

**学生实验实习报告册**

|  |  |
| --- | --- |
| 学年学期： | 2023-2024 学年 🞏春■秋学期 |
| 课程名称： | Python程序设计与应用 |
| 学生学院： | 自动化学院 |
| 专业班级： | 08052102 |
| 学生学号： | 2021212981 |
| 学生姓名： | 王忠全 |
| 联系电话： | 1508669338 |

**重庆邮电大学教务处制**

|  |  |  |  |
| --- | --- | --- | --- |
| **实验测验名称** | **Python面向对象程序设计、文件和数据格式化实验** | | |
| **地点** | C318 | **时间** | 第八周9-12节 |
| **9-12指导教师** | 张 焱 | **成绩** |  |
| **一、实验目的**  1、了解面向对象的程序设计思想；  2、了解对象、类、封装、继承、方法、构造函数和析构函数等面向对象的程序设计的基本概念；  3、学习声明类；  4、学习静态变量、静态方法和类方法；  5、学习类的继承和多态；  6、学习复制对象的方法  7、掌握文件的读写方法以及打开和关闭等基本操作；  8、文件复制、移动、重命名、遍历等文件级操作以及目录操作 | | | |
| **二、实验测验所需仪器设备和软件**  Pychram图标 的图像结果  Pycharm 2022.2.3  环境：Python 3.8 | | | |

|  |
| --- |
| **三、实验题目**   1. 自定义类模拟三维向量及其运算。 2. 编写一个学生和教师数据输入和显示程序。其中，学生数据有编号、姓名、班级和成绩，教师数据有编号、姓名、职称和部门。要求：  * 将编号、姓名输入和显示设计成一个类person； * 设计类person的派生类：学生类student和教师类teacher；  1. 定义一个员工类Employee，有数据成员姓名，编号。定义一个销售员继承自员工类Sales，工资为销售额的提成10%，定义一个经理类Manager，固定工资8000，定义一个销售经理类，继承自销售员类和经理类，工资为固定工资5000加销售额的提成5%。每个类均有display函数输出信息，编写主函数测试。 2. 编写一个程序计算出球、圆柱和圆锥的表面积和体积，要求：  * 定义一个基类，至少含有一个数据成员半径，并设为私有成员； * 定义基类的派生类球、圆柱、圆锥，都含有求表面积和体积的成员函数和打印函数； * 编写主函数，求球、圆柱、圆锥的表面积和体积。   5、批量生成姓名、家庭住址、电子邮箱等随机信息，并以二进制进行文件读写。  6、读写文本文件并添加行号。  7、查找磁盘内大小相同的文件。  **接下来分别对每个题目进行分析：**   1. **【自定义类模拟三维向量及其运算】**     定义一个三位向量类My\_3D，这个类有三个属性即是x,y,z，需要在类里面定义函数\_\_add\_\_和\_\_sub\_\_和\_\_mul\_\_以及show函数，这样分别实现加减乘法，实例化类后，就可以模拟三位向量及其运算。结合所学的数学知识利用Python编写对应函数。  **2.【编写一个学生和教师数据输入和显示程序】**  学生数据有编号、姓名、班级和成绩  教师数据有编号、姓名、职称和部门  要求将编号、姓名输入和显示设计成一个类person；  设计类person的派生类：学生类student和教师类teacher；    这主要是定义一个父类person，然后编写子类的时候，继承父类，其中很重要的函数：super().\_\_init\_\_()函数。在Python中， super() 是一个特殊的函数，用于调用父类的方法或属性。它经常与子类一起使用，以便在子类中扩展或重写父类的方法。  **3.【为一个公司编写职员数据类】**  定义一个员工类Employee，有数据成员姓名，编号。  定义一个销售员继承自员工类Sales，工资为销售额的提成10%。  定义一个经理类Manager，固定工资8000。  定义一个销售经理类，继承自销售员类和经理类，工资为固定工资5000加销售额的提成5%。  每个类均有display函数输出信息，编写主函数测试。  扁平化创意卡通商务青年男士设计素材_漫品狗_MG动画短片素材_flash源文件_动漫矢量图免费素材网  这也主要是需要定义一个父类Employee，父类下有三个子类Sales、Manager、SalesManager，这三个子类都需要继承父类Employee，并且在每个子类都编写自己的money薪资函数以及展示show函数。  这里也利用super()函数继承了父类的Display展示函数。  **4.【编写一个程序计算出球、圆柱和圆锥的表面积和体积】**  定义一个基类，至少含有一个数据成员半径，并设为私有成员；  定义基类的派生类球、圆柱、圆锥，都含有求表面积和体积成员函数和打印函数；  编写主函数，求球、圆柱、圆锥的表面积和体积。      这里需要一个基类，我定义的就是半径数据成员，设置为私有成员，使用公有方法实现访问私有数据成员，即定义def public(self)返回值是self.\_\_r私有的半径。  之后对于三个子类（球、圆柱、圆锥）继承父类，在子类中分定义函数求其对应的表面积和体积。  **5、【批量生成姓名、家庭住址、电子邮箱等随机信息】**  并以二进制进行文件读写  **生成：**  对于批量生成随机信息数据，选取Python中的Faker包，它是开源的GITHUB项目，主要用来创建伪数据，使用Faker包，无需再手动生成或者手写随机数来生成数据，只需要调用Faker提供的方法，即可完成数据的生成。  所以利用Faker包获取随机的姓名、家庭地址、电子邮箱  name = fake.name()  address = fake.address()  email = fake.email()  【邮箱卡通素材】免费下载_邮箱卡通图片大全_千库网png  **写：**  在这里，需要将随机信息转化成二进制文件，于是使用utf-8对随机信息进行编码，编码后还需要记录长度打包为二进制数据的一部分。主要是为了在读取数据时能够方便地找到每个字段的边界。因为在文件中，不同的字段可能会有不同的长度，如果没有一个明确的方式来标记每个字段的起始和结束位置，就很难准确地读取出每个字段的值。  通过在每个字段的前面添加一个表示其长度的 4 字节的无符号整数，就可以保证每个字段的长度都是固定的。在读取数据时，我们先读取 4 字节的长度信息，然后再根据长度信息读取对应长度的二进制数据，就能够准确地分割出每个字段的值。  这里使用struct库，重要的pack()函数对二进制数据进行打包。  **读：**  这里就是将之前打包的数据进行解码，读取指定长度的数据，按照utf-8解码为字符串，并且打印，就完成了读的操作。  **6、【读写文本文件并添加行号】**    这里就是需要读取一个文件，用readlines()函数读出每一行内容，并把读到的每一行的索引值和内容取出，这里考虑用enumerate()函数，然后遍历，对每一行进行写入，即可实现读取文本并添加行号。  **7、【查找磁盘内大小相同的文件】**    这里主要是需要遍历文件目录，将文件的大小size和路径path存储在一个字典里面，按照键值对，如果文件大小相同，则size共用一个键，值path存在多个。  这样只需要寻找size键下路径path数大于2的，就找到了相同文件啦。 |

|  |
| --- |
| **四、实验结果及分析讨论（含原始数据记录等）**   1. **【自定义类模拟三维向量及其运算】**      1. # 自定义类模拟三维向量及其运算，加法、减法、向量与标量的乘法和除法  4. **class** My\_3D(object): 5. **def** \_\_init\_\_(self, x, y, z): 6. self.a = x 7. self.b = y 8. self.c = z 10. **def** \_\_add\_\_(self, other): 11. my\_add = My\_3D(self.a + other.a, self.b + other.b, self.c + other.c) 12. **return** my\_add 14. **def** \_\_sub\_\_(self, other): 15. my\_sub = My\_3D(self.a - other.a, self.b - other.b, self.c - other.c) 16. **return** my\_sub 18. **def** \_\_mul\_\_(self, other): 19. my\_mul = My\_3D(self.a \* other.a, self.b \* other.b, self.c \* other.c) 20. **return** my\_mul 22. **def** show(self): 23. **print**((self.a, self.b, self.c))  26. v1 = My\_3D(1, 2, 3) 27. v2 = My\_3D(4, 5, 6) 28. v\_sum = v1 + v2 29. v\_sub = v1 - v2 30. v\_mul = v1 \* v2 32. # 输出 33. **print**('原来的v1：') 34. v1.show() 35. **print**('原来的v2：') 36. v2.show() 37. **print**('相加得：') 38. v\_sum.show() 39. **print**('相减得：') 40. v\_sub.show() 41. **print**('相乘得：') 42. v\_mul.show()     测试实例化输入第一个三维向量（1，2，3）第二个三维向量（4，5，6），相加得（5，7，9）相减得（-3，-3，-3），相乘得（4，10，18），经测试实验结果符合预期，实验正确。  **2.【编写一个学生和教师数据输入和显示程序】**     1. # 编写一个学生和教师数据输入和显示程序。其中，学生数据有编号、姓名和成绩，教师数据有编号、姓名、职称和部门。要求： 2. # 将编号、姓名输入和显示设计成一个类person 3. # 设计类person的派生类：学生类student和教师类teacher  6. **class** person(object): 7. **def** \_\_init\_\_(self, num, name): 8. self.num = num 9. self.name = name 11. **def** show(self): 12. **print**(self.num, self.name)  15. **class** student(person): 16. **def** \_\_init\_\_(self, num, name, cls, grades): 17. super().\_\_init\_\_(num, name)    # 继承父类 18. self.cls = cls 19. self.grades = grades 21. **def** show(self): 22. **print**(self.num, self.name, self.cls, self.grades)  25. **class** teacher(person): 26. **def** \_\_init\_\_(self, num, name, oppostion, location): 27. super().\_\_init\_\_(num, name)    # 继承父类 28. self.oppostion = oppostion 29. self.location = location 31. **def** show(self): 32. **print**(self.num, self.name, self.oppostion, self.location) 34. # person1 = person(2021212981, '王忠全') 35. # person1.show()  38. student2 = student(2021212981, '王忠全', 8052102, 98,) 39. student2.show() 41. teacher3 = teacher(1013198974, '张老师', 'Python实验老师', '自动化学院') 42. teacher3.show()     测试实例化：  输入student，show展示学号，姓名，班级和成绩为：  2021212981 王忠全 8052102 98  输入teacher，show展示编号，姓名，职位和部门为：  1013198974 张老师 Python实验老师 自动化学院  经测试实验结果符合预期，实验正确。  **3.【为一个公司编写职员数据类】**     1. # 定义一个员工类Employee，有数据成员姓名，编号。 2. # 定义一个销售员继承自员工类Sales，工资为销售额的提成10% 3. # 定义一个经理类Manager，固定工资8000 4. # 定义一个销售经理类，继承自销售员类以及经理类，工资为固定工资5000加销售额提成5% 5. # 每个类均有Display函数输出信息，编写主程序测试函数 7. # 员工类Employee 8. **class** Employee(object): 9. **def** \_\_init\_\_(self, name, num): 10. self.name = name 11. self.num = num 13. **def** Display(self): 14. **print**('姓名：', self.name) 15. **print**('编号：', self.num)  18. # 销售员类Sales 19. **class** Sales(Employee): 20. **def** \_\_init\_\_(self, name, num, money): 21. super().\_\_init\_\_(name, num) 22. self.money = money 24. **def** money\_10per(self): 25. money\_10 = self.money \* 0.1 26. **return** money\_10 28. **def** Display(self): 29. super(Sales, self).Display() 30. **print**("销售员工资：", self.money\_10per())  33. # 经理类Manager 34. **class** Manager(Employee): 35. **def** \_\_init\_\_(self, name, num): 36. super().\_\_init\_\_(name, num) 38. **def** money\_static(self): 39. **return** 8000  42. **def** Display(self): 43. super(Manager, self).Display() 44. **print**("经理工资：", self.money\_static())  47. # 销售经理类SalesManager 48. **class** SalesManager(Sales, Manager): 49. **def** \_\_init\_\_(self, name, number, money): 50. Sales.\_\_init\_\_(self, name, number, money) 51. Manager.\_\_init\_\_(self, name, number) 53. **def** money\_static\_add(self): 54. total\_money = self.money \* 0.05 + 5000 55. **return** total\_money 57. **def** Display(self): 58. super(SalesManager, self).Display() 59. **print**("销售经理工资：", self.money\_static\_add())  62. **print**("----------") 63. xiaowang = Employee('小王', 2021212981) 64. xiaowang.Display() 66. **print**("----------") 67. xiaohu = Sales('小胡', 2021212989, 5400) 68. xiaohu.Display() 70. **print**("----------") 71. xiaonuo = Manager('小虎', 2020202964) 72. xiaonuo.Display() 74. **print**("----------") 75. xiaoguo = SalesManager('小狮', 2019264695, 7000) 76. xiaoguo.Display()     测试实例化，分别输入员工、销售员、经理、销售经理，得出：  ----------  姓名： 小王  编号： 2021212981  ----------  姓名： 小胡  编号： 2021212989  销售员工资： 540.0  ----------  姓名： 小虎  编号： 2020202964  经理工资： 8000  ----------  姓名： 小狮  编号： 2019264695  经理工资： 8000  销售员工资： 700.0  销售经理工资： 5350.0  经测试实验结果符合预期，实验正确。  **4.【编写一个程序计算出球、圆柱和圆锥的表面积和体积】**     1. # 编写一个程序计算出球、圆柱和圆锥的表面积和体积，要求： 2. # 定义一个基类，至少含有一个数据成员半径，并设为私有成员； 3. # 定义基类的派生类球、圆柱、圆锥，都含有求表面积和体积的成员函数和打印函数； 4. # 编写主函数，求球、圆柱、圆锥的表面积和体积。 5. **import** math 6. pi = math.pi  9. **class** My\_R(object): 10. **def** \_\_init\_\_(self, r): 11. self.\_\_r = r 13. **def** public\_r(self):    # 公有方法访问私有r 14. **return** self.\_\_r  17. # 球 18. **class** My\_Qiu(My\_R): 19. **def** \_\_init\_\_(self, r): 20. super().\_\_init\_\_(r) 21. self.r = self.public\_r() 23. **def** My\_Qiu\_V(self): 24. **return** 4 / 3 \* pi \* pow(self.r, 3) 26. **def** My\_Qiu\_S(self): 27. **return** 4 \* pi \* pow(self.r, 2) 29. **def** show(self): 30. **print**("圆的半径：{0},体积：{1:.2f},表面积：{2:.2f}".format(self.r, self.My\_Qiu\_V(), self.My\_Qiu\_S()))  33. # 圆柱 34. **class** My\_Yuanzhu(My\_R): 35. **def** \_\_init\_\_(self, r, h): 36. super().\_\_init\_\_(r) 37. self.r = self.public\_r() 38. self.h = h 40. **def** My\_Yuanzhu\_V(self): 41. **return** pi \* pow(self.r, 2) \* self.h 43. **def** My\_Yuanzhu\_S(self): 44. **return** 2 \* pi \* pow(self.r, 2) + 2 \* pi \* self.r \* self.h 46. **def** show(self): 47. **print**("圆柱的半径：{0},体积：{1:.2f},表面积：{2:.2f}".format(self.r, self.My\_Yuanzhu\_V(), self.My\_Yuanzhu\_S()))  50. **class** My\_Yuanzhui(My\_R): 51. **def** \_\_init\_\_(self, r, h, l): 52. super().\_\_init\_\_(r) 53. self.r = self.public\_r() 54. self.h = h 55. self.l = l 57. **def** My\_Yuanzhui\_V(self): 58. **return** 1 / 3 \* pi \* pow(self.r, 2) \* self.h 60. **def** My\_Yuanzhui\_S(self): 61. **return** pi \* self.r \* self.l + pi \* pow(self.r, 2) 63. **def** show(self): 64. **print**("圆锥的半径：{0},体积：{1:.2f},表面积：{2:.2f}".format(self.r, self.My\_Yuanzhui\_V(), self.My\_Yuanzhui\_S())) 66. # 打印输入 67. Qiu1 = My\_Qiu(2) 68. Qiu1.show() 70. Yuanzhu1 = My\_Yuanzhu(2, 5) 71. Yuanzhu1.show() 73. Yuanzhi1 = My\_Yuanzhui(2, 5, 8) 74. Yuanzhi1.show()     测试实例化：  输入球半径2；圆柱半径2，高5；圆锥半径2，高5，母线8  输出：  圆的半径：2,体积：33.51,表面积：50.27  圆柱的半径：2,体积：62.83,表面积：87.96  圆锥的半径：2,体积：20.94,表面积：62.83  经测试实验结果符合预期，实验正确。  **5、【批量生成姓名、家庭住址、电子邮箱等随机信息】**   1. # 批量生成姓名、家庭住址、电子邮箱等随机信息，并以二进制进行文件读 / 写。 2. # fake 库即可随机生成伪数据 4. **import** random 5. **import** faker 6. **import** struct 8. fake = faker.Faker()    # 实例化fake  11. # 随机生成：姓名、家庭住址、电子邮箱 12. **def** My\_message(): 13. name = fake.name() 14. address = fake.address().replace('\n', ', ')    # 把地址拼接到一段 15. email = fake.email() 16. **return** name, address, email  # 返回随机的姓名，地址，邮箱  19. # 批量生成随机信息写入二进制文件 20. **def** My\_write(file\_path, num\_entries): 21. with open(file\_path, 'wb') as fp: 22. **for** i **in** range(num\_entries):    # 写入多少数量 23. name, address, email = My\_message() 25. name\_2data = name.encode('utf-8')    # 将字符串转换为二进制数据 26. address\_2data = address.encode('utf-8') 27. email\_2data = email.encode('utf-8') 28. data = struct.pack('I', len(name\_2data)) + name\_2data     # 使用struct将数据打包为二进制格式 29. data += struct.pack('I', len(address\_2data)) + address\_2data 30. data += struct.pack('I', len(email\_2data)) + email\_2data 31. fp.write(data)    # 写   35. # 从二进制文件中读取信息 36. **def** My\_read(file\_path): 37. with open(file\_path, 'rb') as fp: 38. **while** True: 39. length\_data = fp.read(4)    # 从文件中读取4字节的数据，解析为字符串长度 40. **if** **not** length\_data: 41. **break** 42. length = struct.unpack('I', length\_data)[0] 43. data = fp.read(length)      # 读取指定长度的数据，并解码为字符串 44. **print**(data.decode('utf-8')) # 打印  47. # 调用函数生成随机信息并写入二进制文件 48. file\_path = 'My\_file.bin'     # 写的文件名字 49. num\_entries = 3                   # 随机生成信息的数量 50. My\_write(file\_path, num\_entries)  # 写入文件 52. # 从二进制文件中读取信息并打印 53. My\_read(file\_path)     运行程序，可以看到终端输出了一系列随机的信息，并且将信息保存在文件“My\_file”中，这样就实现了批量生成姓名、家庭住址、电子邮箱等随机信息，并以二进制进行文件读写。  经测试实验结果符合预期，实验正确。  **6、【读写文本文件并添加行号】**   1. # 读写文本文件并添加行号。 3. **def** My\_Addline(file, file\_change): 4. with open(file, 'r', encoding='utf-8', errors='ignore') as fp:    # 以读模式打开文件 5. lines = fp.readlines()    # 读取每一行 6. **print**(lines) 8. with open(file\_change, 'w', encoding='utf-8') as fp\_change:    # 以写模式打开文件 9. # for i, line in enumerate(lines, 1): 10. #     fp\_change.write(f"{i}: {line}") 11. **for** index, item **in** enumerate(lines):    # 获取行数与行内容 12. **print**(index+1)    # 行数需要+1，不然是从第0行开始的 13. **print**(item) 14. fp\_change.write("第{0}行:{1}".format(index+1, item))  17. file = "My\_file\_2.txt"  # 输入文件名 18. file\_change = "My\_file\_change.txt"  # 输出文件名 19. My\_Addline(file, file\_change)     可以看到终端输出了数字和一行一行的信息，我们在文件中看看。  没有添加行号前My\_file\_2：    添加行号后My\_file\_change：    可以看到程序运行后，给txt文本每一行添加了行号，第一行、第二行……经测试实验结果符合预期，实验正确。  **7、【查找磁盘内大小相同的文件】**   1. # 查找磁盘内大小相同的文件 2. # 遍历目录及其子目录中的所有文件，将它们的文件大小作为键，文件路径作为值存储在一个字典中。 3. # 如果多个文件具有相同的文件大小，则将路径添加到相应的列表中 5. **import** os  8. **def** My\_find\_file(fp): 9. file\_size = {}    # 定义一个文件大小字典 11. **for** dirpath, dirnames, filenames **in** os.walk(fp): 12. # os.walk() 函数 13. # 第一个参数 dirpath：需要遍历的文件夹路径 14. # 第二个参数 dirnames：当前文件夹下所有子文件夹的名称列表 15. # 第三个参数 filenames：当前文件夹下所有文件的名称列表 16. **for** filename **in** filenames: 17. path = os.path.join(dirpath, filename)    # 拼接函数 18. **print**(path) 19. size = os.path.getsize(path) 20. **print**("文件大小：", size) 22. **if** size **not** **in** file\_size:      # 按照文件大小存储文件路径 23. file\_size[size] = [path]   # 键是文件大小，值是文件路径 24. **else**: 25. file\_size[size].append(path)    # 如果有相同文件大小，则直接存路径 27. # 开始查找相同大小文件 28. **for** size, paths **in** file\_size.items(): 29. **if** len(paths) >= 2:     # 如果一个文件大小键下有两个路径值 -- 则重复 30. **print**("\n发现具有相同大小的文件啦:") 31. **for** path **in** paths: 32. **print**(path) 34. **print**("\n程序结束~")  37. My\_path = "C:/Users/Akaxi/Desktop/Akaxi\_python/Python\_experiment\_Akaxi/第三次实验"  # 替换为你所需的目录 38. My\_find\_file(My\_path)       我在指定目录：C:/Users/Akaxi/Desktop/Akaxi\_python/Python\_experiment\_Akaxi/第三次实验  创建了两相同大小的文件My\_file\_chongfu\_1.txt和My\_file\_chongfu\_2.txt，运行程序，输出结果找到了大小相同的文件，并且把目录也打印出来，根据我的目录结构，经测试实验结果符合预期，实验正确。 |

|  |
| --- |
| **五、实验测验总结与体会**  注：至少应该包括以下内容：设计过程存在的各种困难、描述如何解决这些问题的，得到什么样结论 |
| **六、评阅意见**  评阅人签字：  评阅日期： |