第一种计算：两个数值做比较

一共有6种比价方式：== ！== > < >= <=

while True:

print('上供一对童男童女')

t = input('孙悟空来了吗\n')

if t == '来了':

break

print('孙悟空制服了鲤鱼精，陈家庄再也不用上供童男童女了')

while True:

q1 = input('第一问：你一生之中，在什么地方最是快乐逍遥？')

if q1 != '黑暗的冰窖':

continue

print('答对了，下面是第二问：')

q2 = input('你生平最爱之人，叫什么名字？')

if q2 != '梦姑':

continue

print('答对了，下面是第三问：')

q3 = input('你最爱的这个人相貌如何？')

if q3 == '不知道':

break

print('都答对了，你是虚竹。')

import time,random

player\_victory = 0

enemy\_victory = 0

for i in range(1,4):

time.sleep(2) # 让局与局之间有较明显的有时间间隔

print(' \n——————现在是第'+str(i)+'局——————') # 作为局的标记

player\_life = random.randint(100,150)

player\_attack = random.randint(30,50)

enemy\_life = random.randint(100,150)

enemy\_attack = random.randint(30,50)

# 展示双方角色的属性

print('【玩家】\n'+'血量：'+str(player\_life)+'\n攻击：'+str(player\_attack))

print('------------------------')

time.sleep(1)

print('【敌人】\n'+'血量：'+str(enemy\_life)+'\n攻击：'+str(enemy\_attack))

print('------------------------')

time.sleep(1)

# 双方PK

while player\_life > 0 and enemy\_life > 0:

player\_life = player\_life - enemy\_attack

enemy\_life = enemy\_life - player\_attack

print('你发起了攻击，【敌人】剩余血量'+str(enemy\_life))

print('敌人向你发起了攻击，【玩家】剩余血量'+str(player\_life))

print('-----------------------')

time.sleep(1.5)

#打印最终战果

if player\_life > 0 and enemy\_life <= 0:

player\_victory += 1

print('敌人死翘翘了，你赢了！')

elif player\_life <= 0 and enemy\_life > 0:

enemy\_victory += 1

print('悲催，敌人把你干掉了！')

else:

print('哎呀，你和敌人同归于尽了！')

if player\_victory > enemy\_victory :

time.sleep(1)

print('【最终结果：你赢了！】')

elif enemy\_victory > player\_victory:

print('【最终结果：你输了！】')

else:

print('【最终结果：平局！】')

#以下是原始代码

for i in range(1,3):

print('%d X %d = %d' % (i,2,i\*2),end=',')

for i in range(1,4):

print('%d X %d = %d' % (i,3,i\*3),end=',')

for i in range(1,3):

print('%d X %d = %d' % (i,2,i\*2),end = ' ')

print('') #用来换行

for i in range(1,4):

print('%d X %d = %d' % (i,3,i\*3),end = ' ')

for i in range(1,10):

for j in range(1,i+1):

print( '%d X %d = %d' % (j,i,i\*j),end = ' ' )

print(' ')

list1 = [91, 95, 97, 99]

list2 = [92, 93, 96, 98]

# 把 A 组成绩赋值给一个新列表，用来存合并的成绩——这个细节要注意！

list3 = list1.copy()

list3.extend(list2)

print(list3)

list3.sort()

print(list3)

# 函数名：1. 名字最好能体现函数的功能，一般用小写字母和单下划线、数字等组合

# 2. 不可与内置函数重名（内置函数不需要定义即可直接使用）

def math(x):

# 参数：根据函数功能，括号里可以有多个参数，也可以不带参数，命名规则与函数名相同

# 规范：括号是英文括号，后面的冒号不能丢

y = 3\*x + 5

# 函数体：函数的执行过程，体现函数功能的语句，要缩进，一般是四个空格

return y

# return语句：后面可以接多种数据类型，如果函数不需要返回值的话，可以省略

# 提示：x \*\* y 表示 x的y次幂

def math(x):

y = x \*\* 2 + x

return y

a=math(10)

print(a)

def menu(appetizer, course):

print('一份开胃菜：' + appetizer)

print('一份主食：' + course)

menu('话梅花生','牛肉拉面')

def math(x):

y= x\*\*2+x

return y

n=math(3)

print(n)

def my\_len(words):

counter = 0

for i in words:

counter = counter + 1

return counter

a = '三根皮带，四斤大豆'

print(my\_len(a))

#等价于print(my\_len('三根皮带，四斤大豆'))

#直接运行代码即可

def opening():

print('总有一种味道能温暖你～')

print('深夜食堂正式开业啦！')

print('欢迎来自五湖四海的你前来品尝!')

opening()

def menu(appetizer, course):

print('一份开胃菜：' + appetizer)

print('一份主食：' + course + '\n')

#还记得转义字符\n吧，表示换行

menu('牛肉拉面','话梅花生')

menu('话梅花生','牛肉拉面')

#如果采用下面这种形式传递，就不需要理会参数位置

menu(course = '牛肉拉面', appetizer = '话梅花生')

def menu(appetizer, course, dessert = '绿豆沙'):

print('一份开胃菜：' + appetizer)

print('一份主食：' + course)

print('一份甜品：' + dessert)

menu('话梅花生','牛肉拉面')

#因为已经默认将'绿豆沙'传递给dessert，调用时无须再传递。

def menu(\*barbeque):

for i in barbeque:

print('一份烤串：' + i)

menu('烤香肠', '烤肉丸')

menu('烤鸡翅', '烤茄子', '烤玉米')

# 不定长参数可以接收任意数量的值

print('金枪鱼', '三文鱼', '鲷鱼')

print('金枪鱼', '三文鱼', '鲷鱼', sep = '+')

# sep控制多个值之间的分隔符，默认是空格

print('金枪鱼', '三文鱼', '鲷鱼', sep = '+', end = '=?')

# end控制打印结果的结尾，默认是换行)

import random

#引入random模块

appetizer = ['话梅花生','拍黄瓜','凉拌三丝']

def coupon(money):

if money < 5:

a = random.choice(appetizer)

return a

elif 5 <= money < 10:

b = random.choice (appetizer)

return b, '溏心蛋'

print(coupon(3))

print(coupon(6))

import random

appetizer = ['话梅花生','拍黄瓜','凉拌三丝']

def coupon(money):

if money < 5:

a = random.choice(appetizer)

return a

elif 5 < money <10:

b = random.choice (appetizer)

return b, '溏心蛋'

dish, egg = coupon (7)

# 元组的两个元素分别赋值给变量dish和egg

print(dish)

print(egg)

rent = 3000

def cost():

utilities = int(input('请输入本月的水电费用'))

food\_cost = int(input('请输入本月的食材费用'))

variable\_cost = utilities + food\_cost

print('本月的变动成本是' + str(variable\_cost))

return variable\_cost

def sum\_cost():

sum = rent + cost()

print('本月的总成本是' + '%d' %(sum))

sum\_cost()

def div(num1, num2):

growth = (num1 - num2) / num2

percent = str(growth \* 100) + '%'

return percent

def warning():

print('Error: 你确定上个月一毛钱都不赚不亏吗？')

def main():

while True:

num1 = float(input('请输入本月所获利润'))

num2 = float(input('请输入上月所获利润'))

if num2 == 0:

warning()

else:

print('本月的利润增长率：' + div(num1,num2))

break

main()

def bonus(month):

if month < 6:

money = 500

return money

elif 6 <= month <= 12:

money = 120 \* month

return money

else:

money = 180 \* month

return money

def info(name, month):

gain = bonus(month)

print('%s来了%s个月，获得奖金%s元' % (name, month, gain))

info('大聪',14)

import math

# 人力计算

def estimated\_number(size,time):

number = math.ceil(size \* 80 / time)

print('项目大小为%.1f个标准项目，如果需要在%.1f个工时完成，则需要人力数量为：%d人' %(size,time,number))

# 调用人力计算函数

estimated\_number(1,60)

import math

def number(size,time):

number=math.ceil(size\*80/time)

print("项目大小为%.1f个标准项目，如果需要在%.1f个工时完成，则需要人力数量为：%d人" %(size,time,number))

number(1,60)

import math

# 人力计算

def estimated\_number(size,time):

if (size \* 80 % time) != 0:

number = int(size \* 80 / time) + 1

else:

number = size \* 80 / time

print('项目大小为%.1f个标准项目，如果需要在%.1f个工时完成，则需要人力数量为：%d人' %(size,time,number))

# 调用人力计算函数

estimated\_number(1,60)

import math

# 工时计算

def estimated\_time(size,number):

time = size \* 80 / number

print('项目大小为%.1f个标准项目，使用%d个人力完成，则需要工时数量为：%.1f个' %(size,number,time))

# 人力计算

def estimated\_number(size,time):

number = math.ceil(size \* 80 / time)

print('项目大小为%.1f个标准项目，如果需要在%.1f个工时完成，则需要人力数量为：%d人' %(size,time,number))

# 调用工时计算函数

estimated\_time(1.5,2)

# 调用人力计算函数

estimated\_number(1,60)

import math

# 为函数设置了三个参数，并都带有默认参数）

def estimated(size=1,number=None,time=None):

# 人力计算：如果参数中填了时间，没填人数，就计算人力

if (number == None) and (time != None):

number = math.ceil(size \* 80 / time)

print('项目大小为%.1f个标准项目，如果需要在%.1f个工时完成，则需要人力数量为：%d人' %(size,time,number))

# 工时计算：如果参数中填了人数，没填时间，就计算工时

elif (number != None) and (time == None):

time = size \* 80 / number

print('项目大小为%.1f个标准项目，使用%d个人力完成，则需要工时数量为：%.1f个' %(size,number,time))

# 调用函数的时候，传递两个参数，会自动计算出第三个参数

estimated(size=1.5,number=2)

estimated(size=0.5,time=20.0)

import math

def estimated(size=1,number=None,time=None):

if (number == None) and (time != None):

number = math.ceil(size \* 80 / time)

print('项目大小为%.1f个标准项目，如果需要在%.1f个工时完成，则需要人力数量为：%d人' %(size,time,number))

elif (number != None) and (time == None):

time = size \* 80 / number

print('项目大小为%.1f个标准项目，使用%d个人力完成，则需要工时数量为：%.1f个' %(size,number,time))

choice = input('请选择计算类型：（1-人力计算，2-工时计算）')

if choice == '1':

size = float(input('请输入项目大小：（1代表标准大小，可以输入小数）'))

number = int(input('请输入人力数量：（请输入整数）'))

time = None

estimated(size,number,time)

elif choice=='2':

size = float(input('请输入项目大小：（1代表标准大小，可以输入小数）'))

time = int(input('请输入工时数量：（请输入整数）'))

number = None

estimated(size,number,time)

for i in range(1,3):

print('%d X %d=%d' %(i,2,i\*2))

for i in range(1,4):

print('%d X %d = %d' %(i,3,i\*3))

import numpy as np # 导入 numpy库，下面出现的 np 即 numpy库

scores1 = [91, 95, 97, 99, 92, 93, 96, 98]

scores2 = []

average = np.mean(scores1) # 一行解决。

print('平均成绩是：{}'.format(average))

for score in scores1:

if score < average:

scores2.append(score) # 少于平均分的成绩放到新建的空列表中

print(' 低于平均成绩的有：{}'.format(scores2)) # 上个关卡选做题的知识。

# 下面展示一种NumPy数组的操作，感兴趣的同学可以自行去学习哈。

scores3 = np.array(scores1)

print(' 低于平均成绩的有：{}'.format(scores3[scores3<average]))