实 验 报 告

**2023年 月 日 星期 第 节 勤学楼 D653教室**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **课程名称** | 计算机程序设计-Python | | | | | | **课序号** | |  | | **任课教师** | | 蔡京哲 | | |
| **学生姓名** |  | | **学号** |  | | **专业** | |  | | | | **年级** | | |  |
| **进入实验室时间** | | **进入时仪器设备状况** | | | **离开实验室时间** | | | | | **离开时仪器设备状况** | | | | **机器号** | |
|  | | 正常 | | |  | | | | | 正常 | | | |  | |
| **实验项目名称** | | 实验6.1 字典 | | | | | | | | | | | | | |
| **一、实验目的**  1. 掌握字典的创建方法；  2. 掌握字典的基本操作方法；  3. 掌握字典的遍历方法；  4. 掌握字典的嵌套使用方法。  **二、实验内容**  参考实验指导书中实验6.1的实验内容。  **三、实验过程（步骤、命令或源程序代码）及结果（截图）**  1.阅读并运行程序（粘贴代码和运行结果）  （1）  程序：  person1 = {  '身份证号' : '1110112XXXXXXX0101',  '姓名' : '王宏',  '性别' : '男',  '居住地' : '北京市XX区XX街道XX小区'  } #直接创建  person2 = dict(person1 ) #通过其他字典创建  person3 = dict([  ('身份证号','110112XXXXXXXX0101'),  ('姓名','王宏'),  ('性别','男'),  ('居住地','北京市XX区XX街道 XX小区')  ]) #通过“（键，值）”对的序列创建  person4 = dict(  身份证号 = '110112XXXXXXXX0101',  姓名 = '王宏', 性别 = '男',  居住地 = '北京市XX区XX街道XX小区'  ) #通过关键字参数创建  person5 = dict(zip(  ['身份证号','姓名','性别','居住地'],  ['110112XXXXXXXX0101','王宏','男','北京市XX区XX街道XX小区']  )) #通过 dict 和 zip 结合创建  print(person1['姓名'])  print(person2.get('姓名'))  print(person1)  print(person2)  print(person3)  print(person4)  print(person5)  运行结果：  王宏  王宏  {'身份证号': '1110112XXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市XX区XX街道XX小区'}  {'身份证号': '1110112XXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市XX区XX街道XX小区'}  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市XX区XX街道 XX小区'}  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市XX区XX街道XX小区'}  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市XX区XX街道XX小区'}  （2）  程序：  person = {  '身份证号' : '110112XXXXXXXX0101',  '姓名' : '王宏',  '性别' : '男',  '居住地' : '北京市XX区XX街道XX小区',  }  person['居住地'] = '北京市通州区'  person['电话'] = 13801010101  print(person)  运行结果：  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市通州区', '电话': 13801010101}  （3）  程序：  person = {  '身份证号' : '110112XXXXXXXX0101',  '姓名' : '王宏',  '性别' : '男',  '居住地' : '北京市XX区XX街道XX小区',  '电话' : '13801010101'  }  del person['电话']  print(person)  person.popitem()  print(person)  运行结果：  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市XX区XX街道XX小区'}  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男'}  （4）  程序：  person = {  '身份证号' : '110112XXXXXXXX0101',  '姓名' : '王宏',  '性别' : '男',  '居住地' : '北京市XX区XX街道XX小区'  }  person.update({'居住地' : '北京市通州区XX街道XX小区','电话' : '13801010101'})  print(person)  person2 = person.copy()  print (person2)  运行结果：  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市通州区XX街道XX小区', '电话': '13801010101'}  {'身份证号': '110112XXXXXXXX0101', '姓名': '王宏', '性别': '男', '居住地': '北京市通州区XX街道XX小区', '电话': '13801010101'}  （5）  程序：  stu\_class = {  '赵' : '优',  '钱' : '良',  '孙' : '中',  '李' : '及格'  }  for name, report in stu\_class.items():  print(name,'的成绩是',report)  运行结果：  赵 的成绩是 优  钱 的成绩是 良  孙 的成绩是 中  李 的成绩是 及格  （6）  程序：  family\_age= {  "grandmather" : 65,  "grandfather" : 68,  "mather" : 38,  "father" : 38,  "child" : 10  }  old = family\_age["grandmather"]  print("grandmother is "+ str(old)+".")  运行结果：  grandmother is 65.  2.程序改错（粘贴修改后的完整代码，修改的语句用红色标记）  （1）  number = int(input("请输入将要存入字典的英文单词个数:"))  word\_explain = {}  for i in range(number):  word,explain = input("请输入英文:"),input("请输入对应的译文:")  word\_explain[word] = explain  new\_word = input("请输入要翻译的英文单词:")  if new\_word in word\_explain:  print(word\_explain[new\_word])  else:  print("no found")  3.程序填空（粘贴空白区域的代码）  （1）  ①list(milk.items())  ②M  （2）  ①key in a  ②delkey.append(key)  ③del  4.编程题（粘贴代码和运行结果）  （1）  程序：  inventory = {  1: ["A", 100],  2: ["B", 150],  3: ["C", 200]  }  for product\_id, (product\_name, stock) in inventory.items():  print(f"商品编号：{product\_id}, 商品名称：{product\_name}, 库存量：{stock}")  运行结果：  商品编号：1, 商品名称：A, 库存量：100  商品编号：2, 商品名称：B, 库存量：150  商品编号：3, 商品名称：C, 库存量：200  (2)  程序：  card\_numbers = [f"62010090{x:03}" for x in range(1, 101)]  password = "01234567"  cards = {card: password for card in card\_numbers}  print("卡号\t\t\t密码")  for card, pwd in cards.items():  hidden\_card = card[:4] + "\*\*\*\*\*\*\*\*" + card[-4:]  print(f"{hidden\_card}\t{pwd}")  运行结果：  卡号 密码  6201\*\*\*\*\*\*\*\*0001 01234567  6201\*\*\*\*\*\*\*\*0002 01234567  6201\*\*\*\*\*\*\*\*0003 01234567  6201\*\*\*\*\*\*\*\*0004 01234567  6201\*\*\*\*\*\*\*\*0005 01234567  6201\*\*\*\*\*\*\*\*0006 01234567  6201\*\*\*\*\*\*\*\*0007 01234567  6201\*\*\*\*\*\*\*\*0008 01234567  6201\*\*\*\*\*\*\*\*0009 01234567  6201\*\*\*\*\*\*\*\*0010 01234567  6201\*\*\*\*\*\*\*\*0011 01234567  6201\*\*\*\*\*\*\*\*0012 01234567  6201\*\*\*\*\*\*\*\*0013 01234567  6201\*\*\*\*\*\*\*\*0014 01234567  6201\*\*\*\*\*\*\*\*0015 01234567  6201\*\*\*\*\*\*\*\*0016 01234567  6201\*\*\*\*\*\*\*\*0017 01234567  6201\*\*\*\*\*\*\*\*0018 01234567  6201\*\*\*\*\*\*\*\*0019 01234567  6201\*\*\*\*\*\*\*\*0020 01234567  6201\*\*\*\*\*\*\*\*0021 01234567  6201\*\*\*\*\*\*\*\*0022 01234567  6201\*\*\*\*\*\*\*\*0023 01234567  6201\*\*\*\*\*\*\*\*0024 01234567  6201\*\*\*\*\*\*\*\*0025 01234567  6201\*\*\*\*\*\*\*\*0026 01234567  6201\*\*\*\*\*\*\*\*0027 01234567  6201\*\*\*\*\*\*\*\*0028 01234567  6201\*\*\*\*\*\*\*\*0029 01234567  6201\*\*\*\*\*\*\*\*0030 01234567  6201\*\*\*\*\*\*\*\*0031 01234567  6201\*\*\*\*\*\*\*\*0032 01234567  6201\*\*\*\*\*\*\*\*0033 01234567  6201\*\*\*\*\*\*\*\*0034 01234567  6201\*\*\*\*\*\*\*\*0035 01234567  6201\*\*\*\*\*\*\*\*0036 01234567  6201\*\*\*\*\*\*\*\*0037 01234567  6201\*\*\*\*\*\*\*\*0038 01234567  6201\*\*\*\*\*\*\*\*0039 01234567  6201\*\*\*\*\*\*\*\*0040 01234567  6201\*\*\*\*\*\*\*\*0041 01234567  6201\*\*\*\*\*\*\*\*0042 01234567  6201\*\*\*\*\*\*\*\*0043 01234567  6201\*\*\*\*\*\*\*\*0044 01234567  6201\*\*\*\*\*\*\*\*0045 01234567  6201\*\*\*\*\*\*\*\*0046 01234567  6201\*\*\*\*\*\*\*\*0047 01234567  6201\*\*\*\*\*\*\*\*0048 01234567  6201\*\*\*\*\*\*\*\*0049 01234567  6201\*\*\*\*\*\*\*\*0050 01234567  6201\*\*\*\*\*\*\*\*0051 01234567  6201\*\*\*\*\*\*\*\*0052 01234567  6201\*\*\*\*\*\*\*\*0053 01234567  6201\*\*\*\*\*\*\*\*0054 01234567  6201\*\*\*\*\*\*\*\*0055 01234567  6201\*\*\*\*\*\*\*\*0056 01234567  6201\*\*\*\*\*\*\*\*0057 01234567  6201\*\*\*\*\*\*\*\*0058 01234567  6201\*\*\*\*\*\*\*\*0059 01234567  6201\*\*\*\*\*\*\*\*0060 01234567  6201\*\*\*\*\*\*\*\*0061 01234567  6201\*\*\*\*\*\*\*\*0062 01234567  6201\*\*\*\*\*\*\*\*0063 01234567  6201\*\*\*\*\*\*\*\*0064 01234567  6201\*\*\*\*\*\*\*\*0065 01234567  6201\*\*\*\*\*\*\*\*0066 01234567  6201\*\*\*\*\*\*\*\*0067 01234567  6201\*\*\*\*\*\*\*\*0068 01234567  6201\*\*\*\*\*\*\*\*0069 01234567  6201\*\*\*\*\*\*\*\*0070 01234567  6201\*\*\*\*\*\*\*\*0071 01234567  6201\*\*\*\*\*\*\*\*0072 01234567  6201\*\*\*\*\*\*\*\*0073 01234567  6201\*\*\*\*\*\*\*\*0074 01234567  6201\*\*\*\*\*\*\*\*0075 01234567  6201\*\*\*\*\*\*\*\*0076 01234567  6201\*\*\*\*\*\*\*\*0077 01234567  6201\*\*\*\*\*\*\*\*0078 01234567  6201\*\*\*\*\*\*\*\*0079 01234567  6201\*\*\*\*\*\*\*\*0080 01234567  6201\*\*\*\*\*\*\*\*0081 01234567  6201\*\*\*\*\*\*\*\*0082 01234567  6201\*\*\*\*\*\*\*\*0083 01234567  6201\*\*\*\*\*\*\*\*0084 01234567  6201\*\*\*\*\*\*\*\*0085 01234567  6201\*\*\*\*\*\*\*\*0086 01234567  6201\*\*\*\*\*\*\*\*0087 01234567  6201\*\*\*\*\*\*\*\*0088 01234567  6201\*\*\*\*\*\*\*\*0089 01234567  6201\*\*\*\*\*\*\*\*0090 01234567  6201\*\*\*\*\*\*\*\*0091 01234567  6201\*\*\*\*\*\*\*\*0092 01234567  6201\*\*\*\*\*\*\*\*0093 01234567  6201\*\*\*\*\*\*\*\*0094 01234567  6201\*\*\*\*\*\*\*\*0095 01234567  6201\*\*\*\*\*\*\*\*0096 01234567  6201\*\*\*\*\*\*\*\*0097 01234567  6201\*\*\*\*\*\*\*\*0098 01234567  6201\*\*\*\*\*\*\*\*0099 01234567  6201\*\*\*\*\*\*\*\*0100 01234567  (3)  程序：  # 创建摩尔斯电码的编码规则字典  morse\_code = {  'A': '.-', 'B': '-...', 'C': '-.-.',  'D': '-..', 'E': '.', 'F': '..-.',  'G': '--.', 'H': '....', 'I': '..',  'J': '.---', 'K': '-.-', 'L': '.-..',  'M': '--', 'N': '-.', 'O': '---',  'P': '.--.', 'Q': '--.-', 'R': '.-.',  'S': '...', 'T': '-', 'U': '..-',  'V': '...-', 'W': '.--', 'X': '-..-',  'Y': '-.--', 'Z': '--..'  }  def text\_to\_morse(text):  morse = []  for char in text.upper():  if char == ' ':  morse.append('/')  elif char in morse\_code:  morse.append(morse\_code[char])  else:  morse.append(char)  return ' '.join(morse)  text = input("请输入要翻译的文本: ")  morse\_text = text\_to\_morse(text)  print(f"摩尔斯电文: {morse\_text}")  运行结果：  请输入要翻译的文本: ABCD  摩尔斯电文: .- -... -.-. -..  **四、存在问题及解决方法**  **1、存在问题：**试图删除不存在的键时，出现KeyError。  **解决方法：**在删除之前检查键是否存在，或使用pop()方法，它会返回并删除键，如果键不存在，返回默认值。  **2、存在问题：**在遍历字典时试图修改字典的大小（添加或删除键值对）可能会导致意外的行为。  **解决方法：**创建字典的副本并在副本上进行操作，或者收集需要删除的键并在循环结束后删除它们。  **五、实验总结**  **这次实验我们学习了字典的创建和基本操作，包括添加、修改、删除数据以及遍历字典。通过实践，我掌握了不同创建字典的方法，如直接创建和使用zip函数等。在操作上，我学会了安全获取值的get()方法，以及使用del和popitem()删除键值对。编程实践中，我完成了库存管理和摩尔斯电码转换的小项目，深化了对字典在实际应用中的理解。这次实验让我对字典有了更清晰的认识，我相信这会是我以后编程中的有力工具。** | | | | | | | | | | | | | | | |
| **教师评语：** | | | | | | | | | | | | | | | |