py26过年期间python基础复习题

一、填空题

1、Python 3.x默认使用的编码是：**utf-8。**

2、在循环体中使用 **break** 语句可以跳出循环体。

3、在循环体中可以使用 **continue** 语句跳过本次循环后面的代码，重新开始下一次循环。

4、如果希望循环是无限的，我们可以通过设置条件表达式永远为 **True** 来实现无限循环。

5、函数可以有多个参数，参数之间使用 **逗号** 分隔

6、使用 **return** 语句可以返回函数值并退出函数。

二、判断题

- **1、Python使用符号#表示单行注释。（对）**
- **2、标识符可以以数字开头。（错）**
- **3、type()方法可以查看变量的数据类型。（对）**
- **4、Python中标识符不区分大小写。（错）**
- **5、Python中的标识符不能使用关键字。（对）**
- **6、函数的名称可以随意命名。（错）**
- **7、不带return的函数代表返回None。（对）**
- **8、函数定义完成后，系统会自动执行其内部的功能。（错）**
- **9、在使用异常时必须先导入exceptions模块。（错）**
- **10、一个try语句只能对应一个except子句。（错）**
- **11、所有的except语句一定在else和finally的前面。（对）**
- **12、类方法的第一个参数为self (错)**
- **13、对象不能调用类方法 （错）**

三、选择题

1.下列选项中，（D）的布尔值不是Flase。

A.None B.0 C.() D.1

2.下列标识符中，合法的是（A）

A.helloWorld B.2ndObj C.hello#world D.hell-oworld

3.字符串'Hi,Andy'中，字符'A'对应的下标位置为（C）。

A. 1 B.2 C.3 D.4

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.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.finaly D. except

三、编程题

1、 用户登陆程序需求:

1. 输入用户名和密码;

2. 判断用户名和密码是否正确? (name='python', passwd='lemonban')

3. 为了防止暴力破解， 登陆仅有三次机会， 如果超过三次机会， 提示错误次数过多，账号已被冻结,;

```
# 第一题参考答案:
def login():
    for i in range(1, 4):
        user = input('请输入用户名:')
        pwd = input('请输入密码:')
        if user == 'python' and pwd == 'lemonban':
            print('登录成功')
            break
        else:
            print('账号密码错误，您还有{}次机会'.format(3 - i))
    else:
        print('今日的三次机会已用完，错误次数过多，账号已被冻结')
```

2、定义一个函数，将给定一个句子（只包含字母和空格）， 将句子中的单词位置反转，

比如： “hello xiao mi” 转换为 “mi xiao hello”

```
def str_reverse():
    str1 = "hello xiao mi"
    # 将字符串以空格为分割点，分割为列表
    str_list = str1.split(' ')
    # 将列表反转
    str_list.reverse()
    # 将反转的列表重新拼接为字符串
    res = ' '.join(str_list)
    print(res)
```

3、运行程序，提示用户依次输入三个整数x,y,z，请把这三个数由小到大输出。

```python
def number_sort():
    x = int(input('输入数字x：'))
    y = int(input('输入数字y：'))
    z = int(input('输入数字z：'))
    num_list = [x, y, z]
    # 从小到大排序
    num_list.sort()
    # 遍历输出
    for num in num_list:
        print(num)
```

4、编写一个程序，使用for循环输出0-100之间的偶数

```python
def work4():
    for i in range(101):
        if i%2 == 0:
            print(i)
```

5、打开一个文本文件，读取其内容，把其中的大写字母修改为小写字母，再写入文件覆盖原内容。

```python
def work5(file_path):
    # 以r模式打开文件
    with open(file_path,'r',encoding='utf8') as f:
        # 读取内容
        content = f.read()
        # 将内容中的大写字母转换为小写
        new_content = content.lower()
    # 以w模式打开文件
    with open(file_path,'w',encoding='utf8') as f:
        # 写入内容
        f.write(new_content)
```

6、现在有一个字符串s = 'asdf2273788hh90999',请写一段代码来去除字符串中的重复元素，然后转换为列表。

```python
# 利用集合去重
def work6():
    s = 'asdf2273788hh90999'
    # 转换为列表
    li = list(s)
    # 对列表去重，转换为列表
    li2 = list(set(li))
    print(li2)
```

7、小明有100块钱，需要买100本书（钱要刚好花完），a类数5元一本，b类书3元一本，c类书 1元2本。请计算小明有多少种购买的方式？（思路提示：for循环嵌套）

```python
# 分析：
# 100块钱最多买 20本A类书籍  100除5 = 20
# 100块钱最多买 33本B类书籍  100除3 = 33.3
```

```
# 100块最多买200本C类书籍    100/0.5 = 200
# 根据题意，要同时满足a + b + c = 100本数据，a*5+b*3+c*0.5 = 100块钱


def work7():
    count = 0
    # 遍历a类书籍的可能，最少0本，最多20本
    for a in range(100 // 5 + 1):
        # 遍历b类书籍的可能，最少0本，最多33本
        for b in range(100 // 3 + 1):
            # c类书籍等于 100-a-b
            c = 100 - a - b
            if (a*5 + b*3 + c*0.5) == 100:
                count += 1
    return count
```

8、题目：小明买了一对刚出生的兔子，兔子从出生后第3个月起每个月都生一对兔子，生的这对小兔子长到第三个月也开始生兔子（每个月生一对兔子），假如兔子都不死，问10个月后小明的兔子为多少对？（思路提示：重点在分析出兔子增长的规律，分析出规则之后通过for循环即可实现）

```
规律分析：
第一个月：1
第二个月：1
第三个月：2    # 第二个月数量+生的一对小兔子
第四个月：3    # 第三个月数量+生的一对小兔子
第五个月：5    # 第四个月数量+生的2对小兔子（最开始的兔子生一对，第三个月的也生了一对）
第六个月：8    # 第五个月数量+生的3对小兔子（最开始的，第三个月，第四个月的各生了一对）
第七个月：13   # 第六个月数量 +生的5对小兔子（最开始的，第三个月，第四个月，第五个月的两对 各生一对）
# ......以此类推，发现规律：除了第一个月和第二个月，后面每个月的兔子都等于前两个月之和。
# 解决思路：第二个月之后，计算兔子数量，需要知道前两个月的兔子数量，那么通过for循环去遍历月份，将每个月的兔子数量都保存到一个列表中，可以通过下标去列表中获取前两个月兔子的数量，然后计算本月的兔子数

def work8():
    li = []
    for i in range(1,11):
        if i == 1 or i == 2:
            # 第一个月和第二个月的兔子为1对
            num = 1
        else:
            # 当月的兔子数，等于前两个月只和
            num = li[i-2]+li[i-3]
        li.append(num)
    print('第10个月的兔子为：',li[-1])

work8()
```

9、请封装一个函数，来实现要求的数据格式转换功能（思路提示：for循环和zip函数）：

```
# 数据转换有一组用例数据如下：
cases = [
    ['case_id', 'case_title', 'url', 'data', 'excepted'],
```

```
    [1, '用例1', 'www.baudi.com', '001', 'ok'],
    [2, '用例2', 'www.baudi.com', '002', 'ok'],
    [3, '用例3', 'www.baudi.com', '002', 'ok'],
    [4, '用例4', 'www.baudi.com', '002', 'ok'],
    [5, '用例5', 'www.baudi.com', '002', 'ok'],
]
# 需要转换为以下格式：
cases02 = [{'case_id': 1, 'case_title': '用例1', 'url': 'www.baudi.com', 'data':
'001', 'excepted': 'ok'},
           {'case_id': 2, 'case_title': '用例2', 'url': 'www.baudi.com', 'data':
'002', 'excepted': 'ok'},
           {'case_id': 3, 'case_title': '用例3', 'url': 'www.baudi.com', 'data':
'002', 'excepted': 'ok'},
           {'case_id': 4, 'case_title': '用例4', 'url': 'www.baudi.com', 'data':
'002', 'excepted': 'ok'},
           {'case_id': 5, 'case_title': '用例5', 'url': 'www.baudi.com', 'data':
'002', 'excepted': 'ok'}
           ]
```

答案：

```
cases = [
    ['case_id', 'case_title', 'url', 'data', 'excepted'],
    [1, '用例1', 'www.baudi.com', '001', 'ok'],
    [2, '用例2', 'www.baudi.com', '002', 'ok'],
    [3, '用例3', 'www.baudi.com', '002', 'ok'],
    [4, '用例4', 'www.baudi.com', '002', 'ok'],
    [5, '用例5', 'www.baudi.com', '002', 'ok'],
]
def work9(datas):
    # 创建一个新的列表
    new_datas = []
    # 通过下标，获取新数据的key
    title = datas[0]
    # 遍历出第一行之外所有的数据
    for i in datas[1:]:
        # 将遍历的数据和key进行聚合打包，并转换为字典
        c = dict(zip(title, i))
        new_datas.append(c)
    # 返回转换之后的结果
    return new_datas

res9 = work9(cases)
print(转换之后的数据为：res9)
```

10、题目：企业发放的奖金根据利润提成。

利润(I)低于或等于10万元时，奖金可提10%；

利润高于10万元，低于20万元时，低于10万元的部分按10%提成，高于10万元的部分，可提成7.5%；

20万到40万之间时，高于20万元的部分，可提成5%；

40万到60万之间时，高于40万元的部分，可提成3%；

60万到100万之间时，高于60万元的部分，可提成1.5%，高于

100万元时，超过100万元的部分按1%提成，从键盘输入当月利润I，求应发放奖金总数?

```
def work10():
    price = float(input("请输入当月利润:"))
    if price < 0:
        print("您输入的金额有误！")
    elif price <= 100000:
        return price * 0.1
    elif price <= 200000:
        # 10万元以内的提成
        m1 = 100000 * 0.1
        # 超出10万部分的提成
        m2 = (price - 100000) * 0.075
        return m1 + m2
    elif price <= 400000:
        # 20万元以内的提成
        m1 = 100000 * 0.1 + 100000 * 0.75
        # 超出20万部分的提成
        m2 = (price - 200000) * 0.05
        return m1 + m2
    elif price <= 600000:
        # 40万元以内的提成
        m1 = 100000 * 0.1 + 100000 * 0.75 + 200000 * 0.05
        # 超出40万部分的提成
        m2 = (price - 400000) * 0.03
        return m1 + m2
    elif price <= 1000000:
        # 60万元以内的提成
        m1 = 100000 * 0.1 + 100000 * 0.75 + 200000 * 0.05 + 200000 * 0.03
        # 超出60万部分的提成
        m2 = (price - 600000) * 0.015
        return m1 + m2
    else:
        # 100万元以内的提成
        m1 = 100000 * 0.1 + 100000 * 0.75 + 200000 * 0.05 + 200000 * 0.03 +
400000 * 0.015
        # 超出60万部分的提成
        m2 = (price - 1000000) * 0.01
        return m1 + m2
```

11、编写一个自动售货机，运行功能如下:

1、请按下面提示，选择购买的商品

1). 可乐 2.5元 2). 雪碧 2.5元 3). 哇哈哈 3元 4). 红牛 6元 5). 脉动 4元 6). 果粒橙 3.5元

2、提示用户投币（支持1元，5元，10元）

用户输入投币金额，

用户投币金额不够商品价格，继续提示投币，

当投币超过商品价格，则返回商品和找零，然后结束程序

```
def work11():
    print("请按下面提示选择购买的商品:")
    print("1). 可乐   2.5元")
    print("2). 雪碧   2.5元")
    print("3). 哇哈哈    3元")
    print("4). 红牛   6元")
    print("5). 脉动    4元")
    print("6). 果粒橙 3.5元")
    num = input("请输入您的选项:")
    if num == "1":
        print("您购买的是可乐，需要支付金额为2.5元")
        price = 2.5
    elif num == "2":
        print("您购买的是雪碧，需要支付金额为2.5元")
        price = 2.5
    elif num == "3":
        print("您购买的是哇哈哈，需要支付金额为3元")
        price = 3
    elif num == "4":
        print("您购买的是红牛，需要支付金额为6元")
        price = 6
    elif num == "5":
        print("您购买的是脉动，需要支付金额为4元")
        price = 4
    elif num == "6":
        print("您购买的是果粒橙，需要支付金额为3.5元")
        price = 3.5
    else:
        print("您的输入有误！")
        # 输入错误的情况下使用return终止函数的运行（退出程序）。
        return

    # 定义变量money用来保存用户投币总金额
    money = 0
    while money < price:
        m = input("请输入投币金额【支持1元，5元，10元】:")
        # 判断用户投币金额是否为  1，5,10
        if m in ['1','5','10']:
            money += int(m)
        else:
            print("您投币的金额有误或者不支持该面额")

    print("您的投币金额为{}元,商品的价格为{}元,找零{}元".format(money, price, money -
price))


work9()
```

12、封装一个老师类

属性：姓名 年龄 性别 授课科目 授课班级（list类型，可以保存多个班级）

方法： 添加授课班级 、 打印老师的信息

```
class Teacher(object):
    """老师类"""

    def __init__(self, name, age, gender, course, classes):
```

```
        self.name = name
        self.age = age
        self.gender = gender
        self.course = course
        self.classes = classes

    def add_course(self, classes):
        """
        添加授课班级
        """
        self.classes.append(classes)

    def print_info(self):
        print('姓名：{}，年龄：{}，性别：{}'.format(self.name, self.age,
self.gender))
```

13、类和继承

- **要求一：定义一个游戏英雄类（Hero）**
  - **特征（属性）：名字（name）、血量（HP）**
  - **行为（方法）：技能1：移动**
- **要求二：定义一个战士类（继承英雄类）**
  - **除了上面英雄类的属性之外，还多了一个属性：攻击力（attack）和一个方法：技能2（普通攻击）**
- **要求三：定义一个法师类（继承英雄类）**
  - **除了上面英雄类的属性之外，还多了一个属性：法力值（MP）和一个方法：技能2（法术攻击）**

参考答案

```
class Hero:
    def __init__(self, name, HP):
        self.name = name
        self.HP = HP

    def skill1(self):
        print('{}使用了技能1：移动'.format(self.name))
```

```
# 写法一
class Warrior(Hero):
    def __init__(self, name, HP,attack):
        self.name = name
        self.HP = HP
        self.attack = attack

    def skill2(self):
```

```
        print('{}使用了技能2：普通攻击'.format(self.name))


# 写法二
class Warrior(Hero):
    def __init__(self, name, HP,attack):
        # name和HP属性可以调用父类的__init__来进行初始化
        super().__init__(name, HP)
        self.attack = attack

    def skill2(self):
        print('{}使用了技能2：普通攻击'.format(self.name))
```

```
# 方式一
class Master(Hero):
    def __init__(self, name, HP,MP):
        self.name = name
        self.HP = HP
        self.MP = MP

    def skill2(self):
        print('{}使用了技能2：法术攻击'.format(self.name))
# 方式二
class Master(Hero):
    def __init__(self, name, HP,MP):
        # name和HP属性可以调用父类的__init__来进行初始化
        super().__init__(name, HP)
        self.MP = MP

    def skill2(self):
        print('{}使用了技能2：法术攻击'.format(self.name))
```