**中央财经大学**

**实 验 报 告**

**实验项目名称**  字典列表实验

**所属课程名称**  Python程序设计

**实 验 类 型**  综合、设计性

**实 验 日 期**  2019.10.23

**班 级**  计算机科学18

**学 号**  2018312352

**姓 名**  杨智江

**成 绩**

**实验室**

|  |
| --- |
| **实验概述：** |
| **【实验目的及要求】**  编写程序完成题目要求  **【实验原理】**  Python3.7  **【实验环境】（使用的软件）**  Python3.7 IDLE  Visual Studio Code |
| **实验内容：** |
| **【实验方案设计】**  **一. 字典和列表的方法。**  (a) 创建一个字典，并把这个字典中的键按照字母顺序显示出来。  (b) 现在根据已按照字母顺序排序好的键，显示出这个字典中的键和值。  (c)同(b),但这次是根据已按照字母顺序排序好的字典的值，显示出这个字典中的键和值。(注意：对字典和哈希表来说，这样做一般没有什么实际意义，因为大多数访问和排序(如果需要)都是基于字典的键，这里只把它作为一个练习。)  **二、 颠倒字典中的键和值。用一个字典做输入，输出另一个字典，用前者的键做值，前者的值做键**。  **三、下面的问题和例题7.1（userpw.py.） 中管理名字-密码的键值对数据的程序有关。** (a)修改那个脚本，使它能记录用户上次的登录日期和时间(用time 模块)，并与用户密码一起保存起来。程序的界面有要求用户输入用户名和密码的提示。无论户名是否成功登录，都应有提示，在户名成功登录后，应更新相应用户的上次登录时间戳。如果本次登录与上次登录在时间上相差不超过4 个小时，则通知该用户： “You already logged in at: <last\_ login\_timestamp>.” (b) 添加一个“管理”菜单，其中有以下两项:(1)删除一个用户 (2)显示系统中所有用户的名字和他们的密码的清单。 (c) 要求用户名不区分大小写。 (d) 加强对用户名的限制，不允许符号和空白符。 (e)合并“新用户”和“老用户”两个选项。如果一个新用户试图用一个不存在的用户名登录， 询问该用户是否是新用户，如果回答是肯定的，就创建该帐户。否则，按照老用户的方式登录。  **【实验过程】（实验步骤、记录、数据、分析）**  **Q1:**  1、源代码：  dicta=dict(b=2,a=1,c=3)  '''  for key in sorted(dicta):      print(key)  '''  keys = dicta.keys()  print(sorted(keys)) #也可以不通过循环，在一行中显示  for key in sorted(dicta):      print('key=%s, value=%s' %(key,dicta[key]))  listvalue = dicta.values()  for value in sorted(listvalue): #dict\_values不支持sort()用法      for key in dicta.keys():          if value == dicta[key]:              print('key=%s, value=%s' %(key,dicta[key]))   1. 参考代码中使用for循环输出，也可以在一行中显示 2. dict\_values不支持sort()用法，所以用sorted()排序   2、运行结果    **Q2:**  源代码：  dicta={2:'b',1:'a',3:'c'}  print("Previous: "+str(dicta))  def changedict(dict):      newdict = {}      for key in dict:          newdict.update({}.fromkeys(dict[key],key))      print("Current: "+str(newdict))  changedict(dicta)   1. 在这里调试时出现了错误，原因是如果要将字典和字符串同时输出要将字典转换为str类型。   运行结果：  **Q3:**  (a)  为了使时间与密码一起保存，将字典的值更改成列表  passwd = input('Please enter a password: ')  localTime = time.time() #获取当前格式化的时间  info=[passwd,localTime] #创建列表  db[inputName] = info #更新字典  print("Success")  在第二次登录后，更新时间  print('Welcome back! '+name)  previousTime=db.get(name)[1]  currentTime=time.time()  db.get(name)[1]=currentTime #读取时间，并重新存入时间  if (currentTime-previousTime)<=14400:  #判断时间是否在4小时（14400秒）内       print("You already logged in at: ",time.asctime(time.localtime(previousTime))) #利用查到的asctime()函数生成可读的时间  (b)  更改了showmenu()的代码，在showmenu()内新增选项，设置了管理密码  if choice == 'm':           if input("Please input the admin password:")=='123':                  manageMenu()           else:                  print("Wrong password. Please try again.")  重用了showmenu()的代码，生成管理菜单，有删除用户和显示用户的选项  def manageMenu():      prompt = """      (D)elete a User      (S)how User      (Q)uit      Enter choice: """      done = False #重用menu的部分代码      while not done:          chosen = False          while not chosen:              try:                  choice = input(prompt).strip()[0].lower()              except (EOFError, KeyboardInterrupt):                  choice = 'q'              print('\nYou picked: [%s]' % choice)              if choice not in 'dsq':                  print('invalid option, try again')              else:                  chosen = True          if choice == 'q':              done = True          if choice == 'd':              deleteUser()          if choice == 's':              showUsers()  函数的实现  def deleteUser():      inputName=input("Please input the name: ")      flag=0      for i in range(len(db)):              if inputName.lower() in str(list(db.keys())[i]).lower(): #利用循环，将用户名全部小写，查找是否存在                  name=str(list(db.keys())[i]) #读取真正用户名                  flag=1 #增加flag判断用户名是否存在                  break      if flag==0:          print("Error.Name not found.")      else:          db.pop(name)          print("Delete %s successful." % name)  def showUsers():      print("Number of users: %d" % len(db))  #返回用户数量      for key in db:          print('name = %s, password = %s' %(key,db[key][0])) #返回用户信息  (c)  分别在用户登录时和删除用户时能不区分大小写  用户登录  flag=0      for i in range(len(db)):              if inputName.lower() in str(list(db.keys())[i]).lower(): #利用循环，将用户名全部小写，查找是否存在                  name=str(list(db.keys())[i]) #读取真正用户名                  flag=1 #增加flag判断用户名是否存在                  break        if flag == 0:          print("The username does not exist.")  删除用户  flag=0      for i in range(len(db)):              if inputName.lower() in str(list(db.keys())[i]).lower(): #利用循环，将用户名全部小写，查找是否存在                  name=str(list(db.keys())[i]) #读取真正用户名                  flag=1 #增加flag判断用户名是否存在                  break      if flag==0:          print("Error.Name not found.")  (d) 利用第六章介绍过的正则表达式，判断用户名是否合法  import time  import re  db = {}  def userLogin():      prompt = """  ---SIGN UP/IN---  Username: """      while True:          inputName = input(prompt)          if not re.match("^[a-zA-Z0-9]\*$", inputName): #利用正则表达式，判断用户名是否合法              print('Special characters and blanks are not allowed.')              prompt = 'Try another:'              continue          else:              break  (e)判断顺序：用户名是否合法→用户名是否存在→（不存在时）创建账户或重新输入  def userLogin():      prompt = """  ---SIGN UP/IN---  Username: """      while True:          inputName = input(prompt)          if not re.match("^[a-zA-Z0-9]\*$", inputName): #利用第六章的正则表达式，判断用户名是否合法              print('Special characters and blanks are not allowed.')              prompt = 'Try another:'              continue          else:              break      flag=0      for i in range(len(db)):              if inputName.lower() in str(list(db.keys())[i]).lower(): #利用循环，将用户名全部小写，查找是否存在                  name=str(list(db.keys())[i]) #读取真正用户名                  flag=1 #增加flag判断用户名是否存在                  break        if flag == 0:          print("The username does not exist.")          chosen = False          while not chosen:              choice=input("Do you want to register as a new user? Y/N: ").lower() #询问用户选择              if choice not in 'yn':                      print('invalid option, try again')              else:                  chosen = True          if choice == 'y':              passwd = input('Please enter a password: ')              localTime = time.time() #获取当前格式化的时间              info=[passwd,localTime] #创建列表              db[inputName] = info #更新字典              print("Success")      if flag == 1:          while True:              passwd = input('Password: ')              if db.get(name)[0]==passwd:                  print('Welcome back! '+name)                  previousTime=db.get(name)[1]                  currentTime=time.time()                  db.get(name)[1]=currentTime #读取时间，并重新存入时间                  if (currentTime-previousTime)<=14400: #判断时间是否在4小时（14400秒）内                      print("You already logged in at: ",time.asctime(time.localtime(previousTime)))                      #利用查到的asctime()函数生成可读的时间                  break              else:                  print('Wrong password.Please retry.')  **【结论】（结果）**  **程序运行结果**  **(a)**    **(b)**  **(c)**    **(d)(e)** |
| **【小结】**  **通过实验，我练习了字典的用法，复习了相关的内建函数。了解了time模块和re模块的一些操作。在调试程序中出现了很多bug，有的会报错，有的是编译时发现不了的，比如一开始循环后没有考虑到用户名不存在的情形，这需要在调试中测试所有情况并改正。** |
| **指导教师评语及成绩：** |
| **评语：**  **成绩： 指导教师签名：**  **批阅日期：** |

实验报告说明

**1．实验项目名称：**要用最简练的语言反映实验的内容。要求与实验指导书中相一致。

**2．实验类型：**一般需说明是验证型实验还是设计型实验，是创新型实验还是综合型实验。

**3．实验目的与要求**：目的要明确，要抓住重点，符合实验指导书中的要求。

**4．实验原理：**简要说明本实验项目所涉及的理论知识。

**5．实验环境**：实验用的软硬件环境（配置）。

**6．实验方案设计（思路、步骤和方法等）**：这是实验报告极其重要的内容。概括整个实验过程。

对于**操作型实验**，要写明依据何种原理、操作方法进行实验，要写明需要经过哪几个步骤来实现其操作。

对于**设计型和综合型实验**，在上述内容基础上还应该画出流程图、设计思路和设计方法，再配以相应的文字说明。

对于**创新型实验**，还应注明其创新点、特色。

**7．实验过程（实验中涉及的记录、数据、分析）：**写明具体上述实验方案的具体实施，包括实验过程中的记录、数据和相应的分析。

**8．结论（结果）：**即根据实验过程中所见到的现象和测得的数据，做出结论。

**9．小结：**对本次实验的心得体会、思考和建议。

**10．指导教师评语及成绩：**指导教师依据学生的实际报告内容，用简练语言给出本次实验报告的评价和价值。

**注意：**

* 实验报告将记入实验成绩；
* 每次实验开始时，交上一次的实验报告，否则将扣除此次实验成绩。