

2022 年春季沙河联盟《大学计算机基础》实验指导书

实验 1 Python 基本语法

编程之前请认真阅读实验指导，了解题目要求，明确编程思路。

当在本地运行程序成功、并使用实验指导书给出的**所有**输入样例和输出样例检查无误后，再将源代码提交到在线实验平台（OJ 系统，在线评测系统）进行自动测试，以便平台自动记录你的成绩。

请在规定时间内完成所有题目并确保在 OJ 系统上通过测试，测试结果为“AC”。

在线实验平台（OJ 系统）链接：<https://www.comthinking.cn/login>

OJ 上代码提交截止时间：**本周四（2022 年 4 月 7 日）23: 55。**

此外要求按照“实验报告模板”撰写实验报告。请将实验报告和本次实验所有源代码放在以“**学校缩写+学号+姓名**”命名的文件夹中，压缩后在 OJ 的“实验报告”栏目提交。截止时间为**下周四（2022 年 4 月 14 日）23: 55。**

1. 实验目的

- (1) 熟悉 Python 编程环境，进行程序设计的基本训练。
- (2) 熟悉 Python 语言的基本语法，会编写简单 Python 程序，包括编写和运行基本的输入、输出以及算术运算程序。
- (3) 掌握调用 Python 标准库中常用函数的方法。
- (4) 掌握格式化字符串输出的方法。

2. 实验任务

实验任务 1 海伦公式

题目描述：

现在有一个三角形的三边长分别是 a , b , c 。那么它的面积可以由海伦公式计算得到：根号下 $p(p-a)(p-b)(p-c)$ ，其中 $p=(a+b+c)/2$ 。现输入三角形的三个边长，请你计算该三角形的面积。

输入：

输入共三行，分别表示边长 a, b, c 的值，均为浮点数。数据保证能够组成三角形。

输出：

为一个浮点数，表示三角形的面积，保留 1 位小数。

输入样例 1

3

4

5

输出样例 1

6.0

输入样例 2

3.3

4.4

5.5

输出样例 2

7.3

实验指导：

1. 建议按顺序做题，并认真阅读实验指导。因为相同的知识点在前面介绍后，如果在后面的题中再次使用，可能就不会再做介绍。
2. 设 a 是一个浮点数，Python 中可以使用格式化字符串输出，指定其宽度、精度和类型。以下有两种实现方法：
 - i. `print("%.2f" % a)` 输出对 a 保留两位小数后的值。其中的 2 表示精度（保留 2 位小数），f 表示浮点数。当改变 `%.2f` % a 中表示精度的数字 2 时，就可以得到 a 保留不同位数小数的结果。
 - ii. `print("{:.2f}".format(a))` 输出对 a 保留两位小数后的值。其中 2 表示精度，f 表示浮点数。当改变 `{:.2f}` 中表示精度的数字 2 时，就可以得到 a 保留不同位数小数的结果。
3. `math` 库中的 `sqrt()` 函数可以对数字进行开根号的运算，返回浮点数结果。例如首先 `import math`，然后令 `a=math.sqrt(9)`，此时 a 的值为 3.0。或

者采用幂运算也可以实现开根号的运算，例如 $a=9^{1/2}$ 。

4. 要输入一个浮点数，使用 `a=float(input())`。

注意不要把读入的数据写在 `input()` 括号内，读入的数据是在程序以外通过控制台等地方输入，再由 `input()` 接收的。

`input()` 函数括号内可以填写提示语（用一对单引号或双引号括起来），会在运行时输出到控制台作为提醒。**但在提交到 OJ 时需要删除提示语**，否则会 WA (Wrong Answer)。

实验任务 2 分饮料

题目描述：

现在有 t 毫升饮料，要均分给 n 名同学。每个杯子最多可以装 v 毫升饮料。小明想知道每名同学可以获得多少毫升饮料（精确到小数点后 3 位），以及每名同学至少需要多少个杯子（正整数）。

输入：

输入总共三行。第一行包含一个浮点数 t ，第二行包含一个整数 n ，第三行包含一个浮点数 v 。具体含义如题目描述。

输出：

输出两个数字表示答案，使用一个空格隔开。第一个数字表示每名同学可以获得的饮料体积（单位毫升，精确到小数点后 3 位）；第二个数字表示每名同学需要的杯子数目。

输入样例 1

500
2
50

输出样例 1

250.000 5

输入样例 2

640
7

90

输出样例 2

91.429 2

实验指导：

1. 当希望对一个浮点数向上取整时，可以使用 `math` 库中的 `ceil()` 函数。例如令 `a=math.ceil(0.1)`，那么此时 `a` 值为 1。注意必须先使用“`import math`”导入 `math` 库，才能调用 `math` 库里的函数。
2. 使用 `print('%0.3f' % a)` 输出对 `a` 保留三位小数后的值。

实验任务 3 行列式计算

题目描述：

众所周知，行列式计算是一个看起来简单但又较为复杂的过程，用人脑去计算太繁琐了！因此请你设计一个程序计算三阶行列式。

输入：

输入数据包含三行，依次对应三阶行列式的三行。

每行包含三个数，依次对应该行的三个数，各个数之间用一个空格分隔。

a11 a12 a13

a21 a22 a23

a31 a32 a33

输出：

输出数据包含一个数，为该行列式的值，保留两位小数。

输入样例：

11 12 13

21 22 23

31 32 33

输出样例：

0.00

实验指导：

1. 将一行中输入的空格分隔的多个数据转换成整型或浮点型，可以采用 `map` 函数

例如，当一行输入两个用空格分隔的数据时：`x,y=map(int, input().split())`

2. 三阶行列式的值可采用对角线法则计算

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{12}a_{21}a_{33} - a_{11}a_{23}a_{32}$$

3. 使用 `print("%.2f"%a)`，或者 `print("{:.2f}".format(a))`打印保留两位小数的 `a`。

实验任务 4 卖瓜

题目描述：

你是郝哥，经营着一个水果摊，你将展示自己的 n （整数）个西瓜（以输出 n 个数字 0 表示）。你将西瓜单价定为 p （浮点数），同时暗中在电子秤托盘下加上了重量为 m （浮点数）的吸铁石。此时，有一个人前来买瓜，你为他挑选了一个自身重量为 15 斤的西瓜，请输出他所需支付的“抹零”后的金额（向下取整）。同时，这位买家十分精明，如果他发现你的西瓜单价高于 2 ($p > 2$) 且使用了吸铁石 ($m > 0$)，他就会投诉你。为了解最后你是否会被投诉，请输出对应的布尔值（True 表示会被投诉，False 则表示不会被投诉）。

输入：

输入共三行，分别对应西瓜个数 n 、西瓜单价 p 、吸铁石重量 m 。

输出：

输出共三行，分别对应西瓜展示（以若干个数字 0 表示）、买家需支付总金额（向下取整）、你是否会被投诉（布尔值 True/False）。

输入样例：

```
5
2.5
```

1.25

输出样例：

00000

40

True

样例解释：

你有 $n=5$ 个西瓜，因此展示为 00000。

买家挑选的西瓜重量固定为 15 斤，则西瓜总价为

$p*(m+15)=2.5*(1.25+15)=40.625$ ，向下取整后的结果为 40。因为 $p=2.5>2$ 的同时 $m=1.25>0$ ，故会被投诉，则最后一行输出为布尔值的 True。

实验目的：

知晓输出重复若干次的字符（串）的简单方法；熟悉 Python 格式化字符串输出；了解布尔类型以及逻辑运算。

实验指导：

1. `print('a'*n)`可以同时打印 n 个字符'a'。其中 n 为正整数。
2. 向下取整可以使用 Python 的 `math` 库中的 `floor()`函数。简单起见，也可以使用 Python 内建的取整函数 `int`。
3. Python 中的逻辑运算符包括：`and`（布尔“与”），`or`（布尔“或”），`not`（布尔“非”）。

思考：西瓜单价高于 2 ($p>2$) 且使用了吸铁石 ($m>0$)，应使用何种逻辑运算符？

实验任务 5 奶牛吃草（选做）

题目描述：

奶牛 Brown 总是在寻找合适的就餐处。今天，它走到了一条小路，路牌上写着小路的名字。这条路有两侧，左侧是一片空地，右侧也是一片空地。Brown 用眼睛估算了两侧空地的长草情况，现在它想把这条路的情况记在笔记本上。

具体来说，它想记录下这条路的名字、适合就餐的位置（只有当一个位置的两侧都长草时，Brown 才会认为这是一个合适的就餐处）和可以就餐的位置（当一个

位置的两侧至少有一侧长草时，Brown 认为这是一个可以就餐的位置)。

但 Brown 是一头奶牛，不方便写字，所以它请你来帮忙。

输入：

输入一共有两行；

第一行是一个长度不超过 10 的字符串，表示小路的名字；

第二行包括两个以空格分隔的十进制数，表示小路左、右两侧的长草情况（此数若用二进制表示，为 1 的位置表示长草，为 0 的位置表示不长草）。

输出：

输出三行。第一行是一个字符串，即小路的名字；

第二行是一个十进制数，表示适合就餐的位置（此数如果用二进制表示，为 1 的位置表示适合就餐，为 0 的位置不适合）；

第三行是一个十进制数，表示可以就餐的位置（此数如果用二进制表示，为 1 的位置表示可以就餐，为 0 的位置不可以）。

输入样例：

Lane

3 5

输出样例：

Lane

1

7

样例解释：

第二行输入的第一个数 3 表示小路左侧长草情况，其对应的二进制数为 011，从左往右数，第 2 位和第 3 位为 1，表示这两个位置长草；输入的第二个数 5 表示小路右侧长草情况，其对应的二进制数为 101，从左往右数，第 1 位和第 3 位均为 1，表示这两个位置长草。

则根据题意，两个输入进行按位与运算，有：011&101=001，可见某个位置左侧和右侧都为 1 的只有第三个位置，故只有最右边的位置适合就餐。而只要一个位置的两侧至少有一侧长草，则认为可以就餐，即只要两个输入的十进制数所对应的二进制数某一位为 1，该位置就可以就餐。两个输入进行按位或运算，有：011|101=111，故三个位置都可以就餐。所以适合就餐的位置情况是 001，十进制表示为 1；可以就餐的位置情况是 111，十进制表示为 7。

实验目的：

考查对二进制、逻辑运算的理解，以及对输入输出的掌握。

实验指导：

1. 可以用 `input().split()` 来输入用空格分隔的数据。数据较少时，用 `a,b,c = input().split()`，按照空格将输入的多个数据分开，分别赋给等号左边的变量；当一行输入的数据很多时，可以采用 `s = input().split()`，直接将输入的所有数据存入一个列表 `s` 中，以便后续处理，但列表中的每个元素是字符串。

可以使用 `map()` 函数将输入的数据从字符串转化成整数，用法如下：

`map(function, iterable, ...)`

其中 `function` —— 函数（如 `int`, `float`），`iterable` —— 一个或多个序列。

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

2. Python 支持数的位运算。位运算就是对两个十进制数的二进制表示的每一位分别做逻辑运算，返回结果的十进制表示。`&`是按位与运算，`|`是按位或运算。例如：

$9 \& 7 = 1$ ，即 $0b1001 \& 0b0111 = 0b0001 = 1$

$9 | 7 = 15$ ，即 $0b1001 | 0b0111 = 0b1111 = 15$

3. 如果你想把一个十进制数转化成二进制数（当然在本题代码中是不需要这样做的），可以采用 Python 内置函数 `bin()` 来转化。例如，`bin(9)` 返回一个二进制字符串“0b1001”，即 9 的二进制表示是 1001。

4. 关于逻辑运算，我们知道 $(1\&1)=1$, $(1\&0)=0$, $(0\&0)=0$, $(1|1)=1$, $(1|0)=1$, $(0|0)=0$ 。想一想，按位与运算和按位或运算与 Brown 对“适合就餐”和“可以就餐”的定义是不是有异曲同工之妙呢？