1. 字典基础操作
2. 创建字典

• 通过{ }操作符创建字典

• 通过dict()工厂方法创建字典

• 通过fromkeys()创建具有相同值的默认字典

# adict = dict(['ab','cd',('name','zhangsan')])

# print(adict)

# cdict = {}.fromkeys(('bob','alice'),23) #创建具有相同值的默认字典

# print(cdict)

# bdict = {}.fromkeys(['bob','alice','tom'],7)

# print(bdict)

# for key in bdict:

# print('%s :%s' % (key,bdict[key]))

#

# print('%(bob)s' % bdict)

1. 更新字典

• 通过键更新字典

– 如果字典中有该键,则更新相关值

– 如果字典中没有该键,则向字典中添加新值

# bdict['tom'] =8 #更新字典

# bdict['john']=6

# print(bdict)

bdict.pop('alice') #删除字典中的值

7 in bdict # 返回False

'tom' in bdict # 返回True

3.删除字典

• 通过del可以删除字典中的元素或整个字典

• 使用内部方法clear()可以清空字典

• 使用pop()方法可以“弹出”字典中的元素

4.字典操作符

• 使用字典键查找操作符[ ],查找键所对应的值

• 使用in和not in判断键是否存在于字典中

1. 字典相关函数

len():返回字典中元素的数目

hash():本身不是为字典设计的,但是可以判断某个对象是否可以作为字典的键

dict.copy():返回字典(浅复制)的一个副本

dict.get(key, default=None):对字典dict中的键key,返回它对应的值value,如果字中不存在此键,则返回default的值

dict.setdefault(key, default=None):如果字典中不存在key键,由dict[key]=default为它赋值

dict.items():返回一个包含字典中(键,值)对元组的列表

dict.keys():返回一个包含字典中键的列表

dict.values():返回一个包含字典中所有值的列表

dict.update(dict2):将字典dict2的键-值对添加到字典dict

# cdict = bdict.copy() #将bdict的内容赋值给cdict，cdict使用全新的内存

# bdict.get('bob') # 返回bob对应的value，如果没有bob，默认返回None

# bdict.get('jane','not found') # 如果没有jane，返回not found

# bdict.setdefault('bob',10) #bob已经是字典的key，向字典中写入

# bdict.setdefault('jane',10) #jane没在字典中，向字典中写入

# list(bdict.keys())

# list(bdict.values())

# list(bdict.items())

# bdict.update({'aaa':111,'bob':222}) #合并字典

#dict.update(dict2):将字典dict2的键-值对添加到字典dict

课堂练习：

模拟用户登陆信息系统

1. 支持新用户注册,新用户名和密码注册到字典中

2. 支持老用户登陆,用户名和密码正确提示登陆成功

3. 主程序通过循环询问进行何种操作,根据用户的选择,执行注册或是登陆操作

# from getpass import getpass

# userdb = {}

#

# def login():

# username = (input('请输入用户:'))

# password = (input('请输入密码:'))

# #if username not in userdb or userdb[username] != password:

# if userdb.get(username) != password:

# print('登陆失败')

# else:

# print('登陆成功')

# def register():

# username = input('username')

# if username not in userdb:

# password = input('password:')

# userdb[username] = password

# else:

# print('%s以存在' % username)

#

#

# def show\_men():

# cmds = {'0':register,'1':login}

# prompt ='''

# (0) 新用户注册

# (1) 登陆

# (2) 退出

# 请选择(0/1/2): '''

# while True:

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

# if choice not in '012':

# print('无效输入，请重试')

# continue

#

# if choice == '2':

# break

#

# cmds[choice]()

#

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

# show\_men()

编写unix2dos的程序

1. Windows文本文件的行结束标志是\r\n

2. 类unix文本文件的行结束标志是\n

3. 编写程序,将unix文本文件格式转换为windows文本文件的格式

# import sys #sys位置参数

#

# def unix2dox(name):

# dst\_fname = name + '.txt'

# src\_fobj = open(name)

# dst\_fobj = open(dst\_fname,'w')

# for line in src\_fobj:

# line = line.rstrip() + '\r\n' #line.rstrip去掉右边空行 \r回车不换行

# dst\_fobj.write(line)

# src\_fobj.close()

# dst\_fobj.close()

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

# unix2dox(sys.argv[1])

编写类进度条程序

1. 在屏幕上打印20个#号

2. 符号@从20个#号穿过

3. 当@符号到达尾部,再从头开始

# import time

# import sys

# l = 19

# counter = 0

# print('#' \* (l + 1),end='')

#

# while True:

# sys.stdout.flush()

# time.sleep(0.5)

# print('\r%s@%s' % ('#' \* counter, '#' \* (l - counter)),end='')

# counter +=1

# if counter >l:

# counter = 0

1. 集合

• 数学上,把set称做由不同的元素组成的集合,

集合(set)的成员通常被称做集合元素

• 集合对象是一组无序排列的可哈希的值

• 集合有两种类型

– 可变集合set

– 不可变集合frozenset

#集合也用{}表示，就像一个无值的列表

# s1 = set('abc')

# len(s1)

#

# for ch in s1:

# print(ch)

#

# s2 = set('bcd')

#

# s1 | s2 #联合，取并集，把不同的列出来

# s1 & s2 #取交集，把相同的取出来

# s1 - s2 #s1中有的但s2中没有的

#

#

# s1.add(10) #添加值

# s1.update(['hello','world']) #批量添加值

# s1.remove(10) #移出值

# s1.issubset(s2) #s1是s2的子集吗

# s1.issuperset(s2) #s1是s2的超集

# s1.union(s2) # s1 | s2

# s1.intersection(s2) # s1 & s2

# s1.difference(s2) #s1 - s2

列出两个文件中不同

# cp /etc/passwd /tmp

# cp /etc/passwd /tmp/mima

#修改第二个文件与第一个文件有些行不一样

# f1 = open('/tmp/passwd')

# f2 = open('/tmp/mima')

# f3 = open('/tmp/result','w')

#

# s1 = set(f1)

# s2 = set(f2)

#

# f3.writelines(s2-s1)

# f1.close()

# f2.close()

# f3.close()

#

1. 时间方法

1.time模块

# import time

#

# time.localtime() #返回九元祖

#

# time.gmtime() #0时区的九元祖

# time.time() #返回时间戳，常用

# time.mktime() #把九元祖转换成时间戳

# time.sleep(1)

# time.asctime() #默认返回当前的UTC时间

# time.ctime() #默认返回当前的UTC时间，常用

# time.ctime(0) #时间戳作为参数

# time.asctime(t1)

# time.strftime('%Y- %m -%d %H：%M：%S') #常用

# time.strptime('2018- 08 -20','%Y- %m -%d') #转九元祖

2.datetime模块

# from datetime import datetime,timedelta

# t1 = datetime.now()

# t1.year

# t1.month

# t2 = datetime.today() #类似与datetime.now,只是参数不一样

# datetime.strptime('2018-8-20','%Y-%m-%d') #返回datetime对象

#

# dt = timedelta(days=100)

# t1 -dt

# t1 + dt

五．异常

1.什么是异常

当python检测到一个错误时,解释器就会指出当前流已经无法继续执行下去,这时候就出现了异常

• 异常是因为程序出现了错误而在正常控制流以外采取的行为

• 这个行为又分为两个阶段:

– 首先是引起异常发生的错误

– 然后是检测(和采取可能的措施)阶段

1. python中的异常

当程序运行时,因为遇到未解的错误而导致中止运行,便会出现traceback消息,打印异常



3.try-except语句

定义了进行异常监控的一段代码,并且提供了处理异常的机制

1. 异常参数

异常也可以有参数,异常引发后它会被传递给异常处理器

当异常被引发后参数是作为附加帮助信息传递给异常处理器的

1. else子句

在try范围中没有异常被检测到时,执行else子句

在else范围中的任何代码运行前,try范围中的所有代码必须完全成功

1. finally语句

finally子句是无论异常是否发生,是否捕捉都会执行的一段代码

如果打开文件后,因为发生异常导致文件没有关闭,可能会发生数据损坏。使用finally以保证文件总是能正常的关闭

课堂练习

简化除法判断

1. 提示用户输入一个数字作为除数

2. 如果用户按下Ctrl+C或Ctrl+D则退出程序

3. 如果用户输入非数字字符,提示用户应该输入数字

4. 如果用户输入0,提示用户0不能作为除数

# try:

# user = int(input('number:'))

# result = 100 / user

# # print(result)

# except ValueError:

# print('\033[31;1m输入错误\033[0m')

# except ZeroDivisionError:

# print('\033[31;1m输入错误\033[0m')

# except KeyboardInterrupt:

# print('\033[31;1m\n Bye-bye\033[0m')

# except EOFError:

# print('\033[31;1m\n Bye-bye\033[0m')

#

# else:

# print(result) #不发生异常才执行的代码

# print('\033[32;1mdone\033[0m')

# try:

# user = int(input('number:'))

# result = 100 / user

# # print(result)

# except (ValueError,ZeroDivisionError):

# print('\033[31;1m输入错误\033[0m')

# except (KeyboardInterrupt,EOFError):

# print('\033[31;1m\n Bye-bye\033[0m')

#

# else:

# print(result) #不发生异常才执行的代码

# finally:

# print('\033[32;1mdone\033[0m') #不管异常是否发生都要执行的代码

#

# #不是必须把所有的语句写全，常用的有try-except 和 try-finally 组合

自定义异常

1. 编写第一个函数,函数接收姓名和年龄,如果年龄 不在1到120之间,产生ValueError异常

2. 编写第二个函数,函数接收姓名和年龄,如果年龄不在1到120之间,产生断言异常

# def set\_age(name,age):

# if not 0 < age < 120:

# raise ValueError('年龄超过范围')

# print('%s is %d years old.' % (name,age))

#

# def set\_age2(name,age):

# assert 0 < age < 120,'年龄超过范围' #不在范围内报错

# print('%s is %d years old.' % (name,age))

#

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

# set\_age('bob',23)

# set\_age2('bob',233)

1. os模块



# import os

#

# os.getcwd() #pwd

# os.listdir() #ls

# os.listdir('/tmp/mydemo') # ls /tmp/mydemo

# os.mkdir('/tmp/mydemo') #mkdir

# os.chdir('/tmp/mydemo') #cd

#

# os.mknod('mytest') #touch mytest

# os.symlink('/etc/hosts','/zhuji') # ln -s /etc/hosts zhuji

# os.makedirs('aaa/bbb/ccc') #mkdir -p aaa/bbb/ccc

# os.path.isfile('zhuji') #判断文件是否存在

# os.path.isdir('/tmp') #判断目录是否存在

# os.path.exists('/abc') #路径是否存在

# os.path.abspath('zhuji') #返回路径

# os.path.basename('/tmp/abc/aaa.txt') #显示文件名

# os.path.dirname('/tmp/abc/aaa.txt') #显示目录名，文件前面的目录名

# os.path.split('/tmp/abc/aaa.txt') #目录和文件分开

# os.path.splitext('/tmp/abc/aaa.txt') #('/tmp/abc/aaa', '.txt')文件和后缀分离

# os.path.join('/tmp','abc') #'/tmp/abc'

#

# os.remove('mytest')

# os.rmdir('aaa/bbb/ccc') #删除空目录

# os.chdir('aaa',777) #权限

#

1. pickle模块

#import pickle as p

#shop\_file = '/tmp/shop.txt'

#shop\_list = ['apple','eggs','banana']

#with open(shop\_file,'wb') as fobj:

# p.dump(shop\_list,fobj) #将列表写入文件

# with open(shop\_file,'rb') as fobj:

# mylist = p.load(fobj) #从文件取的数据仍然是列表

#

# mylist[1]