随机生成密码

[ 1 ]

from random import choice

from string import ascii\_letters,digits,punctuation

all\_chs = ascii\_letters + digits + punctuation

def randpass(n=8):

result = ''

for i in range(n):

ch = choice(all\_chs)

result += ch

return result

if \_\_name\_\_ == '\_\_main\_\_':

print(randpass())

print(randpass(10))

[ 2 ]

from random import choice

from string import ascii\_letters,digits,punctuation

all\_chs = ascii\_letters + digits + punctuation

def randpass(n=8):

result =[choice(all\_chs) for i in range(n)]

return ''.join(result) #将列表的字符拼接起来

if \_\_name\_\_ == '\_\_main\_\_':

print(randpass())

print(randpass(10))

1.shutil模块

复制和移动(shutil.copyfileobj)

shutil.copyfileobj将类似文件的对象fsrc的内容复制到类似文件的对象fdst。

import shutil

# f1 = open('/etc/passwd','rb')

# f2 = open('/tmp/xxx','wb')

#

# shutil.copyfileobj(f1,f2)

#

# f1.close()

# f2.close()

#

shutil.copyfile将名为src的文件的内容(无元数据)复制到名为dst的文件,然后返回dst。

# shutil.copyfile('/etc/passwd','/tmp/xyz') #拷贝文件

# shutil.copy('/etc/passwd','/tmp')

# shutil.copy('/etc/passwd','/tmp/aabb')

# shutil.copy2('/etc/passwd','/tmp/aabbcc') #copy2 权限拷贝，相当于cp -p

shutil.move递归地将文件或目录(src)移动到另一个位置(dst),并返回目标。

# import shutil

#

# shutil.move(src='/etc/passwd',dst='/mnt/passwd')

shutil.copytree递归地复制以src为根的整个目录树,返回目标目录。由dst命名的目标目录不能已经存在。

# import shutil

#

# shutil.copytree('/etc/security','/tmp/anquan')

# shutil.rmtree(/tmp/anquan) #删除

#

# 变量赋值

# a ,b=[10,20]

# a,b = b, a

#

# import keyword

#

# keyword.kwlist # 查看定义的关键字

# keyword.iskeyword('pass') # 查看是否为关键字

#

#

# import os

# os.path.exists('/tmp') #判断文件是否存在

课堂练习：

创建文件

1. 编写一个程序,要求用户输入文件名

2. 如果文件已存在,要求用户重新输入

3. 提示用户输入数据,每行数据先写到列表中

4. 将列表数据写入到用户输入的文件名中

import os #os模块判断是否存在

def get\_fname():

while True:

fname = input('请输入一个文件名:')

if not os.path.exists(fname):

break

print('文件以存在,请从新输入')

# print('%s 文件以存在，请从新输入' % fname)

return fname

def get\_content():

content = []

print('请输入正文，end结束')

while True:

line = input('>')

if line == 'end':

break

content.append(line)

return content

def wfile(fname, content):

with open(fname, 'w') as fobj:

fobj.writelines(content)

if \_\_name\_\_ == '\_\_main\_\_':

fname = get\_fname()

content = get\_content()

content = ['%s\n' % line for line in content]

wfile(fname, content)

1. 字符串

len(seq):返回seq的长度

max(iter,key=None):返回iter中的最大值

enumerate:接受一个可迭代对象作为参数,返回一个enumerate对象

# users = ['bob','alice','john']

#

# for i in range(len(users)):

# print('#%s :%s' % (i,users[i]))

#

# list(enumerate(users))

# for item in enumerate(users): #应用时不必替换

# print('#%s : %s' % (item[0],item[1]))

#

# for ind,user in enumerate(users):

# print('#%s : %s' % (ind,user))

# for u in reversed(users): #反序

# print(u)

#

# sorted(users) #排序

练习:检查标识符

1. 程序接受用户输入

2. 判断用户输入的标识符是否合法

3. 用户输入的标识符不能使用关键字

4. 有不合法字符,需要指明第几个字符不合法

# from string import ascii\_letters,digits

# from keyword import iskeyword

#

# first\_chs = ascii\_letters + '\_'

# other\_chs = first\_chs + digits

#

# def check\_id(idt):

# if iskeyword(idt):

# return '%s is keyword' % idt

#

# if idt[0] not in first\_chs:

#查看首字母是否符合first\_chs(字母和数字)，不是就返回1st invalid

# return '1st invalid'

#

# for ind, ch in enumerate(idt[1:]): #因为在上面判断了首字母，所以切片，把第一个字母去掉

# if ch not in other\_chs: #当ch（idt）的值不在other\_chs范围内，就写出第几个字符不合法

# return '第%s个字符不合法' % (ind + 2) #因为上面判断了第一个字符，所以在判断第几个字符不合法时，是把第二个当成了0（第一个），所以ind加2，从第一个开始

#

# return '%s是合法的' % idt

#

# if \_\_name\_\_ == '\_\_main\_\_':

# idt = input('代检查的标示符:')

# print(check\_id(idt))

#

格式化操作符

#########常用########

# '%s is %s years old' % ('bob',25)

# '%s is %d years old' % ('bob',25)

# '%12s%8s' % ('name','age') #右对齐

# '%-12s%8s' % ('name','age')

# '%-12s%-8s' % ('bob',25) #左对齐

# #############不常用####################

# '%c' % 98

# '#%o' % 30 #8进制数

# import subprocess #调用shell

# subprocess.call('ls /root' shell=True)

###################################

#创建用户

# 1. 编写一个程序,实现创建用户的功能

# 2. 提示用户输入用户名

# 3. 随机生成8位密码

# 4. 创建用户并设置密码

# 5. 将用户相关信息写入指定文件

# import sys

# import subprocess

# from randpass import randpass

#

# def adduser(username,password,fname):

#

# subprocess.call('useradd %s' % username,shell=True)

# subprocess.call('echo %s | passwd --stdin %s' % (password,username),shell=True)

#

# user\_info = '''user information:

# 用户名: %s

# 密码： %s

# '''% (username,password)

#

# with open(fname,'a') as fobj:

# fobj.write(user\_info)

#

# if \_\_name\_\_ == '\_\_main\_\_':

# username = sys.argv[1]

# password = randpass()

# adduser(username,password,'/tmp/users.txt')

#格式化输出

# 1. 提示用户输入(多行)数据

# 2. 假定屏幕的宽度为50,用户输入的多行数据如下显示(文本内容居中):

astr = '\thello '

astr.strip() #去除两端空白字符

astr.lstrip() #去除左端空白字符

astr.rstrip() #去除右端空白字符

'hello.tar.gz'.split('.')

hi = 'hello world'

hi.title() #首写字母大写

hi.upper() #大写

hi.lower()

hi.islower() #有一个大写或着小写，不统一，返回false

'hao123'.isdigit() #不统一，返回false

hi.isidentifier()

hi.center(50)

hi.center(50,'#')

hi.ljust(50)

hi.rjust(50)

hi.startswith('he')

hi.endswith('d') #查看hi中是否有d

# ##################更新列表

# alist = [1,50,60,100]

# alist[0] =10

# alist[1:3] = [30,40]

# alist[3:3] = [50,60,70,80,90]

# alist.append(30) #追加

# alist.sort() #排序

# alist.reverse() #反序

# alist.remove(30)

# alist.pop() #默认弹出最后一项

# alist.pop(2)

# alist.extend('abc') #字符串

# alist.extend(['abc','xyc']) #列表

# alist.index(50) #50的下表

# alist.insert(3,1000) #把1000放到下标为3的位置

#

#######################用列表构建栈结构

# 1. 栈是一个后进先出的结构

# 2. 编写一个程序,用列表实现栈结构

# 3. 需要支持压栈、出栈、查询功能

# stack = [] #全局变量，从定义开始到程度结束都可见可用

# def push\_it():

# item = input('上传项目 item to push:')

# stack.append(item)

# #print('push\_it')

#

# def pop\_it():

# if stack:

# print('pop:\033[31;1m %s\033[0m' % stack.pop())

# #print('pop\_it')

# def view\_it():

# print('\033[032;1m%s\033[0m' % stack)

#

# def show\_menu():

# cmds = {'0':push\_it,'1':pop\_it,'2':view\_it}

# prompt = '''(0) push it

# (1) pop it

# (2) view it

# (3) exit

# 请输入序号(0/1/2/3): '''

#

# while True:

# choice = input(prompt).strip()[0]

# if choice not in '0123':

# print('无效输入，请重新输入')

# continue

# if choice == '3':

# break

# cmds[choice]()

# # if choice == '0':

# # push\_it()

# # elif choice == '1':

# # pop\_it()

# # else:

# # view\_it()

#

#

# if \_\_name\_\_ == '\_\_main\_\_':

# show\_menu()