图库

##### 练习题58

画图，学用rectangle画方形

分析：学习使用工具画图

import matplotlib.pyplot as plt  
from matplotlib.patches import Rectangle  
  
def draw\_rectangle(x, y, width, height):  
 fig, ax = plt.subplots()  
 rect = Rectangle((x, y), width, height, linewidth=1, edgecolor='b', facecolor='none')  
 ax.add\_patch(rect)  
 ax.set\_aspect('equal')  
 ax.set\_xlim(x-1, x+width+1)  
 ax.set\_ylim(y-1, y+height+1)  
 ax.grid(True)  
 plt.xlabel('X')  
 plt.ylabel('Y')  
 plt.title('Rectangle')  
 plt.show()  
  
# 绘制一个位于(1, 2)的方形，宽度为3，高度为4  
draw\_rectangle(1, 2, 3, 4)

知识点：matplotlib画图库

##### 练习题59

画图，综合例子。**程序分析：**利用for循环控制100-999个数，每个数分解出个位，十位，百位

分析：不理解

##### 练习题60

计算字符串长度。

分析：len()函数

string = "Hello, world!"  
length = len(string)  
print(length) # 输出：13

知识点：字符串

##### 练习题61

打印出杨辉三角形

分析：使用循环和列表来打印出杨辉三角

def print\_pascal\_triangle(rows):  
 triangle = []  
 for i in range(rows):  
 row = []  
 for j in range(i + 1):  
 if j == 0 or j == i:  
 row.append(1)  
 else:  
 value = triangle[i - 1][j - 1] + triangle[i - 1][j]  
 row.append(value)  
 triangle.append(row)  
  
 for row in triangle:  
 print(' '.join(str(num) for num in row))  
  
  
# 打印5行的杨辉三角形  
print\_pascal\_triangle(5)

知识点：嵌套循环

##### 练习题62

查找字符串。

分析：字符串操作，字符串find()和index()函数的使用

string = "Hello, world!"  
substring = "world"  
  
index = string.find(substring)  
if index != -1:  
 print("Substring found at index", index)  
else:  
 print("Substring not found")

string = "Hello, world!"  
substring = "world"  
  
try:  
 index = string.index(substring)  
 print("Substring found at index", index)  
except ValueError:  
 print("Substring not found")

知识点：字符串查找

##### 练习题63

画椭圆

分析：调用库，画出椭圆

import matplotlib.pyplot as plt  
from matplotlib.patches import Ellipse  
  
def draw\_ellipse(center\_x, center\_y, width, height):  
 fig, ax = plt.subplots()  
 ellipse = Ellipse((center\_x, center\_y), width, height, edgecolor='b', facecolor='none')  
 ax.add\_patch(ellipse)  
 ax.set\_aspect('equal')  
 ax.set\_xlim(center\_x - width - 1, center\_x + width + 1)  
 ax.set\_ylim(center\_y - height - 1, center\_y + height + 1)  
 ax.grid(True)  
 plt.xlabel('X')  
 plt.ylabel('Y')  
 plt.title('Ellipse')  
 plt.show()  
  
# 绘制一个位于(2, 3)的椭圆，宽度为4，高度为2  
draw\_ellipse(2, 3, 4, 2)

知识点：matplotlib的Ellipse类

##### 练习题64

利用ellipse 和 rectangle 画图

分析：使用matplotlib画图

import matplotlib.pyplot as plt  
from matplotlib.patches import Ellipse, Rectangle  
  
def draw\_shapes():  
 fig, ax = plt.subplots()  
  
 # 绘制椭圆  
 ellipse = Ellipse((2, 3), 4, 2, edgecolor='b', facecolor='none')  
 ax.add\_patch(ellipse)  
  
 # 绘制矩形  
 rectangle = Rectangle((1, 1), 3, 2, edgecolor='r', facecolor='none')  
 ax.add\_patch(rectangle)  
  
 ax.set\_aspect('equal')  
 ax.set\_xlim(0, 5)  
 ax.set\_ylim(0, 5)  
 ax.grid(True)  
 plt.xlabel('X')  
 plt.ylabel('Y')  
 plt.title('Shapes')  
 plt.show()  
  
# 调用函数绘制图形  
draw\_shapes()

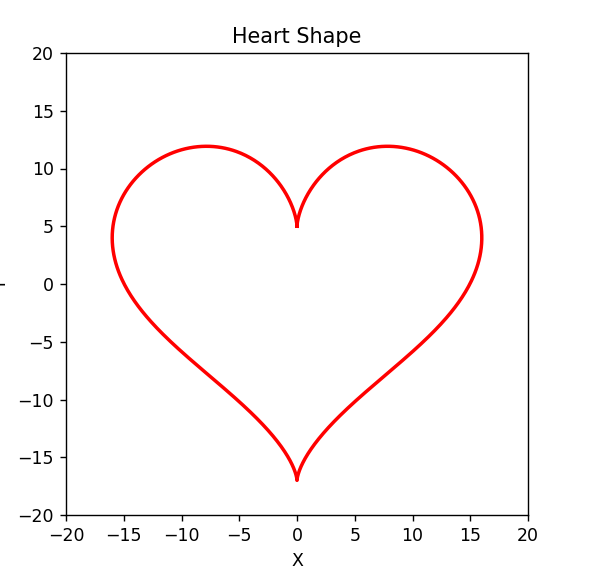
知识点：matplotlib库的使用

##### 练习题65

一个最优美的图案

分析：画出优美的图案

import numpy as np  
import matplotlib.pyplot as plt  
  
def draw\_heart():  
 t = np.linspace(0, 2 \* np.pi, 1000)  
 x = 16 \* np.sin(t) \*\* 3  
 y = 13 \* np.cos(t) - 5 \* np.cos(2 \* t) - 2 \* np.cos(3 \* t) - np.cos(4 \* t)  
  
 fig, ax = plt.subplots()  
 ax.plot(x, y, color='red', linewidth=2)  
 ax.set\_aspect('equal')  
 ax.set\_xlim(-20, 20)  
 ax.set\_ylim(-20, 20)  
  
 plt.xlabel('X')  
 plt.ylabel('Y')  
 plt.title('Heart Shape')  
 plt.show()  
  
# 绘制心形  
draw\_heart()



##### 练习题66

输入3个数a,b,c，按大小顺序输出

分析：使用input()得到三个数，使用sort()函数进行排序，然后输出；或者直接进行排序

def sort\_numbers(a, b, c):  
 if a <= b <= c:  
 return a, b, c  
 elif a <= c <= b:  
 return a, c, b  
 elif b <= a <= c:  
 return b, a, c  
 elif b <= c <= a:  
 return b, c, a  
 elif c <= a <= b:  
 return c, a, b  
 else:  
 return c, b, a  
  
# 输入三个数  
a = int(input("请输入第一个数："))  
b = int(input("请输入第二个数："))  
c = int(input("请输入第三个数："))  
  
# 调用函数进行排序  
sorted\_numbers = sort\_numbers(a, b, c)  
  
# 输出排序结果  
print("按大小顺序输出：", sorted\_numbers)

知识点：排序

##### 练习题67

输入数组，最大的与第一个元素交换，最小的与最后一个元素交换，输出数组

分析：查找最大、最小的元素得到索引，完成元素的交换

def swap\_max\_min(arr):  
 if len(arr) < 2:  
 return arr  
  
 max\_index = arr.index(max(arr))  
 min\_index = arr.index(min(arr))  
  
 arr[0], arr[max\_index] = arr[max\_index], arr[0]  
 arr[-1], arr[min\_index] = arr[min\_index], arr[-1]  
  
 return arr  
  
# 输入数组  
input\_str = input("请输入一个数组，元素之间用空格分隔：")  
arr = list(map(int, input\_str.split()))  
  
# 调用函数进行交换  
swapped\_arr = swap\_max\_min(arr)  
  
# 输出交换后的数组  
print("交换后的数组：", swapped\_arr)

知识点：查找最大元素、最小元素位置，进行交换

##### 练习题68

有 n 个整数，使其前面各数顺序向后移 m 个位置，最后 m 个数变成最前面的 m 个数

分析：使用切片获取前n-m项和后m项，拼接在一起

def shift\_array(arr, m):  
 n = len(arr)  
 m = m % n # 处理m大于n的情况  
  
 # 将后面m个元素移到前面  
 shifted\_arr = arr[n-m:] + arr[:n-m]  
  
 return shifted\_arr  
  
# 输入数组  
input\_str = input("请输入一个数组，元素之间用空格分隔：")  
arr = list(map(int, input\_str.split()))  
  
# 输入移动的位置  
m = int(input("请输入要向后移动的位置数："))  
  
# 调用函数进行移动操作  
shifted\_arr = shift\_array(arr, m)  
  
# 输出移动后的数组  
print("移动后的数组：", shifted\_arr)

知识点：列表操作

##### 练习题69

有n个人围成一圈，顺序排号。从第一个人开始报数（从1到3报数），凡报到3的人退出圈子，问最后留下的是原来第几号的那位

分析：约瑟夫环问题，使用循环列表来解决

class Node:  
 def \_\_init\_\_(self, value):  
 self.value = value  
 self.next = None  
  
  
def find\_last\_person(n):  
 # 创建循环链表  
 head = Node(1)  
 current = head  
 for i in range(2, n + 1):  
 new\_node = Node(i)  
 current.next = new\_node  
 current = new\_node  
 current.next = head  
  
 # 开始报数并删除节点  
 current = head  
 while current.next != current:  
 # 找到要删除的节点的前一个节点  
 for \_ in range(1, 3):  
 prev = current  
 current = current.next  
 # 删除节点  
 prev.next = current.next  
 current = prev.next  
  
 return current.value  
  
  
# 输入总人数  
n = int(input("请输入总人数："))  
  
# 调用函数找到最后留下的人的编号  
last\_person = find\_last\_person(n)  
  
# 输出结果  
print("最后留下的人的编号：", last\_person)

知识点：数据结构，算法基础

##### 练习题70

写一个函数，求一个字符串的长度，在main函数中输入字符串，并输出其长度

分析：遍历字符串，使用计数器

def get\_string\_length(string):  
 length = 0  
 for \_ in string:  
 length += 1  
 return length  
  
def main():  
 # 输入字符串  
 input\_str = input("请输入一个字符串：")  
  
 # 调用函数计算字符串长度  
 length = get\_string\_length(input\_str)  
  
 # 输出字符串长度  
 print("字符串长度：", length)  
  
# 调用main函数  
main()

知识点：字符串操作

##### 练习题71

编写input()和output()函数输入，输出5个学生的数据记录

分析：输入和输出的使用

def input\_students():  
 students = []  
 for i in range(5):  
 print(f"请输入第{i+1}个学生的数据记录：")  
 name = input("姓名：")  
 age = input("年龄：")  
 grade = input("成绩：")  
 student = {  
 "姓名": name,  
 "年龄": age,  
 "成绩": grade  
 }  
 students.append(student)  
 print()  
 return students  
  
def output\_students(students):  
 print("学生数据记录：")  
 for i, student in enumerate(students):  
 print(f"第{i+1}个学生：")  
 print(f"姓名：{student['姓名']}")  
 print(f"年龄：{student['年龄']}")  
 print(f"成绩：{student['成绩']}")  
 print()  
  
def main():  
 students = input\_students()  
 output\_students(students)  
  
# 调用main函数  
main()

知识点：输入、输出

##### 练习题72

创建一个链表。

分析：创建链表，实现增删改查

class Node:  
 def \_\_init\_\_(self, data):  
 self.data = data  
 self.next = None  
  
class LinkedList:  
 def \_\_init\_\_(self):  
 self.head = None  
  
 def append(self, data):  
 new\_node = Node(data)  
 if self.head is None:  
 self.head = new\_node  
 else:  
 current = self.head  
 while current.next is not None:  
 current = current.next  
 current.next = new\_node  
  
 def display(self):  
 if self.head is None:  
 print("链表为空")  
 else:  
 current = self.head  
 while current is not None:  
 print(current.data, end=" ")  
 current = current.next  
 print()  
  
# 创建链表  
linked\_list = LinkedList()  
  
# 添加节点  
linked\_list.append(1)  
linked\_list.append(2)  
linked\_list.append(3)  
linked\_list.append(4)  
linked\_list.append(5)  
  
# 显示链表  
linked\_list.display()

知识点：数据结构链表

##### 练习题73

反向输出一个链表

分析：先创建一个链表，可以使用递归来进行链表的逆序输出

class Node:  
 def \_\_init\_\_(self, data):  
 self.data = data  
 self.next = None  
  
class LinkedList:  
 def \_\_init\_\_(self):  
 self.head = None  
  
 def append(self, data):  
 new\_node = Node(data)  
 if self.head is None:  
 self.head = new\_node  
 else:  
 current = self.head  
 while current.next is not None:  
 current = current.next  
 current.next = new\_node  
  
 def reverse\_display\_recursive(self, node):  
 if node is None:  
 return  
 self.reverse\_display\_recursive(node.next)  
 print(node.data, end=" ")  
  
 def reverse\_display(self):  
 if self.head is None:  
 print("链表为空")  
 else:  
 self.reverse\_display\_recursive(self.head)  
  
# 创建链表  
linked\_list = LinkedList()  
  
# 添加节点  
linked\_list.append(1)  
linked\_list.append(2)  
linked\_list.append(3)  
linked\_list.append(4)  
linked\_list.append(5)  
  
# 反向输出链表节点值  
print("反向输出链表节点值：")  
linked\_list.reverse\_display()

知识点：数据结构与算法

##### 练习题74

列表排序及连接

分析：使用sort()函数进行列表的排序，然后使用“+”对列表进行连接

# 列表排序  
numbers = [5, 2, 4, 1, 3]  
sorted\_numbers = sorted(numbers) # 使用sorted()函数对列表进行排序  
numbers.sort() # 使用sort()方法对列表进行原地排序  
  
print("排序后的列表：")  
print(sorted\_numbers)  
print(numbers)  
print()  
  
# 列表连接  
list1 = [1, 2, 3]  
list2 = [4, 5, 6]  
concatenated\_list = list1 + list2 # 使用+运算符连接列表  
  
print("连接后的列表：")  
print(concatenated\_list)

知识点：列表操作

##### 练习题75

放松一下，算一道简单的题目

#!/usr/bin/python  
# -\*- coding: UTF-8 -\*-  
   
if \_\_name\_\_ == '\_\_main\_\_':  
 for i in range(5):  
 n = 0  
 if i != 1: n += 1  
 if i == 3: n += 1  
 if i == 4: n += 1  
 if i != 4: n += 1  
 if n == 3: print 64 + i

输出结果为67

##### 练习题76

编写一个函数，输入n为偶数时，调用函数求1/2+1/4+...+1/n,当输入n为奇数时，调用函数1/1+1/3+...+1/n

分析：使用input()函数获取输入，判断是否是偶数，然后调用函数

def sum\_series(n):  
 if n % 2 == 0: # 当n为偶数  
 result = 0  
 for i in range(2, n + 1, 2):  
 result += 1 / i  
 return result  
 else: # 当n为奇数  
 result = 0  
 for i in range(1, n + 1, 2):  
 result += 1 / i  
 return result  
  
# 测试  
n = int(input("请输入一个整数n："))  
if n % 2 == 0:  
 series\_type = "1/2 + 1/4 + ... + 1/n"  
else:  
 series\_type = "1/1 + 1/3 + ... + 1/n"  
  
result = sum\_series(n)  
print(f"{series\_type} 的和为：{result}")

知识点：判断语句；函数调用

##### 练习题77

循环输出列表

分析：列表的遍历

# 使用 for 循环输出列表元素  
def print\_list\_elements\_for\_loop(lst):  
 for item in lst:  
 print(item)  
  
# 使用 while 循环输出列表元素  
def print\_list\_elements\_while\_loop(lst):  
 i = 0  
 while i < len(lst):  
 print(lst[i])  
 i += 1  
  
# 测试  
my\_list = [1, 2, 3, 4, 5]  
  
print("使用 for 循环输出列表元素：")  
print\_list\_elements\_for\_loop(my\_list)  
  
print("\n使用 while 循环输出列表元素：")  
print\_list\_elements\_while\_loop(my\_list)

知识点：使用for循环和while循环遍历列表

##### 练习题78

找到年龄最大的人，并输出。

分析：使用字典将人的年龄和性病进行绑定，然后遍历字典，找到年龄最大的人

def find\_oldest\_person(people):  
 oldest\_age = 0  
 oldest\_person = ""  
  
 for person, age in people.items():  
 if age > oldest\_age:  
 oldest\_age = age  
 oldest\_person = person  
  
 return oldest\_person  
  
# 测试  
people = {  
 "Alice": 25,  
 "Bob": 30,  
 "Charlie": 35,  
 "David": 40  
}  
  
oldest\_person = find\_oldest\_person(people)  
print(f"年龄最大的人是：{oldest\_person}")

知识点：字典的遍历

##### 练习题79

字符串排序

分析：将字符串使用sorted()函数或者sort()函数进行排序

# 使用 sorted() 函数对字符串进行排序  
string = "openai"  
sorted\_string = ''.join(sorted(string))  
print("使用 sorted() 函数排序后的字符串：", sorted\_string)  
  
# 使用字符串的 sort() 方法对字符串进行排序  
string\_list = list(string)  
string\_list.sort()  
sorted\_string = ''.join(string\_list)  
print("使用 sort() 方法排序后的字符串：", sorted\_string)

知识点：字符串排序

##### 练习题80

海滩上有一堆桃子，五只猴子来分。第一只猴子把这堆桃子平均分为五份，多了一个，这只猴子把多的一个扔入海中，拿走了一份。第二只猴子把剩下的桃子又平均分成五份，又多了一个，它同样把多的一个扔入海中，拿走了一份，第三、第四、第五只猴子都是这样做的，问海滩上原来最少有多少个桃子？

分析：从1开始遍历到1000，找到满足上面条件的桃子数量即可

def find\_minimum\_peaches():  
 for x in range(1, 1000):  
 remaining\_peaches = x  
 for \_ in range(5):  
 if remaining\_peaches % 5 != 1:  
 break  
 remaining\_peaches = (remaining\_peaches - 1) // 5 \* 4  
 else:  
 return x  
 return -1  
  
# 测试  
minimum\_peaches = find\_minimum\_peaches()  
print("海滩上原来最少有", minimum\_peaches, "个桃子")

知识点：模拟

##### 练习题81

809\*??=800\*??+9\*?? 其中??代表的两位数, 809\*??为四位数，8\*??的结果为两位数，9\*??的结果为3位数。求??代表的两位数，及809\*??后的结果。

分析：穷举的方式来解决这个问题，根据关系式找到满足条件的？？

for x in range(10, 100):  
 product = 809 \* x  
 if product >= 1000 and product < 10000:  
 multiple\_8 = 8 \* x  
 multiple\_9 = 9 \* x  
 if multiple\_8 >= 10 and multiple\_8 < 100 and multiple\_9 >= 100 and multiple\_9 < 1000:  
 if product == 800 \* x + 9 \* x:  
 print("?? 代表的两位数为:", x)  
 print("809 \* ?? 的结果为:", product)  
 break

知识点：模拟；遍历

##### 练习题82

八进制转换为十进制

分析：自己编写代码转换，或者直接使用int函数进行转换

octal\_number = "53" # 八进制数，例如 53  
decimal\_number = int(octal\_number, 8)  
print("转换后的十进制数：", decimal\_number)

知识点：进制的转换

##### 练习题83

求0—7所能组成的奇数个数

分析：组成1位数是4个；组成2位数是7\*4个；组成3位数是7\*8\*4个；组成4位数是7\*8\*8\*4个。

sum = 4  
s = 4  
for j in range(2,9):  
 print sum  
 if j <= 2:  
 s \*= 7  
 else:  
 s \*= 8  
 sum += s  
print 'sum = %d' % sum

知识点：排列组合

##### 练习题84

连接字符串

分析：使用“+”或者join()函数来连接字符串

str1 = "Hello"  
str2 = "World"  
result = str1 + " " + str2  
print(result)

str\_list = ["Hello", "World"]  
result = " ".join(str\_list)  
print(result)

知识点：字符串连接

##### 练习题85

输入一个奇数，然后判断最少几个 9 除于该数的结果为整数

分析：使用一个计数器来判断使用了多少个，然后用算术表达式判断是否能被整除

def count\_nines\_divisible\_by\_odd\_number(odd\_number):  
 nines = 9  
 count = 1  
  
 while nines % odd\_number != 0:  
 nines = nines \* 10 + 9  
 count += 1  
  
 return count  
  
# 从用户输入获取奇数  
odd\_number = int(input("请输入一个奇数："))  
  
result = count\_nines\_divisible\_by\_odd\_number(odd\_number)  
print("最少需要", result, "个9才能整除该奇数")

知识点：函数编写，算术表达式

##### 练习题86

两个字符串连接程序

分析：和练习题84一样，质量不是很高

str1 = "Hello"  
str2 = "World"  
result = str1 + str2  
print(result)

str\_list = ["Hello", "World"]  
result = "".join(str\_list)  
print(result)

知识点：字符串连接

##### 练习题87

回答结果（结构体变量传递）

分析：不理解

##### 练习题88

读取7个数（1—50）的整数值，每读取一个值，程序打印出该值个数的＊

分析：标准输入，然后打印对应的"\*"

for i in range(7):  
 num = int(input("请输入一个整数（1-50）："))  
 if num < 1 or num > 50:  
 print("输入的整数超出范围，请重新输入。")  
 continue  
 print("\*" \* num)

知识点：标准输入

##### 练习题89

某个公司采用公用电话传递数据，数据是四位的整数，在传递过程中是加密的，加密规则如下：每位数字都加上5,然后用和除以10的余数代替该数字，再将第一位和第四位交换，第二位和第三位交换

分析：根据题意进行模拟即可

def encrypt(number):  
 # 将每位数字加上5，并用和除以10的余数代替该数字  
 encrypted\_number = ""  
 for digit in str(number):  
 encrypted\_digit = (int(digit) + 5) % 10  
 encrypted\_number += str(encrypted\_digit)  
  
 # 将第一位和第四位交换  
 encrypted\_number = encrypted\_number[3] + encrypted\_number[1] + encrypted\_number[2] + encrypted\_number[0]  
  
 return int(encrypted\_number)  
  
  
def decrypt(number):  
 # 将第一位和第四位交换  
 number = str(number)  
 decrypted\_number = number[3] + number[1] + number[2] + number[0]  
  
 # 将每位数字减去5，并用和除以10的余数代替该数字  
 decrypted\_number = ""  
 for digit in number:  
 decrypted\_digit = (int(digit) - 5) % 10  
 decrypted\_number += str(decrypted\_digit)  
  
 return int(decrypted\_number)  
  
  
# 加密示例  
original\_number = 1234  
encrypted\_number = encrypt(original\_number)  
print("加密后的数据:", encrypted\_number)  
  
# 解密示例  
decrypted\_number = decrypt(encrypted\_number)  
print("解密后的数据:", decrypted\_number)

知识点：基础代码编写

##### 练习题90

列表使用实例

分析：列表的使用，函数的使用

fruits = ['apple', 'banana', 'orange']  
print(fruits[0]) # 输出：'apple'  
fruits[1] = 'grape'  
print(fruits) # 输出：['apple', 'grape', 'orange']  
fruits.append('kiwi')  
print(fruits) # 输出：['apple', 'grape', 'orange', 'kiwi']  
print(len(fruits)) # 输出：4  
print(fruits[1:3]) # 输出：['grape', 'orange']  
for fruit in fruits:  
 print(fruit)  
fruits.sort()  
print(fruits) # 输出：['apple', 'grape', 'kiwi', 'orange']

知识点：列表的基本操作

##### 练习题91

时间函数举例1

分析：时间模块的调用

if \_\_name\_\_ == '\_\_main\_\_':  
 import time  
 print(time.ctime(time.time()))  
 print(time.asctime(time.localtime(time.time())))  
 print(time.asctime(time.gmtime(time.time())))

知识点：时间模块的调用

##### 练习题92

时间函数举例2

分析：时间模块的调用

if \_\_name\_\_ == '\_\_main\_\_':  
 import time  
  
 start = time.time()  
 for i in range(3000):  
 print(i)  
 end = time.time()  
  
 print(end - start)

知识点：时间模块的调用

##### 练习题93

时间函数举例3

分析：时间模块的调用

if \_\_name\_\_ == '\_\_main\_\_':  
 import time  
 start = time.clock()  
 for i in range(10000):  
 print(i)  
 end = time.clock()  
 print('different is %6.3f' % (end - start))

知识点：时间模块的调用

##### 练习题94

时间函数举例4,一个猜数游戏，判断一个人反应快慢

分析：本题略，python解释器版本不兼容

##### 练习题95

字符串日期转换为易读的日期格式

分析：获得datetime对象，将datetime对象转换为易读的格式

from datetime import datetime  
  
# 输入的字符串日期  
date\_string = "2022-01-31"  
  
# 将字符串日期转换为 datetime 对象  
date\_object = datetime.strptime(date\_string, "%Y-%m-%d")  
  
# 将 datetime 对象转换为易读的日期格式  
readable\_date = date\_object.strftime("%B %d, %Y")  
  
# 打印转换后的日期  
print(readable\_date)

知识点：datetime模块调用，格式化日期

##### 练习题96

计算字符串中子串出现的次数。

分析：使用python中的count()函数

# 输入的字符串  
string = "Hello, hello, hello, how are you?"  
  
# 要搜索的子串  
substring = "hello"  
  
# 使用 count() 方法计算子串出现的次数（不区分大小写）  
count = string.lower().count(substring.lower())  
  
# 打印结果  
print("子串出现的次数:", count)

知识点：字符串的使用

##### 练习题97

从键盘输入一些字符，逐个把它们写到磁盘文件上，直到输入一个 **#** 为止。

分析：使用open函数打开文件，使用write()函数进行写入文件

# 指定要写入的文件名  
filename = "output.txt"  
  
# 打开文件以写入模式  
with open(filename, "w") as file:  
 while True:  
 # 从键盘读取输入字符  
 char = input("请输入一个字符：")  
  
 # 如果输入字符为 "#"，则退出循环  
 if char == "#":  
 break  
  
 # 将输入字符写入文件  
 file.write(char)  
  
# 输出完成消息  
print("字符已写入文件:", filename)

知识点：文件的输入

##### 练习题98

键盘输入一个字符串，将小写字母全部转换成大写字母，然后输出到一个磁盘文件"test"中保存

分析：使用字符串的函数将字母转换为大写，然后使用文件输入将字符串保存到“test”中

# 获取从键盘输入的字符串  
input\_string = input("请输入一个字符串：")  
  
# 将字符串中的小写字母转换为大写字母  
output\_string = input\_string.upper()  
  
# 指定要保存的文件名  
filename = "test.txt"  
  
# 将结果写入磁盘文件  
with open(filename, "w") as file:  
 file.write(output\_string)  
  
# 输出完成消息  
print("转换后的字符串已保存到文件:", filename)

知识点：文件的输入

##### 练习题99

有两个磁盘文件A和B,各存放一行字母,要求把这两个文件中的信息合并(按字母顺序排列), 输出到一个新文件C中

分析：文件的基本操作，打开读取文件，然后写入文件

# 读取文件A  
with open('file\_a.txt', 'r') as file\_a:  
 content\_a = file\_a.read().strip()  
  
# 读取文件B  
with open('file\_b.txt', 'r') as file\_b:  
 content\_b = file\_b.read().strip()  
  
# 合并并按字母顺序排序  
merged\_content = ''.join(sorted(content\_a + content\_b))  
  
# 写入新文件C  
with open('file\_c.txt', 'w') as file\_c:  
 file\_c.write(merged\_content)

知识点：文件的读取和写入

##### 练习题100

列表转换为字典

分析：使用dict()函数将列表转换为字典

my\_list = [('a', 1), ('b', 2), ('c', 3)]  
my\_dict = dict(my\_list)  
print(my\_dict)

知识点：列表转为字典