第一章 初识Python

1.6 课后习题

（1）高级语言翻译为机器语言的方式有两种:一种是编译，一种是解释。Python属于解释型语言

（2）Python程序支持两种运行方式:交互式和脚本式

（3）Python的单行注释以 # 符号作为开始符,该符号后面的文字都是注释

（4）input函数的功能是接收标准输入数据（即从键盘输入)，返回为 String 类型(字符串)

（5）print函数的功能是将各种类型(字符串、整数、浮点数、列表、字典等）的数据输出到屏幕上

（6）IDLE有两种窗口模式: Shell和 Editor（编辑器)。

（7）下面不属于Python 语言的优点的是（C）

A.简单易学 B.免费开源 C.强制缩进 D.丰富的库

（8）有关Python的注释叙述正确的是（B）

A.单行注释以分号“;”开始

B.多行注释以3个单引号“'''”或3个双引号“"""”作为开始符和结束符

C.多行注释以3个“#”作为开始符和结束符

D.注释也是程序代码,参与程序的运行

（9)已知语句r= eval (input("请输入一个有效的表达式:"))，则运行情况错误的是（D）

A.如果输入4+7，通过print(r)可得到结果11

B.如果输入4\*2.5+10，通过 print(r)可得到结果20

C.如果输入5\*/3，则会因其不是一个有效的表达式而报 SyntaxError错误

D.如果输入3+5，通过 print(r)可得到结果3+5

（10）简述Python语言的优点

优点：简单易学：语法简单、代码量少；

免费开源：源代码开放；

跨平 台性：可移植到 Linux、Windows、Macintosh、Android 等平台上；

高层语言： 忽略底层细节；

面向对象：符合人类认识客观世界的方式；

丰富的库：提供了内 在标准库以及丰富的第三方库；

胶水语言：像使用胶水一样把用其他编程语言编写的模块粘合过来等优点

（11）简述Python在科学计算、数据库编程、 Web 编程和人工智能4个领域的应用

科学计算：从1997年起，NASA就大量使用Python进行各种复杂的科学运算。随着NumPy、SciPy、Matplotlib、Enthought、librarys等众多程序库的开发，使得Python越来越适合进行科学计算并绘制高质量的二维和三维图像

数据库编程：程序员可通过遵循Python DB-API（数据库应用程序编程接口)规范的模块与Microsoft SQL Server、 Oracle、 Sybase、 DB2、MySQL、SQLite等数据库通信。另外,Python自带一个Gadfly模块,提供了完整的SQL环境

Web编程：Python拥有很多免费数据函数库、免费Web网页模板系统以及与Web服务器进行交互的库，可以实现Web开发，搭建Web框架。用 Python开发的 Web项目小而精,支持最新的 XML技术，而且数据处理功能较为强大

人工智能：在神经网络、深度学习方面,Python都能够找到比较成熟的包来加以调用另外。Python是面向对象的动态语言,且适用于科学计算

第二章 Python的基础语法

2.7 课后习题

（1）变量是指在程序运行讨程中值可以发生改变的量

（2）已知s="Python语言程序设计”，则print(s[2:4])的输出结果为th，print(s[-4:-2])的输出结果为程序

（3）已知 t=(3.5，2，'abcd' ，4+5j，True，[3,3.5]，5.3),print(t[3])的输出结果为4+5j，print(t[-3])的输出结果为True

（4）10/4的结果为2.5，10//4的结果为2，10%4的结果为2，10\*\*4的结果为10000

（5）已知x=50，则10<=x and x<=30的结果为False

（6）已知x,y=4,5,则x|y的结果为5，x^y的结果为1

（7）下面选项中,正确的变量名是（C）

A.2sum B.for C.圆面积2 D.it is

（8）执行Python语句“name,age='张三',20”之后，下面说法正确的是（C）

A.name的值为“张三”，age的值为20，两个变量的类型不一定

B.程序报错,因为两个变量没有定义,不能直接赋值

C.定义两个变量，name是字符串类型,值为“张三”,age是整型，值为20

D.不能同时给两个变量赋值,程序报错

（9）已知语句:a,b,c=12,0o12,0x12，则 print(a,b,c)的输出结果是（A）

A.12 10 18 B.12 12 12 C.10 8 6 D.12 18 10

（10）已知a={10,2.5, 'test',3+4j, True, 5.3,2.5},则 print(a)的输出结果是(B）

A.{10,2.5, 'test',3+4j,True,5.3,2.5} B.{True,2.5,5.3,10,3+4j, 'test'}

C.10 2.5 'test' 3+4j True 5.3 2.5 D. True 2.5 5.3 10 3+4j 'test'

（11）print("姓名:%5s，年龄:%5d，成绩:%6.2f"%("tom",19,86.5))的输出结果是（D）(注:选项中口表示一个空格)

A.姓名:tom,年龄:19,成绩:86.5

B.姓名:tom口口,年龄:19口口口,成绩:86.50口

C.姓名:口口 tom,年龄:口口口19,成绩:86.5

D.姓名: 口口tom,年龄:口口口19,成绩:口86.50

（12）已知x,y=10,[10,20,30],则x is y和x in y的结果分别为（D）

A. True True B. False False C.True False D. False True

（13）写出下面程序的运行结果

s1,s2="abc","def"

z1,z2=[1,2,"zhang"],[2.2,3.3,"wang"]

x1=[1,2.5,'test',3+4j,True,[3,1.63],5.3]

print(s1+s2)

print(z1+z2)

print(s1\*3)

print(z1[:])

print(x1[:3])

print(x1[3:-1])

#输出结果

abcdef

[1, 2, 'zhang', 2.2, 3.3, 'wang']

abcabcabc

[1, 2, 'zhang']

[1, 2.5, 'test']

[(3+4j), True, [3, 1.63]]

（14）1已知程序段的功能是用户输入数字1~7，输出对应的星期几的字符串，如输入4,输出星期四。请将程序填写完整

week="星期一星期二星期三星期四星期五星期六星期日"

n=eval(input("请输入星期数字(1~7)"))

pos=(n-1)\*3

print(week[pos:pos+3])

（15）通过设置条件，可以使某些语句在条件满足时才会执行

（16）通过循环，可以使得某些语句重复执行多次

（17）下面的程序段循环次数为11，循环结束后i的值为-1

i=10

while i>=0:

i-=1

print(i)

（18）已知程序段：

score=eval(input('请输入成绩（0-100之间的整数）：'))

if score<60:

print('不及格')

elif score<70:

print('及格')

elif score<80:

print('中等')

elif score<90:

print('良好')

elif score<=100:

print('优秀')

若输入77，则输出结果为中等

（19）已知程序段：

score=eval(input('请输入成绩（0-100之间的整数）：'))

if score<60:

print('你的成绩是%d'%score)

print('不及格')

若输入55，则输出结果是（A）

A.你的成绩是55

不及格

B.你的成绩是55

C.不及格

D.无输出

（20）已知程序段：

score=eval(input('请输入成绩（0-100之间的整数）：'))

if score>=60:

pass

else:

print('不及格')

若输入55，则输出结果是（B）

A.无输出

B.不及格

C.pass

D.c程序报错

（21）已知程序段：

n=eval(input('请输入一个整数:'))

if n%2==0:

print("偶数")

else:

print("奇数")

若输入-5，则输出结果是（B）

A.无输出

B.奇数

C.偶数

D.偶数

奇数

（22）已知语句段：

d={'Python':1,'C++':2,'Java':3}

for k in d:

print('%s:%d'%(k,d[k]))

则输出结果是（C）

A.Python

C++

Java

B.1：Python

2：C++

3：Java

C.Python:1

C++:2

Java:3

D.以上都不对

（23）下面程序段的输出结果是（D）

ls=['Python','C++','Java']

for k,v in enumerate(ls,3):

print(k,v)

A.Python

C++

Java

B.1 Python

2 C++

3 Java

C.Python 1

C++ 2

Java 3

D.3 Python

4 C++

5 Java

(24)已知程序段的功能是用户输入数字n，利用for循环求n!。请将程序填写完整

n=eval(input('请输入一个大于0的整数：'))

s=1

for i in range(1,n+1):

s=s\*i

print(s)

（25）下面程序的功能是求100以内能被7整除的最大整数，请将程序填写完整

n=100

while n>=0:

if n%7==0:

print(n)

break

n-=1

（26）判断素数的程序，请将程序填写完整

for n in range(2,101):

m=int(n\*\*0.5)

i=2

while i<=m:

if n%i==0:

break

i+=1

if i>m:

print(n,end=' ')

（27）水仙花数是3位整数（100~999），它的各位数字立方和等于该数本身。下面的程序求水仙花数，请写出程序运行结果

for n in range(100,1000):

bai=n//100

shi=n//10%10

ge=n%10

if bai\*\*3+shi\*\*3+ge\*\*3==n:

print(n)

#输出结果

153

370

371

407

（28） 下面的程序输出九九乘法表，请将程序补充完整

for i in range(1,10):

for j in range(1,i+1):

print(j,"\*",i,"=",i\*j,end=' ')

print('\n')

第三章 函数

3.7 课后习题

（1）在Python语言中，使用函数分为两个步骤：定义函数和调用函数

（2）在Python语言中，函数定义需要使用def关键字

（3）形参是在定义函数时函数后面的一对小括号中给出的参数列表

（4）实参是在调用函数时函数后面的一对小括号中给出的参数列表

（5）能够将一个函数的运算结果返回到调用函数的位置，以便可以继续用该运算结果参与其他运算，此时则应使用return语句

（6）使用import将整个模块导入，也可以使用from import将模块中的标识符直接导入当前环境

（7）可以适用 “ from 模块名 import \* ”的方式导入一个模块中的所有标识符

（8）定义一个包，就是创建一个文件夹并在该文件夹下创建一个\_init\_.py文件，文件夹的名字就是包名

（9）猴子补丁是指在运行时动态替换已有的代码，而不需要修改原始代码

（10）下列说法中错误的是（B）

A.当调用函数时，如果没有为某些形参传递对应的实参，则这些形参会自动使用默认参数值

B.在使用关键字参数调用函数时，实参的传递顺序必须与形参列表中形参的顺序一致

C.当普通实参传递给形参后，如果在函数体中对形参值做修改，则该修改并不会影响实参，即实参值不会改变

D.如果实参是列表等对象，那么可在函数体中通过形参修改实例列表中对应元素的值

（11）下列关于拆分参数列表的说法正确的是（A）

A.列表、元组拆分出来的结果作为位置参数，字典拆分出来的结果作为关键字参数

B.列表、元组和字典拆分出来的结果都作为关键字参数

C.列表、元组和字典拆分出来的结果都作为位置参数

D.列表、元组拆分出来的结果作为关键字参数，而字典拆分出来的结果作为位置参数

（12）写出下面程序段的输出结果

def ModifyVal(x,y):

z = x

x = y

y = z

def ModifyListElement(ls,idx,val):

ls[idx] = val

a,b = 3,5

print(a,b)

ModifyVal(a,b)

print(a,b)

c = [7,8,9]

print(c)

ModifyListElement(c,1,6)

print(c)

#输出结果

3 5

3 5

[7, 8, 9]

[7, 6, 9]

（13）写出下面程序段的输出结果

def Cir(radius,color = 'red'):

print('半径：%d，颜色：%s'%(radius,color))

Cir(10)

Cir(20,color='green')

Cir(color='blue',radius=30)

#输出结果

半径：10，颜色：red

半径：20，颜色：green

半径：30，颜色：blue

（14）函数sum的功能是求参数的和并返回，参数个数不限。请将程序填写完整

def Sum(\*args):

s=0

for i in args:

s += i

return s

print(Sum(1,2,3,4,5))

（15）已知模块文件arithmetic.py的内容如下：

def plus(x,y):

return x+y

def multi(x,y):

return x\*y

编写程序，将模块导入，调用其中的函数，请将程序填写完整

使用import导入：

import arithmetic

print(arithmetic.plus(3,5))

print(arithmetic.multi(5,6))

使用from import依次导入函数：

from arithmetic import plus,multi

print(plus(3,5))

print(multi(5,6))

使用from import导入所有标识符：

from arithmetic import \*

print(plus(3,5))

print(multi(5,6))

（16）按照作用域的不同，Python中的变量可以分为局部变量和全局变量

（17）在一个函数中使用global关键字，可以声明在该函数中使用的是全局变量，而非局部变量

（18）通过nonlocal关键字，可以使内层的函数直接使用外层函数中定义的变量

（19）递归函数是指在一个函数内部通过调用自己来求解一个问题

（20）lambda函数也称为匿名函数，是一种不适用def定义函数的形式，其作用是能快速定义一个简短的函数

（21）lambda函数的函数体只是一个表达式，所以lambda函数通常只能实现比较简单的功能

（22）如果内层函数使用了外层函数中定义的局部变量，并且外层函数的返回值是内层函数的引用，就构成了闭包

（23）下列说法中错误的是（D）

A.在一个函数中定义的变量就是局部变量

B.局部变量的作用域是从定义位置到函数结束位置

C.在所有函数外定义的变量就是全局变量

D.全局变量的作用域是从定义位置到程序结束位置

（24）下列关于修饰器的说法错误的是（C）

A.利用装饰器，可以在不修改已有代码的情况下向已有函数中注入代码，使其具备新的功能

B.一个装饰器可以为多个函数注入代码，一个函数也可以注入多个装饰器的代码

C.一个装饰器可以为多个函数注入代码，而一个函数只能注入一个装饰器的代码

D.装饰器实际上就是 一种闭包

（25）写出下面程序的输出结果

def outer():

x=100

global y

y=200

def inner():

nonlocal x

x=1000

y=2000

print('inner函数中的x,y值为',x,y)

inner()

print('outer函数中的x,y值为',x,y)

x=10

y=20

outer()

print("执行outer,inner函数之后x,y值为",x,y)

#输出结果

inner函数中的x,y值为 1000 2000

outer函数中的x,y值为 1000 200

执行outer,inner函数之后x,y值为 10 200

（26）写出下面闭包程序的输出结果

def f(x):

y = 100

def inner(z):

return x \* y + z

return inner

a10 = f(10)

a20 = f(20)

print(a10(29))

print(a20(29))

#输出结果

1029

2029

（27）已知高阶函数的程序如下

def FunAdd(f,x):

return f(x)

def area(r):

return 3.14\*r\*r

def perimeter(r):

return 2\*3.14\*r

print(FunAdd(area,5))

print(FunAdd(perimeter,5))

请将其中的area函数和perimeter函数改写为lambda函数。请将改写后的程序填写完整

def FunAdd(f,x):

return f(x)

print(FunAdd(lambda r:3.14\*r\*r,5))

print(FunAdd(lambda r:2\*3.14\*r,5))

(28)请编写递归函数fib(n),实现斐波那契数列第n项的值

def fib(n):

if n <= 1:

return n

else:

return (fib(n - 1) + fib(n - 2))

num = int(input("输入想要输出的第几项："))

if num < 0:

print("输入正数")

elif num == 0:

print("0")

else:

print("斐波那契数列:")

for i in range(num):

print(fib(i+1))

第四章 面向对象

4.5 课后习题

（1）类和对象的关系即数据类型与变量的关系，类规定了可以用于存储什么数据，而对象用于实际存储数据,每个对象可存储不同的数据

（2）类中包含属性和方法。属性对应一个类可用来保存哪些数据，而方法对应一个类可以支持哪些操作（即数据处理)

（3）私有属性，是指在类内可以直接访问，而在类外无法直接访问的属性

（4）构造方法是Python类中的内置方法之一,它的方法名为\_init\_，在创建一个类对象时会自动执行，负责完成新创建对象的初始化工作

（5）析构方法是类的另一个内置方法，它的方法名为\_del\_，在销毁一个类对象时会自动执行，负责完成待销毁对象的资源清理工作，如关闭文件等

（6）\_str\_方法的返回值必须是字符串

（7）通过继承，可以基于已有类创建新的类，新类除了继承已有类的所有属性和方法，还可以根据需要增加新的属性和方法

（8）通过多态，可以使得在执行同一条语句时，能够根据实际使用的对象类型决定调用哪个方法

（9）如果一个类C1通过继承已有类C而创建，则将C1称作子类，将C称作基类、父类或超类

（10）如果一个子类只有—个父类，则将这种继承关系称为单继承；如果一个子类有两个或更多父类，则将这种继承关系称为多重继承

（11）方法重写是指子类可以对从父类中继承过来的方法进行重新定义，从而使得子类对象可以表现出与父类对象不同的行为

（12）内置函数isinstance用于判断一个对象所属的类是否是指定类或指定类的子类

（13）内置函数issubclass用于判断一个类是否是另一个类的子类

（14）内置函数type用于获取一个对象所属的类

（15）Python提供了\_slots\_变量以限制可动态扩展的属性

（16）Python提供了@property装饰器，可以将类中属性的访问和赋值操作自动转为方法调用，这样可以在方法中对属性值的取值范围做一些条件限定

（17）直接使用@property 可以定义一个用于获取属性值的方法（即getter）。如果需要对属性score定义一个设置属性值的方法（setter)，需要用到的装饰器为@属性名.setter

（18）元类可以看成创建类时所使用的模板，也可以理解为用于创建类的类

（19）单例模式是指在程序运行时确保某个类最多只有一个实例对象

（20）在执行\_init\_前,还会执行类的一个内置的类方法\_new\_，其作用是创建对象并返回

（21）关于类和对象说法正确的是（A）

A.根据一个类可以创建多个对象，而每个对象只能是某一个类的对象

B.根据一个类只能创建一个对象，而每个对象只能是某一个类的对象

C.根据一个类可以创建多个对象，而每个对象可以属于多个类

D.根据一个类只能创建一个对象，而每个对象可以属于多个类

（22）下列关于构造方法的说法错误的是（C）

A.构造方法是 Python类中的内置方法之一，在创建一个类对象时会自动执行，负责完成新创建对象的初始化工作

B.构造方法中,除了self，也可以设置其他参数

C.构造方法的参数只能是self

D.构造方法也可以设置默认参数

（23）进行self>=other运算时自动执行的内置方法是（C）

A.\_gt\_ (self,other) B.\_It\_(self,other) C. \_ge\_ (self,other) D. \_le\_(self,other)

（24）关于super 方法的说法错误的是（D）

A.super 方法用于获取父类的代理对象以执行已在子类中被重写的父类方法

B. super方法有两个参数，第一个参数是要获取父类代理对象的类名

C.在一个类A的定义中调用super方法时，可以将两个参数都省略，此时，super()等价于super(A, self）

D.第二个参数必须传入对象名，该对象所属的类必须是第一个参数指定的类或该类的子类，找到的父类对象的self 会绑定到这个对象上

（25）关于类方法的说法错误的是（D）

A.类方法是指使用@classmethod修饰的方法

B.类方法的第一个参数是类本身（而不是类的的实例对象)

C.类方法既可以通过类名直接调用，也可以通过类的实例对象调用

D.类方法只能通过类名直接调用

（26）关于静态方法的说法错误的是（B）

A.静态方法是指使用 @staticmethod 修饰的方法

B.静态方法的第一个参数是类本身（而不是类的实例对象)

C.静态方法既可以通过类名直接调用，也可以通过类的实例对象调用

D.静态方法中没有类方法中的第一个类参数

（27）已知 Complex类的类方法add(cls,c1,c2)实现两个复数的相加运算,程序可以使用不同的方式调用该方法，下面错误的调用方式是（C）

A.c3=Complex.add(c1,c2) B..c3=c1.add(c1,c2)

C.c3.add(Complex,c1,c2) D.c3=Complex().add(c1,c2)

（28）关于动杰扩展类的说法正确的是（A）

A. Python除了可以在定义类时定义属性和方法外，还可以动态地为已经创建的对象绑定新的属性和方法

B.Python 只能在定义类时定义属性和方法，不能动态扩展

C.Python只能动态扩展属性，不能动态扩展方法

D.Python只能动态扩展方法，不能动态扩展属性

（29）写出下面程序的输出结果

class Student:

name='Unknown'

age=0

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

print('姓名：',Student.name)

print('年龄：',Student.age)

stu=Student()

print('stu姓名：%s,年龄：%d'%(stu.name,stu.age))

Student.name='张三'

Student.age=18

print('stu姓名：%s,年龄：%d'%(stu.name,stu.age))

stu.name='李四'

stu.age=19

print('stu姓名：%s,年龄：%d'%(stu.name,stu.age))

Student.name='张三'

Student.age=18

print('stu姓名：%s,年龄：%d'%(stu.name,stu.age))

#输出结果

姓名： Unknown

年龄： 0

stu姓名：Unknown,年龄：0

stu姓名：张三,年龄：18

stu姓名：李四,年龄：19

stu姓名：李四,年龄：19

（30）写出下面程序的输出结果

class Person:

def \_\_init\_\_(self,name):

self.name=name

def Display(self):

print(self.name)

class Student(Person):

def \_\_init\_\_(self,sno,name):

super().\_\_init\_\_(name)

self.sno=sno

def Display(self):

super(Student,self).Display()

print(self.sno)

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

per=Person("张三")

per.Display()

stu=Student(78,"李四")

stu.Display()

#输出结果

张三

李四

78

（31）写出下面程序的输出结果

class Person:

pass

class Student(Person):

pass

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

per=Person()

stu=Student()

print("stu所属类是Person类的子类",isinstance(stu,Person))

print("student所属类是Person类的子类",issubclass(Student,Person))

print("per所属类是Student类的子类",isinstance(per,Student))

print("Person类是Student类的子类",issubclass(Person,Student))

print('stu对象所属的类：',type(stu))

print('per所属的类：',type(per))

#输出结果

stu所属类是Person类的子类 True

student所属类是Person类的子类 True

per所属类是Student类的子类 False

Person类是Student类的子类 False

stu对象所属的类： <class '\_\_main\_\_.Student'>

per所属的类： <class '\_\_main\_\_.Person'>

（32）写出下面程序的输出结果

class Cylinder:

Radius=5

Height=10

def Display(self):

print("圆柱体半径为",self.Radius)

class Circle:

Radius=50

def Display(self):

print("圆半径为",self.Radius)

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

c1=Cylinder()

c2=Circle()

c1.Display()

c2.Display()

#输出结果

圆柱体半径为 5

圆半径为 50

（33）下面程序对圆柱体类Cylinder实现动态绑定方法，请将程序填写完整

from types import MethodType #从types模块导入MethodType方

class Cylinder:

pass

def SetRadius(self,r):

self.Radius=r

def SetHeight(self,h):

self.Height=h

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

c1=Cylinder()

c2=Cylinder()

c1.SetRadius=MethodType(SetRadius,c1)

Cylinder.SetHeight=SetHeight

c1.SetRadius(10)

c2.SetRadius(20) #c2对象未绑定SetRadius方法，此语句有误

c1.SetHeight(30)

c2.SetHeight(40)

（34）定义Circle类，要求：包括私有属性\_radius，构造函数为半径赋值，构造函数的默认参数值为0，析构函数输出适当信息，普通方法SetRadius用于设置半径，普通方法Area返回圆面积，内置方法\_str\_用于输出圆面积，内置方法\_gt\_用于比较两个圆面积大小，并创建两个实例分别验证上述功能

PI=3.14

class Circle:

\_\_radius=0

def \_\_init\_\_(self, r=0):

self.\_\_radius=r

def SetRadius(self,r):

self.\_\_radius=r;

def \_\_str\_\_(self):

return '面积：'+str(self.\_\_radius\*PI\*PI)

def \_\_del\_\_(self):

print('半径为',self.\_\_radius,"的圆被销毁")

def Area(self):

return self.\_\_radius\*self.\_\_radius\*PI

def \_\_gt\_\_(self,other):

return self.Area()>other.Area()

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

c1=Circle()

c2=Circle(10)

c1.SetRadius(5)

print(c1)

print(c2)

if(c1>c2):

print("1>2")

else:

print("1<=2")

del c1

del c2

第五章 序列、集合和字典

5.8 课后习题

（1）Python 中，通过列表中的 index 方法可以根据指定值查找第一个匹配的列表元素 的位置。

（2）Python 中，通过列表中的 insert 方法可以将一个元素插入到列表的指定位置。

（3）若在列表的最后添加新元素，则可以直接使用列表的 append 方法。

（4）使用 del 语句可以删除某个变量或列表中的某个元素。

（5）使用 max 方法可以获取一个列表中最大元素的值。

（6）使用 min 方法可以获取一个列表中最小元素的值。

（7）使用列表中的 count 方法可以统计某个值在列表中出现的次数。

（8）Python 中集合有两种插入元素的方法，分别是 add 和 update。

（9）集合中的 issubset 方法可以用于判断一个集合是否是另一个集合的子集。

（10）集合中的 issuperset 方法可以用于判断一个集合是否是另一个集合的父集。

（11） 使用 del 可以删除某个元素，也可以使用字典中的 pop 方法删除指定键的元素。

（12）使用字典中的 clear 方法可以一次将一个字典中的所有元素都清除。

（13）判断字典中是否存在某个键可以使用字典中的 get 方法，也可以使用成员运算符 in。

（14）已知 ls=[x\*2 for x in range(5)]，则 print(ls)的输出结果为[0,2,4,6,8]。

（15）迭代器是指可以通过 next 函数不断获取下一个值的对象。

（16） 下列属于可变类型的是（A）

A. 列表 B. 元组 C. 字符串 D. 数字

（17）下列叙述正确的是（B）

A. 列表和元组都是用一对中括号括起来的多个元素的有序集合，各元素之间用逗号分隔

B. 列表是用一对中括号括起来的多个元素的有序集合，各元素之间用逗号分隔，元组是用一对小括号括起来的多个元素的有序集合，各元素之间用逗号分隔

C. 列表是用一对小括号括起来的多个元素的有序集合，各元素之间用逗号分隔，元组是用一对中括号括起来的多个元素的有序集合，各元素之间用逗号分隔

D. 列表和元组都是用一对小括号括起来的多个元素的有序集合，各元素之间用逗号分隔

（18）关于复制列表元素说法错误的是（C）

A. Python 中，通过赋值运算实际上是将两个变量指向同一个对象，而不是将一 个变量的值赋给另一个变量

B. 采用元素截取的方法，可以根据一个已有列表对象复制出另一个新的列表对 象，后面对两个对象的操作完全独立

C. 如果一个列表中包含列表类型的元素，元素截取的方法完全可行，两个列表中 的相同列表类型的元素完全独立

D. 为了真正实现列表的复制操作，可以使用 Python 在 copy 模块中提供的 deepcopy 函数

（19）已知学生类中有属性 name 和 sno，列表 ls 中含有若干学生对象，若要求列表 ls 按照学生的姓名的降序排序，相应的语句是（A）

A. ls.sort(key=lambda stu:stu.name,reverse=True)

B. ls.sort(key=lambda stu:stu.name)

C. ls.sort(key=name,reverse=True)

D. ls.sort(name)

（20）下列叙述错误的是（C）

A. 可以使用一对大括号{}或 set 函数创建集合

B. 可以使用一对大括号{}或 dict 函数创建字典

C. 可以使用{}或 set()创建空集合

D. 可以使用{}或 dict()创建空字典

（21）已知定义： d=dict(sno='1810100', name='李晓明', age=19,t=(3,4),s=[3,4,5])， 则 len(d)的值为（A）

A. 5 B. 6 C. 7 D. 8

（22） 已知字典 d，获取字典中键值集合的语句是（C）

A. d.items() B. d.values() C. d.keys() D. d.get()

（23）下列有关生成器叙述错误的是（D）

A. 将列表生成表达式中的一对中括号改为一对小括号即可得到生成器

B. 对于生成器对象，也可以像其他可迭代对象一样使用 for 循环遍历对象中的每 一个元素

C. 如果生成元素的方法比较复杂，不适合用 for 循环方式实现，我们还可以借助 yield 关键字利用函数实现生成器的功能

D. 生成器就是一次性在内存中产生大量列表元素，占用大量的内存空间

（24）关于说法正确的是（B）

A. 列表是可迭代对象、但不是迭代器；同样，生成器是可迭代对象、但不是迭代 器

B. 列表是可迭代对象、但不是迭代器；而生成器既是可迭代对象、又是迭代器

C. 列表既是可迭代对象、又是迭代器；而生成器是可迭代对象、但不是迭代器

D. 列表既是可迭代对象、又是迭代器；而生成器是迭代器、但不是迭代对象

（25）写出下面程序的运行结果。

t1=("zhangsan",18,95.5)

t2=()

t3=(33,)

t4=([44,55,66])

t5=t1+t3

print(t1,t2,t3,t4,t5)

print(t5)

print(max(t4))

print(min(t4))

#输出结果

('zhangsan', 18, 95.5) () (33,) [44, 55, 66] ('zhangsan', 18, 95.5, 33)

('zhangsan', 18, 95.5, 33)

66

44

（26）写出下面程序的运行结果。

ls=[1,2,3,2,3,4]

print(ls.index(4))

print(ls.count(3))

print(max(ls))

print(min(ls))

print(len(ls))

del ls[3]

print(ls)

ls.sort(reverse=True)

print(ls)

#输出结果

5

2

4

1

6

[1, 2, 3, 3, 4]

[4, 3, 3, 2, 1]

（27）写出下面程序的运行结果。

import copy

s1=[4,5,6]

s2=s1

s2[1]="nk"

print(s1,s2)

s3=s1[1:3]

s3[1]="cn"

print(s1,s3)

s4=copy.deepcopy(s1)

s4[1]=333

print(s1,s4)

s5=[4,[5,6]]

s6=s5[1:3]

s5[1][0]="cn"

print(s5,s6)

s7=copy.deepcopy(s5)

s7[1]="nk"

print(s5,s7)

#输出结果

[4, 'nk', 6] [4, 'nk', 6]

[4, 'nk', 6] ['nk', 'cn']

[4, 'nk', 6] [4, 333, 6]

[4, ['cn', 6]] [['cn', 6]]

[4, ['cn', 6]] [4, 'nk']

（28）写出下面程序的运行结果。

s1={1,2,3}

s2=set([2,3,3,4])

s1.add(3)

s2.update('ab')

s3=s1.intersection(s2)

s4=s1.union(s2)

s5=s1.difference(s2)

s6=s1.symmetric\_difference(s2)

print(s1)

print(s2)

print(s3)

print(s4)

print(s5)

print(s6)

#输出结果

{1, 2, 3}

{2, 3, 4, 'b', 'a'}

{2, 3}

{1, 2, 3, 4, 'b', 'a'}

{1}

{1, 4, 'a', 'b'}

（29）写出下面程序的运行结果。

d1={'name':"zhangsan",'sno':"001",'score':99}

d2=dict().fromkeys(['radius','height'],0)

print(d1)

print(d2)

d1.update({'age':19})

d2.update(radius=10)

print(d1)

print(d2)

del d1['age']

height=d2.pop('height','not found')

print(d1)

print(d2)

print(height)

print(len(d1))

print(len(d2))

#输出结果

{'name': 'zhangsan', 'sno': '001', 'score': 99}

{'radius': 0, 'height': 0}

{'name': 'zhangsan', 'sno': '001', 'score': 99, 'age': 19}

{'radius': 10, 'height': 0}

{'name': 'zhangsan', 'sno': '001', 'score': 99}

{'radius': 10}

0

3

1

（30）写出下面程序的运行结果。

s1=[1,2,3,4,5,6,7,8,9,10]

s2=list(range(10,20))

s3=s1[2:8:2]

s4=s2[-8::-2]

print(s1)

print(s2)

print(s3)

print(s4)

#输出结果

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

[3, 5, 7]

[12, 10]

（31）该程序用到字典的浅拷贝和深拷贝，已知程序运行结果，请将程序填写完整。

import copy

d1={'name':"zhangsan",'sno':"001",'score':{'math':99,'C++':88}}

d2=d1.copy()

d3=copy.deepcopy(d1)

d1['name']="li"

d1['score']['C++']=90

print(d1)

print(d2)

print(d3)

已知程序运行结果为：

{'name': 'li', 'sno': '001', 'score': {'math': 99, 'C++': 90}}

{'name': 'zhangsan', 'sno': '001', 'score': {'math': 99, 'C++': 90}}

{'name': 'zhangsan', 'sno': '001', 'score': {'math': 99, 'C++': 88}}

第六章 字符串

6.5 课后习题

（1）Python 中,创建字符串时,可以使用单引号、双引号和三引号

（2）Python 中,使用字符串的 split 方法可以按照指定的分隔符对字符串进行切割，返回由切割结果组成的列表

（3）可以利用运算符“+”连接两个字符串,也可以 join 使用方法将序列中的元素以指定的字符连接成一个新的字符串

（4）Python 中对正则表达式中的普通字符需要做精确匹配，而特殊字符指定的匹配模式则对应了用于模糊匹配的规则

（5）Python 中正则表达式的特殊序列是由 \ 和一个字符构成

（6）使用 Python提供的Re模块,可以实现基于正则表达式的模糊匹配

（7）compile函数用于将一个字符串形式的正则表达式编译成一个正则表达式对象，供match、search 以及其他函数使用

（8）re模块中的match函数用于对字符串开头的若干字符进行正则表达式的匹配。匹配成功，返回一个Match对象；匹配失败,返回None

（9）re模块中的 finditer函数与re.findall函数功能都是用于在字符串中找到所有与正则表达式匹配的子串,区别在于re.findall 函数返回列表形式的结果,而 re.finditer返回迭代器形式的结果。

（10）re模块中的split函数用于将字符串按与正则表达式匹配的子串分割

（11）已知s1="I"am a student.",s2='I \'am a student'，则 print(s1,s2)的输出结果为（D）

A. I 'am a student. I 'am a student B.I "am a student.I 'am a student

C. I "am a student. I \'am a student D.程序报错

（12）下列说法错误的是（D）

A.find方法是在指定检索范围中按照从左至右的顺序检索,找到子串第一次出现的位置

B.rfind方法在指定检索范围中按照从右至左的顺序检索,找到子串第一次出现的位置

C. index 与find作用相同，rindex与rfind作用相同，只是find和rfind 在检索不到字符串时返回-1,而index和 rindex会引发ValueError 异常

D.index与 find作用相同,rindex 与 rfind作用相同，只是find和rfind找到一个就返回,而index和rindex会检索到所有的子串

（13)下列关于replace方法str.replace(old, new[,max])的说法错误的是（C）

A.使用字符串中的replace方法可以将字符串中的指定子串替换成其他内容

B.str是要做替换操作的字符串，old和new分别是要替换的子串和替换成的字符串

C. max是最多替换的子串数量，如果不指定max参数,则只替换第一个满足条件的子串

D. replace方法返回替换后的字符串

（14）print(len(”中国\"china" "))的输出结果是（B）

A.7 B.8 C.9 D.10

（15）下列匹配模式叙述错误的是（D）

A.“^”用于匹配字符串开头的若干字符 B.“\*”用于匹配前一个模式0次或多次

C.“?”用于匹配前一个模式0次或1次 D.A|B表示同时匹配模式A和模式B

（16）关于正则表达式特殊序列叙述错误的是（C）

A.A匹配字符串开头的若干字符,功能同“^”

B.d 匹配任一数字字符,等价于[0-9]

C.\S匹配任一空白字符

D.\D与\d作用相反，匹配任一非数字字符，等价于[^0-9]

（17）写出下面程序的运行结果

print("ad" > "abcd")

print("AD" > "abcd")

print("AD" > "ADC")

print("tianjin" > "beijing")

print("天津" > "北京")

#输出结果

True

False

False

True

True

（18）写出下面程序的运行结果

s1="I am a student."

s2="C++90分Python88分Java85分"

s3="I am a student.\nI like programming.\n"

print(s1.split())

print(s2.split("分"))

print(s2.split("分",2))

print(s3.splitlines())

#输出结果

['I', 'am', 'a', 'student.']

['C++90', 'Python88', 'Java85', '']

['C++90', 'Python88', 'Java85分']

['I am a student.', 'I like programming.']

（19）写出下面程序的运行结果

str=" a b c "

print("123"+str.strip()+"456")

print("123"+str.lstrip()+"456")

print("123"+str.rstrip()+"456")

#输出结果

123a b c456

123a b c 456

123 a b c456

（20）写出下面程序的运行结果

n,f = 34,56.78

print("%d,%o,%x"%(n,n,n))

print("%f,%e"%(f,f))

r = 5

s = 3.14\*r\*r

s1 = "半径为{0}的圆面积为{1}"

s2 = "半径为{radius}的圆面积为{area}"

print(s1.format(r,s))

print(s2.format(area = s,radius=r))

#输出结果

34,42,22

56.780000,5.678000e+01

半径为5的圆面积为78.5

半径为5的圆面积为78.5

（21）写出下面程序的运行结果

import re

pattern = re.compile(r'Student',re.I)#生成正则表达式对象

r1 = pattern.match('Students study programming')

r2 = pattern.match('I am a student! ',3)

r3 = pattern.match('I am a student! ',7)

r4 = re.search(r'Student','I am a student',re.I)

r5 = re.match(r'Student','I am a student',re.I)

print(r1)

print(r2)

print(r3)

print(r4)

print(r5)

#输出结果

<re.Match object; span=(0, 7), match='Student'>

None

<re.Match object; span=(7, 14), match='student'>

<re.Match object; span=(7, 14), match='student'>

None

（22）写出下面程序的运行结果

import re

str = '''sno:#1810101#,name:#李晓明#,age:#19#,major:#计算机#

sno:#1810102#,name:#马红#,age:#20#,major:#数学#'''

rlt = re.search(r'sno:#([\s\S]\*?)#[\s\S]\*?major:#([\s\S]\*?)#',str,re.I)

if rlt:#判断是否有匹配结果

print('匹配到的整个字符串:',rlt.group())

print('sno:%s,startpos:%d,endpos:%d'%(rlt.group(1),rlt.start(1),rlt.end(1)))

print('major:%s,startpos:%d,endpos:%d'%(rlt.group(2),rlt.start(2),rlt.end(2)))

print('所有分组匹配结果:',rlt.groups())

else:

print('未找到匹配信息')

#输出结果

匹配到的整个字符串: sno:#1810101#,name:#李晓明#,age:#19#,major:#计算机#

sno:1810101,startpos:5,endpos:12

major:计算机,startpos:41,endpos:44

所有分组匹配结果: ('1810101', '计算机')

（23）写出下面程序的运行结果

import re

html = '''%abc%%def%python(ghi)'''

content = re.sub(r'%[\s\S]\*%','&',html)

content = content.strip()

print('替换之后的内容为',content)

content2 =re.subn(r'%[\s\S]\*%','&',html)

print('替换之后的内容及替换次数为',content2)

#输出结果

替换之后的内容为 &python(ghi)

替换之后的内容及替换次数为 ('&python(ghi)', 1)

（24）下面的程序从键盘输入一个字符串，然后将其中的大写字母转换为小写字母，小写字母转换为大写字母，其他字符不变，请将程序填写完整

str = input("请输入一个字符串")

ns=''

for c in str:

if c>='A' and c<='Z':

ns+=c.lower()

elif c>='a'and c<='z':

ns+=c.upper()

else:

ns+=c

print(ns)

第七章 I/O编程与异常

7.7 课后习题

（1）利用 os 模块查看当前系统的名字，应当使用 os.name 。

（2）利用 os 模块获取当前系统平台路径的分隔符，应当使用 os.sep 。

（3） 利用 os 模块获取当前工作目录，应当使用 os.getcwd 方法。

（4）利用 os 模块创建目录，可以使用 os.mkdir 方法或 os.makedirs 方法。

（5）判断指定路径目标是否是文件的方法是 os.path.isfile 。

（6）os 模块中判断指定路径是否存在的方法是 os.path.exists 。

（7）根据文件对象的 closed 属性可以判断文件是否已关闭。

（8）使用文件对象的 seek 方法可以移动文件指针，从而实现文件的随机读写。

（9）使用 writer 对象的 writerow 方法或 writerows 方法可以向 CSV 文件中写入数据。

（10）os 模块中用于依次创建路径中所有不存在的目录的方法是（ A ）。

A. makedirs B. makedir C. mkdirs D. mkdir

（11）下面说法中，错误的是（ B ）。

A. 如果要创建的目录已经存在，则 os.mkdir 函数会报错

B. 如果要创建的目录已经存在，则 os.makedirs 函数不会报错

C. 如果要删除的目录不存在，则 os.rmdir 函数会报错

D. 如果要删除的目录已存在但目录不为空，则 os.rmdir 函数会报错

（12）os 模块中用于删除指定路径的最后多层目录的方法是（ A ）。

A. removedirs B. removedir C. rmdirs D. rmdir

（13） open 函数的默认打开方式是（ C ）。

A. w B. w+ C. r D. r+

（14）下面文件文件打开方式中，不能对打开的文件进行写操作的是（ C ）。

A. w B. wt C. r D. a

（15）要从文件中按行读取所有数据，则应使用（ D ）方法。

A. read B. readall C. readline D. readlines

（16）无论 try 子句执行时是否发生异常，都会执行的子句是（ B ）。

A. else B. finally C. except D. 不存在

（17）写出下面程序的运行结果。

for i in range(3):

try:

num=(i+1)\*5

assert num%2!=0

print(num)

except AssertionError:

print('断言失败！num=%d'%num)

#输出结果

5

断言失败！num=10

15

（18）写出下面程序的运行结果。

import os

for n in os.path.split('D:\\mydir\\subdir1\\test.txt'):

print(n)

#输出结果

D:\mydir\subdir1

test.txt

（19）下面程序在 D 盘的 mydir 目录下创建一个名字为 test.txt 的文件并向文件中写入字符串“南开大学”，请将程序填写完整。

with open('D:\\mydir\\test.txt','w+') as f:

f.write('南开大学')

（20）下面程序在 D 盘的 mydir 目录下创建一个名字为 score.csv 的文件，并将 2 名学生的 3 门课程成绩写入到文件中，请将程序填写完整。

import csv #导入 csv 模块

data2D=[[90,98,87], #第 1 名学生的 3 门课程成绩

[70,89,92]] #第 2 名学生的 3 门课程成绩

with open('D:\\mydir\\score.csv','w',newline='') as f: #打开文件

csvwriter=csv.writer(f)

csvwriter.writerow(['语文','数学','英语']) #先将列标题写入 CSV 文件

csvwriter.writerows(data2D) #将二维列表中的数据写入 CSV 文件

（21）下面的 UserError 是自定义的异常类，请将程序填写完整。

class UserError(BaseException):

def \_\_init\_\_(self,msg): #定义构造方法

self.msg=msg

def \_\_str\_\_(self): #将 UserError 类对象转换为字符串时自动调用

return self.msg