实验练习02 参考答案

1. 判断BMI指数

```
w = input('请输入您的体重（千克）')
h = input('请输入您的身高（米）')
w = float(w)  # input函数返回的均是字符串类型，需要将变量转换类型才能进行数学运算
h = float(h)

bmi = w/(h**2)  # **运算符为幂运算符

'''
与C/C++语言不同，Python不使用大括号表示一个语句块，取而代之的是缩进
且C/C++语言中的else if 在Python中简写为elif
屏幕输出的print函数有两种常用写法：类C/C++写法和format函数法，下方代码均有示例
'''

if bmi < 18.5:
    print('您的BMI指数为%.2f，体重过轻。' % bmi)  # 类C/C++写法
elif bmi < 24:
    print('您的BMI指数为{:.2f}，体重正常。'.format(bmi))  # format函数法
elif bmi < 28:
    print('您的BMI指数为%.2f，体重超重。' % bmi)
else:
    print('您的BMI指数为%.2f，肥胖。' % bmi)
```

---

2. 打印乘法表

```
for i in range(1, 10):
    for j in range(1, i + 1):
        print('%d * %d = %-2d' % (j, i, i * j), end = '  ')
    print()
```

上述代码中，`for i in range(1, 10)`等同于C语言的`for (i = 1; i < 10; ++i)`

`range`方法会生成一个迭代器，for 语句在每次循环中会依次将迭代器的值赋给循环变量，具体可参考这里

`print`方法在输出完毕所有内容后默认再输出一个换行符，在内层循环中我们需要继续输出后续内容，所以需要设置`end`参数，这里修改为两个空格

---

3. 简易数据处理

方法1

```
data = [42.5066, 42.1662, 41.5372, 34.6255, 37.3276, 39.8443, 42.1836, 43.9896,
33.8646, 29.7325, 25.0662, 30.7486, 50.2659, 45.9638, 46.231, 45.1657, 46.3536,
35.5503, 39.0717, 35.6499, 36.1918, 27.7638, 30.3279, 21.3128, 22.5103, 20.8208,
28.4853, 40.9561, 33.0661, 27.646, 44.8478, 37.008, 31.3161, 33.9328, 32.8175,
0.0, 35.4657, 15.6781, 21.3214, 22.86, 34.7026, 35.1135, 38.3306, 39.1602,
33.9511, 34.5887, 36.5907, 38.8869, 40.4551, 41.9876, 47.3987, 42.8304, 41.8136,
41.4267, 41.1339, 43.0457, 43.1788, 42.9944, 43.1197, 43.6908, 44.028, 44.2258,
44.2353, 43.0444, 41.0781, 40.7676, 39.8084, 41.101, 41.3097, 42.3416, 39.4119,
30.624, 34.8342, 36.3268, 40.5735, 41.3087, 42.5451, 41.8595, 42.7566, 43.2405,
40.9939, 41.5584, 43.4783, 32.5032, 33.7413, 33.2457, 21.4866, 20.5409, 19.9374,
19.0859, 18.5459, 18.2533, 18.2445, 17.6128, 16.3594, 13.7795, 14.7944, 9.1407,
9.2812, 9.7828, 10.2063, 11.266, 12.2981, 13.8566, 13.7167, 15.6883, 18.6352,
2.2823, 2.3855, 18.805, 20.673, 23.9261, 24.6246, 30.2574, 3.3165, 18.6829,
14.3383, 0.0, 12.6839, 27.2058, 32.4048, 13.4586, 27.2213, 28.3622, 24.3946,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 5.2039, 0.0, 0.0, 25.0867, 13.6438,
18.1579, 13.744, 0.0268, 0.0, 9.7793, 20.8685, 19.2211, 2.5308, 16.9071, 9.3399,
0.0, 0.0, 0.0, 13.0796, 6.2657, 21.3445, 0.8795, 0.0, 11.6431, 0.0, 25.0044,
26.3105, 0.1709, 3.0508, 9.3071, 20.5849, 16.8375, 28.4828, 19.3334, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 18.7642, 0.0, 0.0, 5.7915, 13.4646, 20.9461, 17.2267,
16.0981, 16.2664, 7.0976, 21.1133, 13.9825, 14.6345, 18.0536, 21.7614, 24.5763,
36.0805, 49.4293, 24.6263, 29.2808, 14.3954, 42.4788, 52.5958, 43.4743, 52.0785,
51.5324, 47.749, 49.079, 44.6604, 47.4435, 45.071, 49.2676, 45.8031, 55.5259,
38.7524, 50.6233, 43.689, 37.0371, 51.842, 49.2054, 46.1528, 44.3969, 34.4268,
37.1501, 30.9934, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.4096, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.237, 18.0189, 26.9652, 0.0, 5.2103, 11.0394, 20.6039, 23.3864, 22.0404,
20.4137, 23.5754, 21.9315, 38.3556, 40.0982, 39.8685, 40.265, 40.3801, 21.7365,
41.6788, 25.4619, 33.568, 39.3528, 35.3225, 19.1063, 26.3904, 34.2191, 35.7893,
33.0357, 31.0345, 31.4784, 29.766, 30.8925, 19.5689, 23.1998, 23.3943, 25.9599,
28.1127, 26.3468, 24.5809, 21.9321, 20.1663, 18.4004, 22.5428, 25.0512, 27.0523,
13.0395, 18.6672, 22.6522, 26.5445, 27.0092, 23.1247, 30.0724, 15.4722, 18.6594,
27.5433, 21.7538, 25.3547, 49.8238, 41.3235, 26.8149, 29.2639, 30.7344, 17.1679,
24.6506, 26.5829, 25.4842, 20.7713, 34.2889, 31.5209, 12.2958, 12.9876, 15.3268,
18.7237, 11.3998, 9.878, 45.2508, 0.0456, 0.0138, 0.341]
zero_count = 0
non_zero_sum = 0

for i in data:
    if i == 0:
        zero_count += 1
    else:
        non_zero_sum += i

non_zero_avg = non_zero_sum / (len(data) - zero_count)

print('列表中零值个数为{:d},非零值平均数为{:.5f}。'.format(zero_count, non_zero_avg))
```

注意Python中的变量在定义的同时必须赋值，不能类似C/C++语言那样预先定义一个变量而不赋值。

`len`函数可返回列表、元组、字典的长度。

方法2

```
data = [42.5066, 42.1662, 41.5372, 34.6255, 37.3276, 39.8443, 42.1836, 43.9896,
33.8646, 29.7325, 25.0662, 30.7486, 50.2659, 45.9638, 46.231, 45.1657, 46.3536,
35.5503, 39.0717, 35.6499, 36.1918, 27.7638, 30.3279, 21.3128, 22.5103, 20.8208,
28.4853, 40.9561, 33.0661, 27.646, 44.8478, 37.008, 31.3161, 33.9328, 32.8175,
0.0, 35.4657, 15.6781, 21.3214, 22.86, 34.7026, 35.1135, 38.3306, 39.1602,
33.9511, 34.5887, 36.5907, 38.8869, 40.4551, 41.9876, 47.3987, 42.8304, 41.8136,
41.4267, 41.1339, 43.0457, 43.1788, 42.9944, 43.1197, 43.6908, 44.028, 44.2258,
44.2353, 43.0444, 41.0781, 40.7676, 39.8084, 41.101, 41.3097, 42.3416, 39.4119,
30.624, 34.8342, 36.3268, 40.5735, 41.3087, 42.5451, 41.8595, 42.7566, 43.2405,
40.9939, 41.5584, 43.4783, 32.5032, 33.7413, 33.2457, 21.4866, 20.5409, 19.9374,
19.0859, 18.5459, 18.2533, 18.2445, 17.6128, 16.3594, 13.7795, 14.7944, 9.1407,
9.2812, 9.7828, 10.2063, 11.266, 12.2981, 13.8566, 13.7167, 15.6883, 18.6352,
2.2823, 2.3855, 18.805, 20.673, 23.9261, 24.6246, 30.2574, 3.3165, 18.6829,
14.3383, 0.0, 12.6839, 27.2058, 32.4048, 13.4586, 27.2213, 28.3622, 24.3946,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 5.2039, 0.0, 0.0, 25.0867, 13.6438,
18.1579, 13.744, 0.0268, 0.0, 9.7793, 20.8685, 19.2211, 2.5308, 16.9071, 9.3399,
0.0, 0.0, 0.0, 13.0796, 6.2657, 21.3445, 0.8795, 0.0, 11.6431, 0.0, 25.0044,
26.3105, 0.1709, 3.0508, 9.3071, 20.5849, 16.8375, 28.4828, 19.3334, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 18.7642, 0.0, 0.0, 5.7915, 13.4646, 20.9461, 17.2267,
16.0981, 16.2664, 7.0976, 21.1133, 13.9825, 14.6345, 18.0536, 21.7614, 24.5763,
36.0805, 49.4293, 24.6263, 29.2808, 14.3954, 42.4788, 52.5958, 43.4743, 52.0785,
51.5324, 47.749, 49.079, 44.6604, 47.4435, 45.071, 49.2676, 45.8031, 55.5259,
38.7524, 50.6233, 43.689, 37.0371, 51.842, 49.2054, 46.1528, 44.3969, 34.4268,
37.1501, 30.9934, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.4096, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.237, 18.0189, 26.9652, 0.0, 5.2103, 11.0394, 20.6039, 23.3864, 22.0404,
20.4137, 23.5754, 21.9315, 38.3556, 40.0982, 39.8685, 40.265, 40.3801, 21.7365,
41.6788, 25.4619, 33.568, 39.3528, 35.3225, 19.1063, 26.3904, 34.2191, 35.7893,
33.0357, 31.0345, 31.4784, 29.766, 30.8925, 19.5689, 23.1998, 23.3943, 25.9599,
28.1127, 26.3468, 24.5809, 21.9321, 20.1663, 18.4004, 22.5428, 25.0512, 27.0523,
13.0395, 18.6672, 22.6522, 26.5445, 27.0092, 23.1247, 30.0724, 15.4722, 18.6594,
27.5433, 21.7538, 25.3547, 49.8238, 41.3235, 26.8149, 29.2639, 30.7344, 17.1679,
24.6506, 26.5829, 25.4842, 20.7713, 34.2889, 31.5209, 12.2958, 12.9876, 15.3268,
18.7237, 11.3998, 9.878, 45.2508, 0.0456, 0.0138, 0.341]

zero_count = data.count(0)
non_zero_sum = 0

for i in data:
    if i != 0:
        non_zero_sum += i

non_zero_avg = non_zero_sum / (len(data) - zero_count)

print('列表中零值个数为{:d},非零值平均数为{:.5f}。'.format(zero_count, non_zero_avg))
```