第 **3** 天作业（函数）

# 基本概念

1. 简述位置参数，关键字参数，默认参数，动态参数的区别

* 位置参数是在调用函数时根据函数定义时所设定的参数位置来传递参数；
* 关键词参数是用于函数调用，通过“键-值”形式加以指定。同时参数无顺序需求；
* 默认参数是在定义函数时，为参数提供默认值，调用函数时可传可不传该默认参数的值
* 动态参数是函数在定义时不确定想传入多少个参数时使用的

1. 简述命名空间的 LEGB 规则。

python中查找命名空间时，先从函数内找，再从外部嵌套函数中找，再从函数定义的模块中找，最后在python内置模块中找

1. 什么是高阶函数，思考并举例说明高阶函数的应用场景。

高阶函数是指可以接受一个或多个函数作为参数，并/或返回一个函数作为结果的函数。如回调函数：高阶函数经常用于实现回调机制。在这种情况下，函数作为参数传递给其他函数，并在特定事件发生时被调用。

1. 什么是装饰器？用装饰器可以带来什么好处？思考并举例说明装饰器的应用场景。

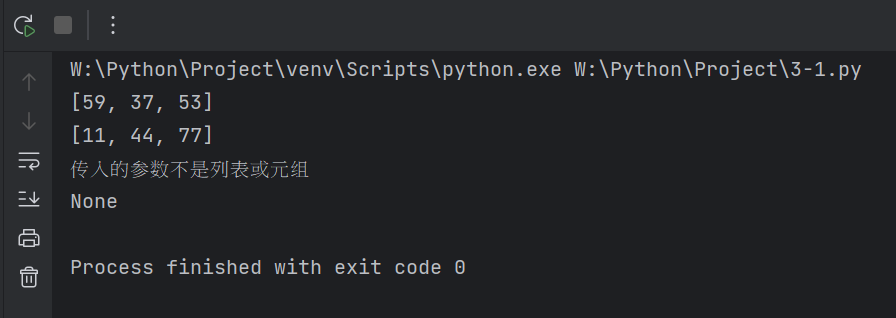
装饰器是用于拓展原来函数功能的一种函数，它可以在不改变原函数内容的情况下拓展原函数功能。当多人合作开发一个大型程序时，要想扩展别人设计的函数的功能，可以不用修改原函数，而是设计一个修饰器就可以解决这个问题。

1. 简述下迭代器和可迭代对象的区别，两者如何转换。

迭代器是一种可迭代对象，可迭代对象除了迭代器之外还有其他（如列表、字典等）集合类型数据。集合类型数据可以通过iter（）函数变成迭代器。

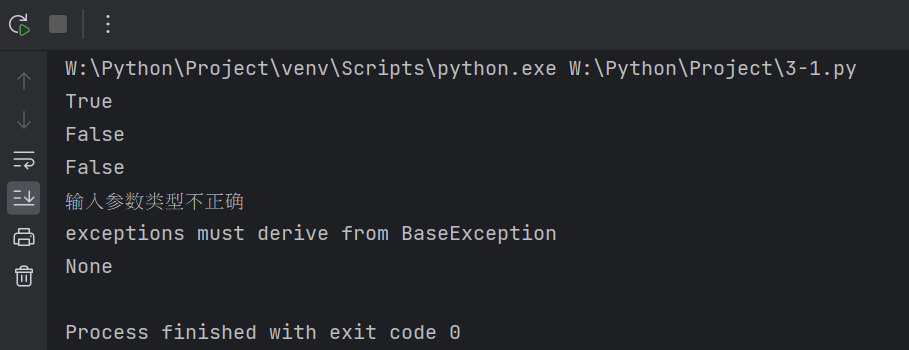
# 编程练习

1. 编写函数：输入一个列表或元组，输出的奇数位对应的元素，并返回一个新的列表，注意异常 处理（如果传入的参数不是列表或元组，则报异常）



def func(ele):  
 try:  
 if (type(ele) != type([0])) and (type(ele) != type((0,))):  
 raise Exception('传入的参数不是列表或元组')  
 else:  
 return [ele[i] for i in range(len(ele)) if i % 2 == 0]  
 except Exception as e:  
 print(e)  
  
  
print(func([59, 28, 37, 81, 53, 4]))  
print(func((11, 258, 44, 68, 77, 4)))  
print(func('tsinghua'))

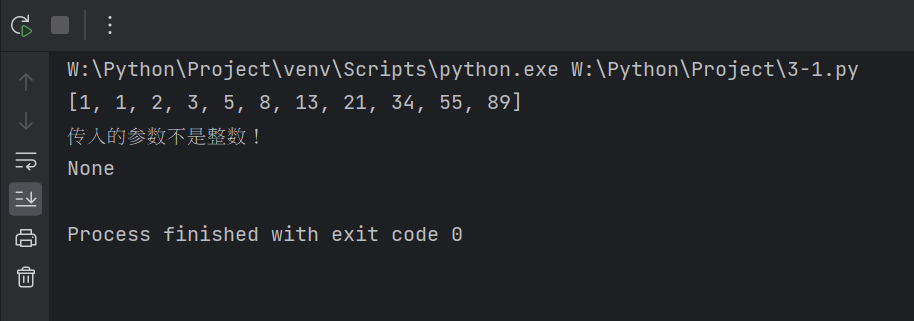
1. 编写函数：检查用户传入的对象（字符串、列表、元组）的每一个元素是否含有空格。若有， 返回 true，否则返回 false，注意异常处理（如果传入的参数不是字符串、列表或元组，则报异常）



def func(ele):  
 try:  
 if type(ele) != type('string') and type(ele) != type([0]) and type(ele) != type((0,)):  
 raise print('输入参数类型不正确')  
 else:  
 flag = False  
 if type(ele) == type('string') and ele.count(' '):  
 flag = True  
 else:  
 for i in ele:  
 if type(i) == type('string') and i.count(' '):  
 flag = True  
 return flag  
 except Exception as e:  
 print(e)  
  
print(func([1, 'I love you']))  
print(func('Tsinghua'))  
print(func((11, 12)))  
print(func(8))

1. 编写函数：输入一个整数 n，返回一个列表，该列表是斐波那契数列的前 n 项，注意异常处理

（如果传入的参数不是正整数，则报异常）



def func(n):  
 try:  
 if type(n) != type(1):  
 raise Exception('传入的参数不是整数！')  
 else:  
 if n == 1:  
 return [1]  
 if n == 2:  
 return [1, 1]  
 if n > 2:  
 list = [1, 1]  
 a, b, tem, num = 1, 1, 1, 1  
 while num < n:  
 temp = b  
 b = a + b  
 a = temp  
 list.append(b)  
 num += 1  
 return list  
 except Exception as e:  
 print(e)  
  
  
print(func(10))  
print(func('hi'))

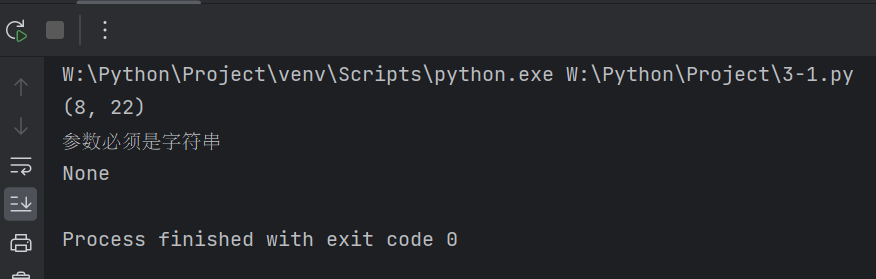
1. 编写函数，传入 n 个数，返回字典：{‘max’:最大值,’min’:最小值}，注意异常处理（如果传入的参数不是数字，则报异常）

一張含有 文字, 軟體, 多媒體軟體, 字型 的圖片

自動產生的描述

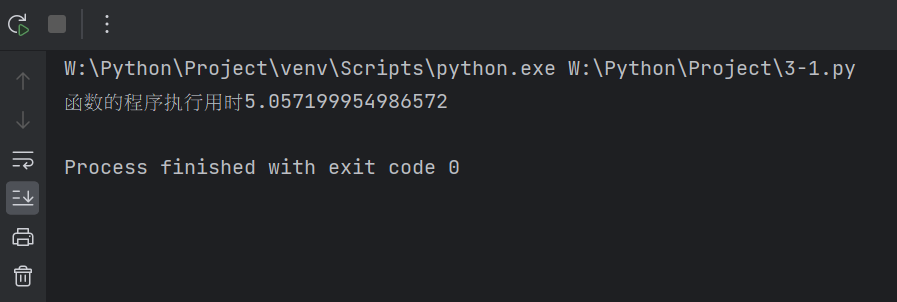
def func(\*args):  
 flag = False  
 for i in args:  
 if type(i) != type(1) and type(i) != type(1.0):  
 flag = True  
 try:  
 if flag:  
 raise Exception('输入的参数不是数字')  
 else:  
 return {'max': max(args), 'min': min(args)}  
 except Exception as e:  
 print(e)  
  
  
print(func(1, 22, 33, -5, 48, 5.5))  
print(func(1, 2, 3, 'hi'))

1. 编写函数, 接收字符串参数, 返回一个元组, 元组的第一个值为字符串中大写字母的个数, 第二个值为小写字母个数，注意异常处理（传入参数不是字符串，则报异常）



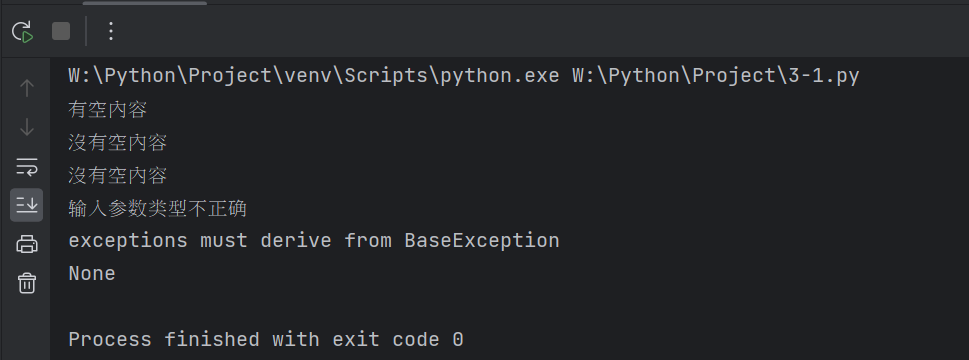
def func(str):  
 try:  
 if type(str) != type('hi'):  
 raise Exception('参数必须是字符串')  
 else:  
 low, up = 0, 0  
 for i in str:  
 if 'a' <= i <= 'z':  
 low += 1  
 elif 'A' <= i <= 'Z':  
 up += 1  
 return (up, low)  
 except Exception as e:  
 print(e)  
  
  
print(func('DjofDJjijfdMELowjqeoojVNpidjwe'))  
print(func(1))

1. 编写装饰器，计算某个函数的程序执行用时（结束时间-开始时间）；



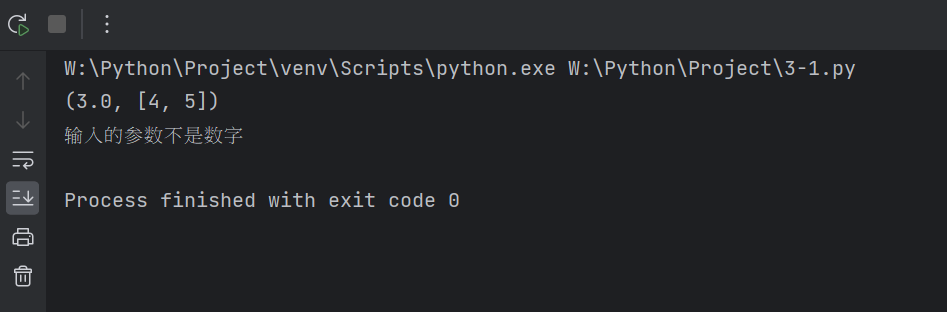
import time  
  
  
def during(func):  
 def inter():  
 t = time.time()  
 func()  
 print(f'函数的程序执行用时{time.time() - t}')  
  
 return inter  
  
  
@during  
def func():  
 return [i + 1 for i in range(100000000)]  
  
  
func()

1. 编写函数，检查用户传入的对象（字符串、列表、元组）的每一个元素是否含有空内容，注意 异常处理（如果传入的参数不是字符串、列表、元组，则报异常）



def func(ele):  
 try:  
 if type(ele) != type('string') and type(ele) != type([0]) and type(ele) != type((0,)):  
 raise print('输入参数类型不正确')  
 else:  
 flag = "沒有空內容"  
 if type(ele) == type('string') and ele.count(' '):  
 flag = "有空內容"  
 else:  
 for i in ele:  
 if type(i) == type('string') and i == (' '):  
 flag = "有空內容"  
 return flag  
 except Exception as e:  
 print(e)  
  
  
print(func([1, 'I love you', " "]))  
print(func('Tsinghua'))  
print(func((11, 12)))  
print(func(8))

1. 编写函数, 可以接收任意多个数, 返回的是一个元组. 元组的第一个值为所有参数的平均值, 第二个值是大于平均值的所有参数构成的列表，注意异常处理（如果传入的参数不是数字，则报 异常）



def func(\*args):  
 try:  
 total = sum(args)  
 count = len(args)  
 average = total / count  
  
 greater = [num for num in args if type(num) in (int, float) and num > average]  
  
 return (average, greater)  
 except TypeError:  
 raise Exception('输入的参数不是数字')  
  
try:  
 print(func(1, 2, 3, 4, 5))  
 print(func(10, 20, 30, 'hi'))  
except Exception as e:  
 print(e)

1. 编写装饰器，为多个函数加上认证的功能（用户的账号密码来源于文件），要求登录成功一 次，后续的函数都无需再输入用户名和密码；

一張含有 文字, 軟體, 多媒體軟體, 繪圖軟體 的圖片

自動產生的描述

def authenticate(func):  
 logged\_in = False  
  
 def wrapper(\*args, \*\*kwargs):  
 nonlocal logged\_in  
  
 if not logged\_in:  
 username = input("请输入用户名：")  
 password = input("请输入密码：")  
  
 def check(username, password):  
 if username == "hi" and password == "bye":  
 return True  
 else:  
 return False  
  
 if check(username, password):  
 logged\_in = True  
 else:  
 print("用户名或密码错误！")  
 return  
  
 return func(\*args, \*\*kwargs)  
  
 return wrapper  
  
  
@authenticate  
def func():  
 return sum([i + 1 for i in range(1000)])  
  
  
print(func())

1. 如下，每个小字典的 name 对应股票名字，shares 对应多少股，price 对应股票的价格

portfolio = [

{'name': 'IBM', 'shares': 100, 'price': 91.1},

{'name': 'AAPL', 'shares': 50, 'price': 543.22},

{'name': 'FB', 'shares': 200, 'price': 21.09},

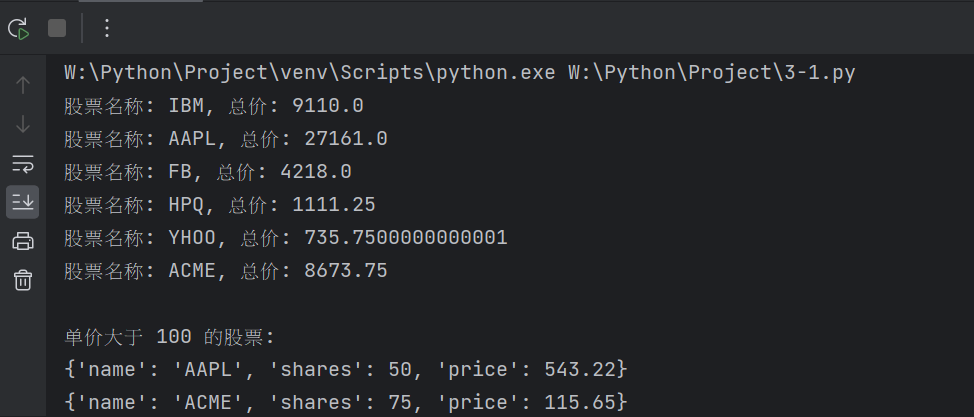
{'name': 'HPQ', 'shares': 35, 'price': 31.75},

{'name': 'YHOO', 'shares': 45, 'price': 16.35},

{'name': 'ACME', 'shares': 75, 'price': 115.65}

]

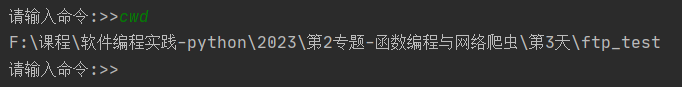
* 1. 使用内置函数计算购买每支股票的总价
  2. 用 filter 过滤出单价大于 100 的股票

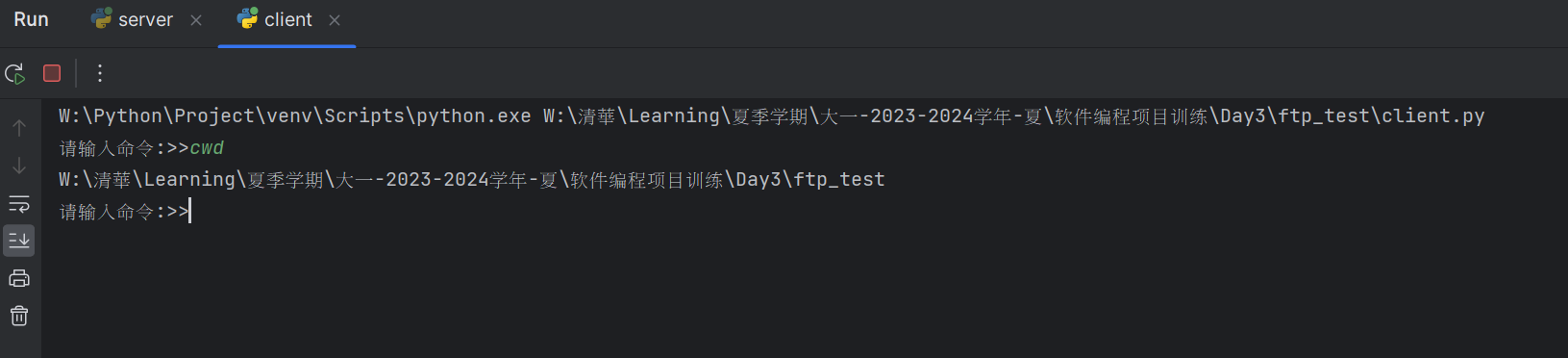


portfolio = [  
 {'name': 'IBM', 'shares': 100, 'price': 91.1},  
 {'name': 'AAPL', 'shares': 50, 'price': 543.22},  
 {'name': 'FB', 'shares': 200, 'price': 21.09},  
 {'name': 'HPQ', 'shares': 35, 'price': 31.75},  
 {'name': 'YHOO', 'shares': 45, 'price': 16.35},  
 {'name': 'ACME', 'shares': 75, 'price': 115.65}  
]  
  
for stock in portfolio:  
 value = stock['shares'] \* stock['price']  
 print(f"股票名称: {stock['name']}, 总价: {value}")  
  
filtered\_stocks = filter(lambda value: value['price'] > 100, portfolio)  
  
print("")  
print("单价大于 100 的股票:")  
for value in filtered\_stocks:  
 print(value)

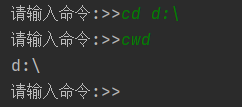
# 编程实践

1. 使用函数，封装“进击的小鸟”程序。
2. import pygame  
   import random  
     
     
   def birdgame():  
    # 设置画面大小  
    pygame.init()  
    width = 800  
    height = 600  
    screen = pygame.display.set\_mode((width, height))  
     
    # 设置背景  
    background\_color\_list = [(0, 0, 0), (0, 0, 255), (255, 0, 0), (0, 255, 0), (255, 255, 255)]  
    background\_color\_idx = 0  
     
    # 添加小乌  
    bird = pygame.image.load("W:/Python/Project/pic/bird.png")  
    bird\_rect = bird.get\_rect()  
    bird\_rect.right = 100  
    bird\_rect.top = 100  
     
    # 修改图片大小  
    bird = pygame.transform.scale(bird, (80, 60))  
     
    # 修改页面刷新频率  
    tick = 200  
    clock = pygame.time.Clock()  
     
    # 是否暂停  
    is\_pause = False  
    # 是否按下  
    b\_mouse\_press = False  
     
    # 初始化移动步长  
    step = [0, 0]  
     
    # 初始化小球列表  
    ball\_list = []  
    last\_time = pygame.time.get\_ticks()  
    # 得分  
    scores = 0  
    # 游戏结束  
    is\_game\_over = False  
     
    # 初始化字體  
    pygame.font.init()  
    score\_font\_size = 40  
    game\_over\_font\_size = 60  
    score\_font = pygame.font.SysFont("Song", score\_font\_size)  
    game\_over\_font = pygame.font.SysFont("Song", game\_over\_font\_size)  
     
    # 顯示初始分數  
    score\_text = score\_font.render("Scores: " + str(scores), True, (255, 255, 255))  
    screen.blit(score\_text, (10, 10))  
    pygame.display.flip()  
     
    # 背景音樂  
    pygame.mixer.init()  
    pygame.mixer.music.load("W:/Python/Project/sound/bgm.mp3")  
    pygame.mixer.music.play(-1, 0)  
     
    # 音效  
    write\_sound = pygame.mixer.Sound("W:/Python/Project/sound/bomb.wav")  
    red\_sound = pygame.mixer.Sound("W:/Python/Project/sound/succeed.wav")  
     
    while True:  
    clock.tick(tick)  
    b\_quit = False  
     
    screen.blit(score\_text, (10, 10))  
    # 小鳥運動  
    x\_step = 1  
    y\_step = 1  
     
    # 添加对鼠标和键盘操作的应用  
    for event in pygame.event.get():  
     
    # 点击关闭按钮  
    if event.type == pygame.QUIT:  
    b\_quit = True  
    break  
     
    # 小鸟跟随鼠标操作  
    if event.type == pygame.MOUSEBUTTONDOWN:  
    x = event.pos  
    if bird\_rect.collidepoint(x):  
    b\_mouse\_press = True  
    if event.type == pygame.MOUSEBUTTONUP:  
    b\_mouse\_press = False  
    if event.type == pygame.MOUSEMOTION:  
    if b\_mouse\_press:  
    x = event.pos[0] - bird\_rect.width / 2  
    y = event.pos[1] - bird\_rect.height / 2  
    bird\_rect.left = max(0, min(x, width - bird\_rect.width))  
    bird\_rect.top = max(0, min(y, height - bird\_rect.height))  
     
    # 按下回车键或空格键  
    if event.type in {pygame.KEYDOWN, pygame.KEYUP}:  
    press\_dict = {pygame.K\_UP: (0, -1), pygame.K\_DOWN: (0, 1), pygame.K\_LEFT: (-1, 0),  
    pygame.K\_RIGHT: (1, 0)}  
    if event.key in (pygame.K\_RETURN, pygame.K\_SPACE):  
    b\_quit = True  
    break  
    move\_flag = 1 if event.type == pygame.KEYDOWN else -1  
    if event.key in press\_dict:  
    press = press\_dict[event.key]  
    step[0] += press[0] \* move\_flag  
    step[1] += press[1] \* move\_flag  
     
    if b\_quit:  
    break  
     
    if is\_game\_over:  
    continue  
     
    # 生成小球  
    this\_time = pygame.time.get\_ticks()  
    if this\_time - last\_time > 500:  
    last\_time = this\_time  
    rand\_val = random.randint(0, 10)  
    is\_red = rand\_val <= 3  
     
    if is\_red:  
    ball\_body = pygame.image.load("W:/Python/Project/pic/red\_ball.png")  
    else:  
    ball\_body = pygame.image.load("W:/Python/Project/pic/ball.png")  
    ball\_rect = ball\_body.get\_rect()  
    ball\_rect.right = width  
    ball\_rect.top = random.randint(0, height - ball\_rect.height)  
    ball = {"body": ball\_body, "rect": ball\_rect, "red": is\_red}  
    ball\_list.append(ball)  
     
    for ball in ball\_list:  
    ball\_rect = ball['rect']  
    ball\_rect.left -= 1  
    if ball\_rect.right <= 0:  
    ball\_list.remove(ball)  
    scores += 1 if not ball['red'] else 0  
     
    if scores < 0:  
    is\_game\_over = True  
     
    # 是否碰撞  
    b\_collider = False  
    for ball in ball\_list:  
    ball\_rect = ball['rect']  
    if bird\_rect.colliderect(ball\_rect):  
    if ball['red']:  
    ball\_list.remove(ball)  
    scores += 3  
    red\_sound.play()  
    else:  
    b\_collider = True  
    write\_sound.play()  
    is\_game\_over = True  
    break  
     
    bird\_rect = bird\_rect.move(step)  
     
    # 填充背景色  
    screen.fill(background\_color\_list[background\_color\_idx])  
     
    screen.blit(score\_text, (10, 10))  
    # 绘制小球  
    for ball in ball\_list:  
    ball\_rect = ball['rect']  
    ball\_body = ball['body']  
    screen.blit(ball\_body, ball\_rect)  
     
    screen.blit(bird, bird\_rect)  
    pygame.display.flip()  
     
    # 更新分数文本  
    score\_text = score\_font.render("Scores: " + str(scores), True, (255, 255, 255))  
     
    # 碰撞後退出遊戲  
    if is\_game\_over:  
    pygame.time.wait(500)  
    game\_over\_text = game\_over\_font.render("Final Scores: " + str(scores), True, (255, 0, 0))  
    game\_over\_rect = game\_over\_text.get\_rect(center=(width // 2, height // 2))  
    screen.blit(game\_over\_text, game\_over\_rect)  
    pygame.display.flip()  
    while True:  
    for event in pygame.event.get():  
    if event.type == pygame.QUIT:  
    pygame.quit()  
    exit()  
    elif event.type == pygame.MOUSEBUTTONDOWN or event.type == pygame.KEYDOWN:  
    pygame.quit()  
    exit()  
     
     
   birdgame()
3. 编程实现简易 FTP 程序，要求： 客户端：
   * 输入 cwd，显示当前所在目录（示意）





* + 输入 cd {dir\_name}，修改当前工作目录（示意）



一張含有 文字, 軟體, 多媒體軟體, 繪圖軟體 的圖片

自動產生的描述

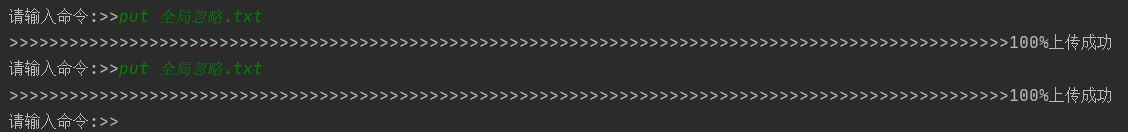
* + 输入 ls，浏览当前目录下的所有文件与文件夹信息（示意）

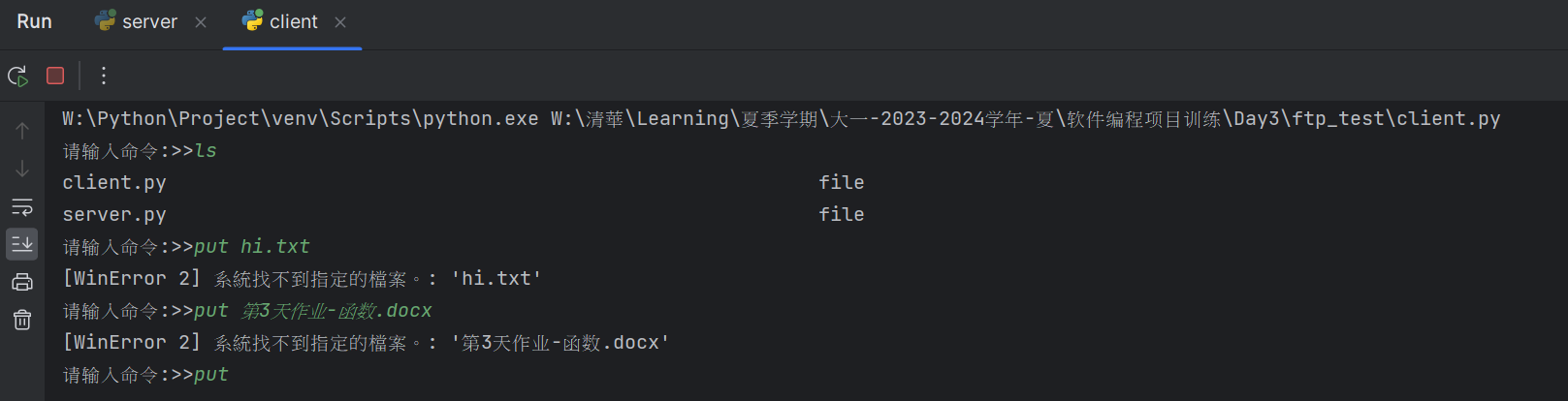


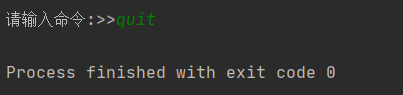
一張含有 文字, 多媒體軟體, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

* + 输入 put {file\_name}，上传文件（检查是否为文件，文件是否存在）





* + 输入 quit， 则退出 FTP 连接（示意）

一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

自動產生的描述