2021-2022-1 python课程期末出题

- 数量：10
- 难度：适中

---

1. Hello Python

- 类型：简单输出
- 考点：转义字符

题目：

> 请输出：\hello "python"/

input:

> 无

output:

> \hello "python"/

```
print("\\hello \"python\"/")
```

2. 格式化输出

- 类型：输入输出
- 考点：循环、格式化输出

题目：

> 一行中输入不定多个数字，求平均值并保留两位小数

input:

> 1.2 3 4.56

output:

> 2.92

```
cnt = 0

for a in input().split():
    cnt += 1
    sum += float(a)

sum /= cnt

print("%.2f" % sum)
```

|

3. str.count()

- 类型：字符串方法
- 考点：str.count()、循环

题目：

> 输入不定多行字符串，以单行 `-1` 结束。输出其中一共有多少个 `banana`

input:

> banana banana banana
> banana banana
> banana
> -1

output:

> 6

```
cnt = 0

while True:
    line = input()
    if line == "-1":
        break
    cnt += line.count("banana")

print(cnt)
```

4. break

- 类型：循环
- 考点：循环、break

题目：

> 输入两个整数a、b（保证a <= b），按格式输出两个整数之间的最小素数。

input1:

> 14 15

output1:

> 14 到 15 区间中没有素数

input2:

> 2 5

output2:

> 2 到 5 区间中最小的素数是 2

```
a, b = map(int, input().split())

pri = -1
```

```
for i in range(a, b + 1):
    flag = True
    for k in range(2, i):
        if (i % k) == 0:
            flag = False
            break
    if flag:
        pri = i
        break

if pri == -1:
    print("{} 到 {} 区间中没有素数".format(a, b))
else:
    print("{} 到 {} 区间中最小的素数是 {}".format(a, b, i))
```

5. 冒泡排序

- 类型：函数
- 考点：自定义函数、函数返回值

题目：

> 输入一个序列，要求将序列使用冒泡排序的方法从小到大排序，并输出交换次数。主函数已给出，要求完成冒泡排序函数部分。

input:

> 2 3 4 5 1

output:

> 冒泡排序交换了 4 次
> 排序后结果是： [1, 2, 3, 4, 5]

```
def bubbleSort(arr):
    cnt = 0
    n = len(arr)
    for i in range(n):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                cnt += 1
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
    return cnt
```

> (题目提供)
>
> ```
> def main():
>     line = input()
>     arr = [int(n) for n in line.split()]
>     print("冒泡排序交换了 {} 次".format(bubbleSort(arr)))
>     print("排序后结果是：", arr)
>
>
> main()
> ```

6. set()

- 类型：集合
- 考点：set()、字符串大小写、字符串截取

题目：

> 输入一句话，要求按顺序输出句子中（首字母大写、其余字母小写）的单词，每个一遍。
>
> 输入保证只含有大小写的英文字母，并且一定有符合要求的单词。

input:

> James invited Paul Anthony and Wade to dinner but only Wade was available

output:

> James
> Paul
> Anthony
> Wade

```
name = set()

for item in input().split():
    if item[0].isupper() and item[1:len(item)].islower():
        if item in name:
            continue
        name.add(item)
        print(item)
```

7. 面向对象

- 类型：类
- 考点：类的定义和初始化、__str__

题目：

> 输入一个学生的基本信息，要求输出这个学生的介绍。主函数已提供，请完成类的部分。

input:

> 张三 23 男

output:

> 张三同学的年龄是23，性别是男

```
class Student:
    def __init__(self, name, age, sex):
        self.__name = name
        self.__age = age
        self.__sex = sex

    def __str__(self):
        return self.__name + "同学的年龄是" + self.__age + "，性别是" + self.__sex
```

> (题目提供)

```
def main():
 info = []
 info = input().split()
 A = Student(info[0], info[1], info[2])
 print(A)


main()
```

8. 类的继承

- 类型：类
- 考点：类的继承

题目：

> 要求定义一个父类 `Parent` 和一个继承它的派生类 `Child`。
>
> - Parent 提供有参构造方法，对 `name` 和 `job` 进行初始化
> - Child 提供有参构造方法， 要求调用父类的构造方法
> - 修改 Parent 和 Child 的返回值，使使输出如样例所示
>
> 输入两行，每行是一个人的姓名的职业。主函数部分已提供，要求完成类的定义部分。

input:

> 张三 律师
> 赵四 护士

output:

> Parent(name:张三, job:律师)
> Child(name:赵四, job:护士)
> Parent的父类是 <class 'object'>
> Child的父类是 <class '__main__.Parent'>

```
class Parent:
    def __init__(self, name, job):
        self._name = name
        self._job = job

    def __str__(self):
        return "Parent(name:" + self._name + ", job:" + self._job + ")"


class Child(Parent):

    def __init__(self, name, job):
        super().__init__(name, job)

    def __str__(self):
        return "Child(name:" + self._name + ", job:" + self._job + ")"
```

> (题目提供)
>
> ```
> def main():
> ```

```
    a = input().split()
    b = input().split()

    parent = Parent(a[0], a[1])
    child = Child(b[0], b[1])

    print(parent)
    print(child)

    print("Parent的父类是", Parent.__base__)
    print("Child的父类是 ", Child.__base__)


main()
```

9. 异常

- 类型：异常
- 考点：自定义异常

题目：

> 成绩录入时总会犯一些粗心的错误，希望你能定义一个 `Score` 类来记录分数，同时定义一个异常类 `ScoreException` 标识录入分数时可能出现的一些错误。
>
> Score类包括：
>
> - 私有数据域 score，存放分数
> - 构造方法，初始化分数，检测到错误时则抛出异常
>   - 当分数小于0时，抛出 ScoreException，输出 `Negative exception`
>   - 当分数大于100时，抛出 ScoreException，输出 `Beyond exception`
>
> ScoreException包括：
>
> - 私有数据域 message，存放异常信息
> - 构造方法，设置异常信息
>
> 主函数已提供，要求完成类的定义和异常的定义

input1:

> -1

output1:

> Negative exception

input2:

> 99

output2:

> 该学生的分数是：99

```
class ScoreException(Exception):
    def __init__(self, message):
        super().__init__()
```

```
        self.__message = message

    def __str__(self):
        return self.__message


class Score:
    def __init__(self, score):
        if score < 0:
            raise ScoreException("Negative exception")
        elif score > 100:
            raise ScoreException("Beyond exception")
        self.__score = score

    def __str__(self):
        return "该学生的分数是：" + str(self.__score)
```

> (题目提供)

```
def main():
 try:
     score = Score(int(input()))
     print(score)
 except ScoreException as ex:
     print(ex)


main()
```

10. 文件读取

- 类型：文件
- 考点：文件的读取

题目：

> 现在有一个`input.txt`的文件，里面含有多行字符串。要求你读取该文件，并判断文件中有多少行字符串包含字符串`banana`
>
> - 如果含有`banana`，则直接输出一共有多少行包含
> - 如果文件中从未出现过`banana`，则输出`wit`

input:

> 文件 input.txt

output:

> 略

```
with open("input.txt", "r") as file:
    cnt = 0;
    lines = file.readlines()
    for line in lines:
        if "banana" in line:
            cnt += 1
    if cnt == 0:
        print("without banana!")
    else:
        print(cnt)
```