《Python 程序设计（微课版）》课后习题答案

单元 1　认识 Python

一、填空题

1. Python 是一种__面向对象的解释型__计算机程序设计语言。

2. Python 3.x 自带官方集成开发环境是___IDLE_____。

3. Python 程序源文件的扩展名是___.py_____。

二、思考题

1.列举三个常用的 Python 集成开发环境。

答：PyCharm、Eclipse（with PyDev）、Visual Studio Code

2.思考并理解 Python 语言的特性。

答：Python 是一种面向对象的解释型程序设计语言，语法简洁、可读性强。Python 提供了非常完善的基础代码库，覆盖了网络、文件、GUI、数据库、文本等大量内容。用 Python 开发，许多功能不必从零编写，直接使用现成的即可。除了内置的库外，Python 还有大量的第三方库，也就是别人开发的，供你直接使用的东西。Python 是跨平台的，它可以运行在 Windows、Mac 和各种 Linux/Unix 系统上。在 Windows 上写 Python 程序，放到 Linux 上也是能够运行的。

单元 2　变量和简单数据类型

一、填空题

1.布尔值是整型的子类，用数值 1 和 0 代表常量__True_ 和___False_ 。

2.表达式由___运算符__、__操作数__和圆括号组成。

二、如果 a = 1、b = 2、c = 0，写出下列表达式的逻辑值：

1. a > b or a + b < c

False

2. not (a > b)

True

3. a – b < c and not c

True

三、编程题

1. 输入矩形的长与宽，计算矩形面积。

```
length = int(input("请输入矩形的长："))
width = int(input("请输入矩形的宽："))
area = length * width
print("矩形面积为：",area)
```

单元 3　流程控制

一、填空题

1.Python 中的流程控制语句有_顺序结构__、_分支结构_和_循环结构_。

2._continue_语句用于跳出本次循环，进入下一次循环。

二、写出下列程序的运行结果

1. 请使用字符串格式化的方式写一个新年问候语模板。

```
a = 0
if a:
    print(“Hello”)
else:
    print(“World”)
```

结果：World

2. 将一串字符串中以.com 或.cn 为域名后缀的 URL 网址匹配出来，过滤掉其他无关信息，如： 百度 。

```
i = sum = 0
while i <= 4:
        sum += i
        i = i+1
print(sum)
```

结果：10

三、编程题

1. 编程输出九九乘法表。

```
1x1=1
1x2=2  2x2=4
1x3=3  2x3=6  3x3=9
1x4=4  2x4=8  3x4=12  4x4=16
1x5=5  2x5=10  3x5=15  4x5=20  5x5=25
1x6=6  2x6=12  3x6=18  4x6=24  5x6=30  6x6=36
1x7=7  2x7=14  3x7=21  4x7=28  5x7=35  6x7=42  7x7=49
1x8=8  2x8=16  3x8=24  4x8=32  5x8=40  6x8=48  7x8=56  8x8=64
1x9=9  2x9=18  3x9=27  4x9=36  5x9=45  6x9=54  7x9=63  8x9=72  9x9=81
```

```
for i in range(1, 10):
    for j in range(1, i+1):
        print(str(i) + "x" + str(j) + "=" + str(i * j), end=" ")
    print()
```

单元 4　字符串与正则表达式

一、填空题

1.使用__len()__函数可以查看字符串的长度。

2.字符串中从左往右的第一个字符索引为__0__。

3.表达式"helloworld"[-5:]的结果为__world__。

4.表达式"helloworld"[-5]的结果为__w__。

5.正则表达式模块 re 的__search()__方法用来在整个字符串中进行指定模式的匹配。

二、编程题

1.请使用字符串格式化的方式写一个新年问候语模板。

```
template="--*"*7+"\n\t 节 日 快 乐\t\n{0}:\n\t 祝您{1}快乐！ \n\t\t\t{2}\n"+"--*"*7
print(template.format("李老师","春节","李梅"))
```

```
--*--*--*--*--*--*--*
    节 日 快 乐
李老师：
    祝您春节快乐！
            李梅
--*--*--*--*--*--*--*
```

2.将一串字符串里面以.com 或.cn 为域名后缀的 URL 网址匹配出来，过滤掉其他无关信息。如：百度

```
import re
str1 = "<a href='www.baidu.com'>百度</a>"
pattern = "(http://){0,1}[a-zA-Z0-9.]*(com|cn)"
result = re.search(pattern,str1)
if result:
    print(result.group())
else:
    print("not matched")
```

单元 5　组合数据类型

一、简单题

1.能否直接修改字符串的某个字符？例如 s="abc"，s[0]="1" 可以吗？

答：在 Python 中，字符串是不可变类型，即无法直接修改字符串的某一位字符。

2. 元组和列表有什么不同？

答：list 是处理一组有序项目的数据结构，即你可以在一个列表中存储一个序列的项目。列表中的项目。列表中的项目应该包括在方括号中，这样 python 就知道你是在指明 5261 一个列表。一旦你创建了一个列表，你就可以添加，删除，或者是搜索列表中的项目。由于你可以增加或删除项目，我们说列表是可变的数据类型，即这种类型是可以被改变的元祖和列表十分相似，不过元组是不可变的。即你不能修改元组。元组通过圆括号中用逗号分隔的项目定义。元组通常用在使语句或用户定义的函数能够 1653 安全的采用一组值的时候，即被使用的元组的值不会改变。

二、编程题

1. 设计一个字符串函数 reverse(s)，返回字符串 s 的反串，例如 reverse("abc")返回 "cba"。

def reverse(s):

```
    if len(s) <1:
        return s
    return reverse(s[1:])+s[0]
r = reverse(s)
print(r)
```

2. 用一个字典描述一个日期，包含年（year）、月（month）、日（day）的键字。

```
today_time = datetime.datetime.now()
today_time_dict = {'year':today_time.year,'month':today_time.month,'day':today_time.day}
```

3. 写出下列程序执行的结果

```
Traceback (most recent call last):
  File "C:/Users/Everdeen/Desktop/123.py", line 4, in <module>
    for k3 in d2.keys():
NameError: name 'd2' is not defined
```

单元 6　Python 函数与模块

1.编程计算 1+2+4+…+100。

```
sum=0
for i in range (1,101):
    sum=sum+i
print (sum)
```

2. 从键盘输入一个字符串，直到按 Enter 键结束，编程统计字符串中的大小写英文字母各有多少个。

python 中输入字符串复,统计字符串中制大小写 2113 英文字母 5261 各有多少 4102 个？

```
class countNum(object):
def __init__(self):
    s = input("input a string")
    self.s = s
```

```
def judge(self):
    bigger = 0
    smaller = 0
    for i in self.s:
        if i < 'A' or i > 'z':
            continue
        elif 'A' < i < 'Z':
            bigger += 1
        elif 'a' < i < 'z':
            smaller += 1
        return bigger, smaller
if __name__ == '__main__':
p = countNum()
big, small = p.judge()
print("大写字母 1653%d 个" % big)
print("小写字母%d 个" % small)
```

3. 小华今年 12 岁，她妈妈比她大 20 岁，编写程序计算多少年后她妈妈年龄比她大一倍。

```
y=1
while y<150:
    if 32+y==2*(12+y):
        print(y)
    y+=1
```

4. 假设世界人口是 60 亿，如果每年按 1.5%的比例增长，编写程序计算经过多少年后世界人口可以达到 80 亿。

```
x = 60
year = 1
while 1:
    x *= 1.015
    if x >= 80:
```

```
            print(x,year)
            break
        else:
            year += 1
```

5. 一个小球从 80 米高空自由下落，每次落地后返回原高度的一半，再落下。编写程序计算小球在第 10 次落地时共经过多少米，以及第 10 次反弹有多高。

```
high = 80
n = 10
high_all = 0 #第 n 次落地时走过的长度
high_each = 0 #每次落地的高度内
def ball_lands(n):
    global high_all, high_each, high
    #小球第一次落地时
    if n == 0:
        high_each = high
        high_all += high_each
        #print('1', high_each, high_all)
        return high_each
    #从小球第 n 次落地往前容推
    high_each = high * (1 / 2) ** n
    print(u'第%d 次小球弹起的高度为：%.2f' %(n, high_each))
    high_all += high_each * 2
    #print('2', high_each, high_all)
    ball_lands(n - 1)

ball_lands(n)
print(u'小球落地%d 次，共经过%.2f 米。' % (n,high_all))
```

单元 7　Python 面向对象

1. 定义一个数学中的复数类 Complex，该类有一个构造函数和一个显示函数，建立一个 Complex 对象，调用设计的显示函数并显示。

```
class Complex:
    def __init__(self, a):
        self.a = a

    def display(self):
        print(self.a)



com = Complex("aaa")
com.display()
```

2. 定义一个计算机类 MYComputer，该类包含 CPU 类型（String 类型）、RAM 内存大小（Integer 类型）、HD 硬盘大小（Integer 类型），设计它的构造函数，再设计一个显示函数，建立一个 Complex 对象，调用设计的显示函数并显示。

```
class MyComputer:
    def __init__(self, CPU, RAM, HD):
        self.CPU = CPU
        self.RAM = RAM
        self.HD = HD

    def show(self):
        str1 = "cpu 是{}，RAM 内存大小是{}G,HD 硬盘大小是{}G".format(self.CPU, self.RAM, self.HD)
        print(str1)



cpu = "8 核 32 线程"
RAM = 8
HD = 500
mc = MyComputer(cpu, RAM, HD)
```

```
mc.show()
```

3. 设计一个整数类 MYInteger，该类有一个整数变量，并有一个 Value 属性，可以通过 Value 存取该变量的值，还有一个转二进制字符串的成员函数 toBin 和一个转十六进制字符串的成员函数 toHex。

```
class MyInteger:
    def __init__(self, num):
        self.num = num

    def Value(self):
        print(self.num)

    def toBin(self):
        print("转换为二进制为：", bin(self.num))

    def toHex(self):
        print("转换为十六进制为：", hex(self.num))

    def tootc(self):
        print("转换为八进制为：", oct(self.num))


num = 789
mi = MyInteger(num)
mi.Value()
mi.toBin()
mi.toHex()
mi.tootc()
```

4. 建立一个普通人员类 Person，该类包含姓名（m_name）、性别（m_gender）、年龄（m_age）成员变量。

（1）建立 Person 类，包含 m_name、m_sex、m_age 成员变量。

（2）建立 Person 的构造函数。

（3）建立一个显示函数 Show()，并显示该对象的数据。

（4）派生一个学生类 Student，增加班级（m_class）、专业（m_major），并设计这些类的构造函数。

（5）建立 m_class、m_major 对应的属性函数 sClass()、sMajor()。

（6）建立显示成员函数 Show()，显示该学生对象的所有成员数据。

```
class Person:
    def __init__(self, m_name, m_sex, m_age):
        self.__m_name = m_name
        self.__m_sex = m_sex
        self.__m_age = m_age

    def Show(self):
        print(self.__m_name, self.__m_sex, self.__m_age,end=" ")


class Student(Person):
    def __init__(self, m_name, m_sex, m_age, m_class, m_major):
        # Person.__init__(self, m_name, m_sex, m_age)          #  普通方法调用
        super(Student,self).__init__(m_name, m_sex, m_age)    #  使用 super 方法调用父类——__init__
        self.__m_class, self.__m_major = m_class, m_major

    def sClass(self):
        print(self.__m_class)

    def sMajor(self):
        print(self.__m_major)

    def Show(self):
        # Person.Show(self)    #  普通方法调用
```

```
        super(Student, self).Show()        # 使用 super 方法调用父类 —— Show 方法
        print(self.__m_class, self.__m_major)


stu = Student("张三", "男", "18", "高三", "理科")
stu.Show()
```

5. 建立一个时间类 Time，该类包含时（hour）、分（minute）、秒（second）的实例属性。

（1）设计时间显示函数 show(self)。

（2）设计一个时间大小比较函数 compare(self,t)，其中 t 是另外一个时间变量。

```
class Time:
    def __init__(self,hour, minute, second):
        self.hour = hour
        self.minute = minute
        self.second = second

    def show(self):
        format_time = "{}:{}:{}".format(self.hour,self.minute,self.second)
        print(format_time)

    def compare(self):
        t = "23:45:07"
        t_time = t.split(":")
        t_hour = t_time[0]
        t_minute = t_time[1]
        t_second = t_time[2]
        t_seconds = int(t_hour) * 3600 + int(t_minute) * 60 + int(t_second)
        compare_sec = abs(t_seconds - (self.hour*3600 + self.minute*60 +self.second))
        compare_hour = compare_sec//3600
        compare_minute = (compare_sec%3600)//60
```

```
        compare_second = (compare_sec%3600)%60
        print("{}:{}:{}".format(compare_hour,compare_minute,compare_second))


    time = Time(12, 30, 1)
    time.show()
    time.compare()
```

单元 8　异常与异常处理

一、单选题

1.下列程序运行以后，会产生如下（　B　）异常。

```
def test:
    print(123)
```

A. SyntaxError　B. NameError　C. IndexError　D.KeyError

2.下列选项中，（　C　）是唯一不在运行时发生的异常。

A. ZeroDivisionError　B.NameError　C.SyntaxError　D.KeyError

3.当 try 语句中没有任何错误信息时，一定不会执行（　D　）语句。

A. try　　B. else　　C.finally　　D. except

4.在完整的异常语句中，语句出现的顺序正确的是（　A　）。

A. try---->except----->else---->finally　　B. try---->else---->except----->finally

C. try---->except----->finally--->else　　D.try----->else---->else----->except

5.下列选项中，用于触发异常的是（　C　）。

A. try　　B.catch　　C.raise　　D. except

二、填空题

1．Python 中自定义异常时，用于主动抛出异常的关键字是__raise__。

2．一个 try 语句对应一个__except___子句。

3. 当使用序列中不存在　　索引　　时，会引发 IndexError 异常。

4. Python 中所有的异常类都是__BaseException___子类。

三、简答题

1. 请简述什么是异常。

程序在运行时，如果 python 解释器遇到一个错误，会停止程序的执行，并且提示一些错误信息，这就是异常。异常是一个事件，该事件会在程序执行过程中发生，影响程序的正常执行，一般情况下，在 python 中无法处理程序时就会发生异常，异常是 Python 的一个

对象，表示一个错误，当 Python 脚本发生异常时，我们需要捕获并处理异常，否则程序就会终止执行。

2. 常见的异常类型有哪些？

NameError　　未声明/初始化对象 (没有属性)

ZeroDivisionErro　除(或取模)零 (所有数据类型)

SyntaxError　　语法错误

IndexError　　序列中没有没有此索引(index)

KeyError　映射中没有此键

IOError　　输入/输出操作失败

AttributeError　未知的对象属性

ValueError　　传入无效的参数

四、编程题

编写一个计算减法的方法，当被减数小于减数时，抛出“被减数不能小于减数"的异常。

```
def jianfa(a, b): #  定义一个函数
    try: #  尝试
        if a < b: #  如果 a 小于 b
            raise BaseException('被减数{}不能小于减数{}'.format(b,a))
        else: #否则
            print(a - b) #  输出 a-b
    except BaseException as f : #
        print(f) #  触发，抛出异常
jianfa(4,5) #  第一个数 4，第二个数是 5
```

单元 9　Python 文件操作

一、单选题

1.打开一个已有文件，然后在文件末尾添加信息，正确的打开方式为（　C　）。

A. 'r'　　B. 'w'　　C.'a'　　D. 'w+'

2.假设文件不存在，如果使用 open 方法打开文件会报错，那么该文件的打开方式是下列哪种模式？（ A ）

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

3.假设 file 是文本文件对象，下列选项中，哪个用于读取一行内容？（ C ）

A. file.read() B. file.read(200)

C. file.readline() D. file.readlines()

4.下列方法中，用于向文件中写出内容的是（ B ）。

A.open B.write C.close D.read

5.下列方法中，用于获取当前目录的是（ C ）。

A.open B.write C.getcwd D.read

二、填空题

1. 打开文件对文件进行读写，操作完成后应该调用___close___方法关闭文件，以释放资源。

2.使用 readlines 方法把整个文件中的内容进行一次性读取，返回的是一个___列表___。

3.os 模块中的 mkdir 方法用于创建___目录__。

三、简答题

1. 二进制文件与文本文件有什么区别？

（1）存储数据类型不同

文本文件只能存储 char 型字符变量。二进制文件可以存储 char/int/short/long/float/等各种变量值。

（2）数据长度不同

文本文件每条数据通常是固定长度的。以 ASCII 为例，每条数据(每个字符)都是 1 个字节。进制文件每条数据不固定。如 short 占两个字节，int 占四个字节，float 占 8 个字节。

（3）读取的软件不同

文本文件编辑器就可以读写，比如记事本、NotePad++、Vim 等。二进制文件需要特别的解码器，比如 bmp 文件需要图像查看器，rmvb 需要播放器。

2. 解释一下以下三个参数的分别作用 open(f_name,'r',encoding="utf-8")

f_name 文件名

r 模式

encoding 编码方式

四、编程题

现有一个文本文件，文件内容全为英文，编写程序读取其内容，并将其中的大写字母变为小写字母，小写字母变为大写字母。

```
with open("D:\\english.txt", 'r+') as f:
    s = f.read()
    ss = [i.swapcase() for i in s]
    f.seek(0)
    f.writelines(ss)
    print(s)
```

单元 10　项目综合实训

一、编程题

电影《无双》上映后，发行商和导演为了了解用户对电影的反馈，现需要爬取猫眼平台的电影影评数据。数据爬取 URL 地址为“http://m.maoyan.com/mmdb/comments/movie/342166.json?_v_=yes&offset=0”，筛选出从影片上映日期到当前时间的影评数据，将其保存在“d:\\comments.csv”文件中，最终部分数据展示效果如下图所示。

1078586414	洪仔	2	佛山	可以看下一般	3	2019/8/16 8:38
1078600781	zqiuqi007	2	扬州	和对象看的第一	5	2019/8/16 1:57
1078604026	Alone『Alone』	2	中山	太好看了，进场	5	2019/8/16 1:23
1078591127	OIf279721527	2	广州	手法比较独特。	4.5	2019/8/15 23:02
1078592148	QJD871302483	2	东莞	发哥太帅了，一	5	2019/8/15 23:01
1078577373	Bx西西	2	淮安	还好，开头有点	4	2019/8/15 22:34

```
import requests
import json
import time
import random
import csv
from datetime import datetime, timedelta


def get_headers():
    user_agent_list = [
```

```
        "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/22.0.1207.1 Safari/537.1",

        "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1092.0 Safari/536.6",

        "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.6 (KHTML, like Gecko) Chrome/20.0.1090.0 Safari/536.6",

        "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.1 (KHTML, like Gecko) Chrome/19.77.34.5 Safari/537.1",

        "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.9 Safari/536.5",

        "Mozilla/5.0 (Windows NT 6.0) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.36 Safari/536.5",

        "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1063.0 Safari/536.3",

        "Mozilla/5.0 (Windows NT 5.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1063.0 Safari/536.3",

        "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1062.0 Safari/536.3",

        "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1062.0 Safari/536.3",

        "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3",

        "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3",

        "Mozilla/5.0 (Windows NT 6.1) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.1 Safari/536.3",

        "Mozilla/5.0 (Windows NT 6.2) AppleWebKit/536.3 (KHTML, like Gecko) Chrome/19.0.1061.0 Safari/536.3",

        "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.0.1055.1 Safari/535.24",

        "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/535.24 (KHTML, like Gecko) Chrome/19.0.1055.1 Safari/535.24"

    ]

    user_agent = random.choice(user_agent_list)

    headers = {'User-Agent': user_agent}

    return headers
```

```
def get_data(url):
    headers = get_headers()
    try:
        with requests.Session() as s:
            response = s.get(url, headers=headers, timeout=3)
            content = response.text
            return content
    except Exception as e:
        print(e)


# 处理数据
def parse_data(html):
    try:
        data = json.loads(html)['cmts']  # 将 str 转换为 json
    except Exception as e:
        return None
    comments = []
    for item in data:
        comment = [item['id'], item['nickName'], item["userLevel"], item['cityName'] if 'cityName' in item else '',
                   item['content'].replace('\n', ' '), item['score'], item['startTime']]
        comments.append(comment)
    return comments


# 存储数据
def save_to_csv():
    start_time = datetime.now().strftime('%Y-%m-%d %H:%M:%S')  # 获取当前时间，从当前时间向前获取
    end_time = '2019-08-12 00:00:00'  # 影片的上映日期
    while start_time > end_time:  # 如果时间开始时间大于结束时间
```

```
            url = 'http://m.maoyan.com/mmdb/comments/movie/342166.json?_v_=yes&offset=0&startTime=' + start_time.replace(
                ' ', '%20')
            html = None
            try:
                html = get_data(url)
            except Exception as e:
                time.sleep(0.5)
                html = get_data(url)
            else:
               time.sleep(1)
            comments = parse_data(html)
            if comments:
               start_time = comments[14][-1]  # 获得末尾评论的时间
               start_time = datetime.strptime(start_time, '%Y-%m-%d %H:%M:%S') +
          timedelta( seconds=-1)  # 转换为 datetime 类型，减 1 秒，避免获取到重复数据
          start_time = datetime.strftime(start_time,'%Y-%m-%d %H:%M:%S')  # 转换为 str

                print(comments)

                with open("d:\\comments.csv", "a", encoding='utf-8', newline='') as csvfile:
                    writer = csv.writer(csvfile)
                    writer.writerows(comments)
    if __name__ == '__main__':
        save_to_csv()
```