深 圳 大 学 实 验 报 告

课程名称：现代程序设计

实验项目名称：复合数据类型

学院：电子与信息工程学院

专业：

指导教师：邹文斌

报告人：　　学号：　　班级：

实验时间：2023 年 10 月 9 日

实验报告提交时间：2023 年 10 月 14 日

教务部制

一、 实验要求

a) 列表

b) 元组

c) 字典

d) 算法

e) 需上交实验报告、py 源程序。

二、 实验环境

Python IDLE, Pycharm 等

三、 实验内容

1. 数字判断

输入一些数字，每个数字以逗号分隔，其中有一个数字出现 1 次，其余数字均会出现 2 次。请找出那个只出现一次的数字。

样例输入 1:

2,2,1

样例输出 1:

1

样例输入 2:

4,1,2,1,2

样例输出 2：

4

2. 分数

up 表示分子，down 表示分母，请计算出这个分数对应的小数是多少。

如果小数部分为循环小数，则将循环的部分括在括号内。

如果存在多个答案，只需输出任意一个。

[输入格式]输入

[输出格式]输出一行，是对应的小数。以字符串类型输出。

输入

一行，有两个数字，用空格隔开，第一个是 up，第二个是 down。

输出

输出一行，是对应的小数。以字符串类型输出。

输入样例 1

1 2

输出样例 1

0.5

输入样例 2

2 3

输出样例 2

0.(6)

输入样例 3

4 333

输出样例 3

0.(012)

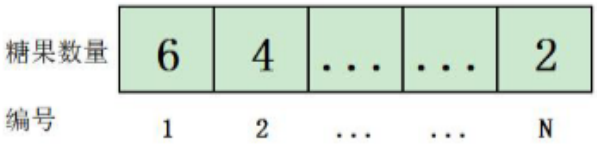
3. **分糖果**

有 N 个小朋友从左到右排成一排，每个小朋友手中都有一定数量的糖果，且糖果总数量是 N 的倍数。计算出最少调整几次可以使每个小朋友的糖果数量相同。调整规则如下:

规则 1:每个小朋友的糖果只能调整到左右相邻的两个小朋友手中；

规则 2: 第一个小朋友的糖果只能调整到第二个小朋友手中；

规则 3:最后一个小朋友的糖果，只能调整到倒数第二个小朋友手中。



例如: 1~3 号小朋友原有糖果数量分别为 6，4，2。

1)1 号小朋友拿出两块给 2 号小朋友;

2)2 号小朋友拿出两块给 3 号小朋友;

两次操作后三个小朋友手中糖果分别为 4，4，4。

即按照调整规则最少操作 2 次可以使 3 个小朋友手中糖果数量都相同。

现按照顺序给出 1~N 号小朋友手中原有糖果数量，按照调整规则计算出最少调整几次可以使小朋友手中的糖果数量都相同。

输入描述:

输入 N 个正整数(正整数 100)，表示 1 到 N 号小朋友手中原有糖果数量，正整数之间以个英文逗号隔开，且所有正整数之和是 N 的倍数。

输出描述:

按照调整规则计算出最少操作几次可以使小朋友手中糖果数量都相同。

输入样例 1

6,4,2

输出样例 1

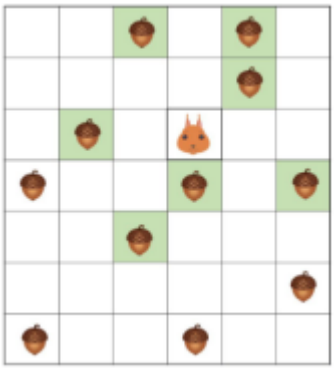
2

4. **寻找坚果**

在一个 M 行 N 列的网格中放有若干个坚果 （一个小格子里最多放一个坚果），第 X 行 Y 列的小格子是小松鼠的家。

小松鼠可以向上下左右的格子移动寻找坚果，但它每次从家出发后，最多可以移动 K 个小格子，发现格子中有坚果，就会将其运回家储藏起来(运回家所移动的格子不做计算)，然后再从家出发寻找其他坚果。小松鼠最多可以储藏几个坚果。

例如:M=7，N=6，X=3，Y=4，K=3，在 7 行 6 列的网格中有若干个坚果(如下图)，小松鼠的家在第 3 行第 4 列的位置，最多可以移动 3 个小格。



小松鼠最多可以储藏 7 个坚果（小格子底色为绿色的坚果）。

输入描述

第一行输入两个正整数 M 和 N(2<=M<=30，1<=N<=30)，表示 M 行 N 列的网格，两个正整数之间以一个英文逗号隔开;

第二行输入两个正整数 X 和 Y(1<=X<=M，1<=Y<=N)，表示小松鼠家的位置在第 X 行第 Y 列，两个正整数之间以一个英文逗号隔开;

第三行输入一个正整数 K(1<=K<=max(M，N)-1)，表示小松鼠从家出发后，最多可以移动的小格子数;

第四行开始，输入 M 行，每行 N 个整数，除了第这行列的小格子用 2 表示小松鼠的家，其他小格子的整数只能是 0 或者 1。0 表示小格子中没有坚果，1 表示小格子中有 1 个坚果，整数之间以一个英文逗号隔开。

输出描述:

输出一个整数，表示小松鼠最多可以储藏的坚果数量

样例输入:

7,6
3,4
3
0,0,1,0,1,0
0,0,0,0,1,0
0,1,0,2,0,0
1,0,0,1,0,1
0,0,1,0,0,0

0,0,0,0,0,1

1,0,0,1,0,0

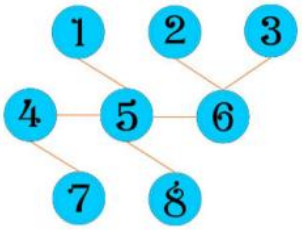
样例输出:

7

5. **蚂蚁王国（此为选做题，仅给有余力的同学挑战）**

蚂蚁王国住着 N 只蚂蚁，每只蚂蚁都有自己的领地，领地之间可以直接到达或经过其他领地间接到达，可以直接到达的领地之间的道路距离都为 1，但所有领地都有一条唯一的最短路径可以相互到达。现要在 N 块领地(依次编号为 1~N)中，选出一块领地建立游乐场，使得所有蚂蚁到游乐场的最小距离总和是 种情况中最小的。

例如: N = 8，1~8 号领地之间的连接关系为: 1 和 5、2 和 6、3 和 6、4 和 5、5 和 6、4 和 7、5 和 8。



如果将游乐场创建在 5 号领地，最小距离总和为 10。

1 号到 5 号距离为 1；2 号到 5 号距离为 2；3 号到 5 号距离为 2；4 号到 5 号距离为 1；6 号到 5 号距离为 1；7 号到 5 号距离为 2；8 号到 5 号距离为 1。

如果将游乐场创建在 6 号领地，最小距离总和为 12。

1 号到 6 号距离为 2；2 号到 6 号距离为 1；3 号到 6 号距离为 1；4 号到 6 号距离为 2；5 号到 6 号距离为 17 号到 6 号距离为 3；8 号到 6 号距离为 2。

.....

可以发现，将游乐场创建在 5 号领地，最小距离总和 10 是最小的，故输出 10。

输入描述:

第一行输入一个正整数 N(2<=N<=20)，表示领地数量

接下来输入 N-1 行，每行包含两个正整数(1<=正整数<=N，两个正整数不相同)，表示两块领地相万之间可以直接到达，正整数之间以一个英文逗号隔开(数据保证 块领地相互之间可以到达)

输出描述:

输出一个整数，表示 N 种情况中最小距离总和的最小值

样例输入:

8

1,5

2,6

3,6

4,5

5,6

4,7

5,8

样例输出:

10

四、 实验过程

思路:

（涉及到算法实现的实验需阐述算法的逻辑关系）

一、数字判断

1.首先让用户按要求输入各数字。

2.创建一个空字典用于统计。

3.将各个数字记录到字典中并初始化它们出现的次数。

4.根据字典的键值判断只出现一次的数字。

5.按照题目要求输出该只出现过一次的数字。

二、分数

1.用户按要求输入分子和分母，程序对其数据进行预处理。

2.计算分数的整数部分。

3.计算分数的余数部分并将其位置保存到一个字典中。

4.对于除法过程未结束（余数不为0），如果出现循环部分，则插入括号，按照题目要求将循环部分放在括号内。

5.同时记录当前余数的位置，将余数乘以10后将商添加到小数字符串中并更新余数。

6.按要求打印输出结果。

三、分糖果

1.用户按要求输入各个小朋友手中的糖果数量。

2.统计小朋友的数量、糖果的总数量以及糖果的平均数。

3.从第一个小朋友到倒数第二个小朋友，依次进行以下操作：

如果当前小朋友手中糖果数量大于平均数量，则将多出的糖果分给下一个小朋友。

如果当前小朋友手中糖果数量小于平均数量，需要从下一个小朋友处获得糖

果。

4.每次调整糖果数量使计数器 count+1，同时返回更新 count 的值。

5.按要求读取输入进行计算结果并输出。

四、寻找坚果

1.首先，我们定义了一个函数 find，用于计算小松鼠可以收集的最大坚果数量。该函数接受以下参数：

grid：表示整个棋盘的二维数组。

x 和 y：表示当前小松鼠所在的位置坐标。

k：表示小松鼠最多可以移动的小格子数。

home_x 和 home_y：表示小松鼠的家的位置坐标。

visited：用于标记位置是否已经访问过的二维数组。

2.在函数的开头，首先进行一些条件的判断：如果 k 为 0，或者当前位置超出了棋盘的范围，则返回 0，表示无法继续移动和收集坚果。

3.在递归部分，首先检查当前位置是否有坚果，并且该位置没有被访问过。如果满足这两个条件，我们可以收集到坚果：

首先，将当前位置的坚果数量加到 nuts 变量中，并将该位置的坚果数量设置为 0，表示已经收集过并将当前位置标记为已访问。

接着，将小松鼠的位置重置回家的位置，并递归调用 find 函数，将 k 减 1，表示小松鼠移动了一步并收集了一个坚果。

然后，向上、向下、向左、向右四个方向递归调用 find 函数，分别将 k 减 1，表示小松鼠在该方向上移动了一步。

在递归调用结束后，我们将当前位置标记为未访问，以便其他路径可以经过此位置。

最后，将小松鼠的位置重置回当前位置，并将当前位置的坚果数量恢复为原始值。

4.如果当前位置没有坚果或者已经被访问过，我们直接向上、向下、向左、向右四个方向递归调用 find 函数，将 k 减 1，表示小松鼠在该方向上移动了一步。在递归调用结束后，返回 nuts 变量，表示小松鼠可以收集到的最大坚果数量。

5.在主程序部分，首先获取输入的棋盘大小 M 和 N，以及小松鼠家的位置坐标 x 和 y，以及最多可以移动的小格子数 k。然后，创建一个空的棋盘 grid 和一个用于标记位置是否已经访问过的二维数组 visited。接下来通过循环获取每行小格子的值，并将其添加到棋盘 grid 中。最后，调用 find 函数，将小松鼠的家的位置作为参数传递给函数，并将计算得到的最大坚果数量赋值给变量 nuts。

6.输出小松鼠最多可以储藏的坚果数量。

完整代码:

（必须有详细的注释）

一、数字判断 Num_judgment.py

```
#用户输入各数字
num = input('请输入数字，以逗号分隔，其中有一个数字只出现一次，其他数字均出现两次:')
num = num.split (',')

#使用字典统计
dict ={}

#将数字记录到字典中并初始化次数
for number in num :
    dict[number]= dict.get(number,0) + 1

#判断只出现一次的数字
for number, count in dict.items():
    if count ==1:
      print(number)
      break
```

二、分数 Fraction.py

```
#获取用户输入的分子和分母，并对数据进行处理
up,down = input('请输入分子和分母，用空格隔开：').split(' ')
up = int(up)
down = int(down)

#整数部分
decimal = str(up // down) + "."

#计算余数并记录出现的位置
remainder = up % down
dict ={}

while remainder != 0:
  if remainder in dict :#出现循环部分
    #在小数字符串中插入括号将循环部分放在括号内
    decimal = decimal [:dict[remainder]] + "(" + decimal[dict[remainder]:] + ")"
    break
  #记录当前余数的位置
 dict[remainder] = len(decimal)
 remainder *=10
 #将商添加到小数字符串中
 decimal += str(remainder // down)
 #更新余数
 remainder %= down
print(decimal)
```

三、分糖果 candies.py

```
#定义一个函数用于计算调整糖果的次数
def adjust(candies):
```

```
        #统计有几个小朋友、糖果的总数量以及糖果的平均数量
        n = len(candies)
        total = sum(candies)
        avg = total // n

        #初始化次数为0
        count = 0

        #从第一个到倒数第二个小朋友
        for i in range(n - 1):
          #如果该小朋友的糖果大于平均值，则将其多出平均值的糖果分给右边的小朋友
          if candies[i] > avg:
            diff = candies[i] - avg
            candies[i] -= diff
            candies[i + 1] += diff
            count +=1

         #如果该小朋友的糖果小于平均值，则将从右边的小朋友手中获得糖果以达到平均值
          elif candies[i] < avg:
            diff = avg - candies[i]
            candies[i] += diff
            candies[i + 1] -= diff
            count += 1

      return count
```

```
#读取输入
input_str = input()
candies = list(map(int, input_str.split(',')))

#调用函数计算结果并输出
result = adjust(candies)
print(result)
```

四、寻找坚果 nuts.py

```
def find(grid, x, y, k, home_x, home_y, visited):
    if k == 0 or x < 0 or x >= len(grid) or y < 0 or y >= len(grid[0]):
        return 0

    nuts = 0

    if grid[x][y] > 0 and not visited[x][y]:
        nuts += grid[x][y]
        grid[x][y] = 0
        visited[x][y] = True
        # 在每次递归调用之前将当前位置标记为已访问

        # 将小松鼠的位置重置回家
        nuts += find(grid, home_x, home_y, k - 1, home_x, home_y, visited)

        nuts += find(grid, x - 1, y, k - 1, home_x, home_y, visited)  # 向上移动
        nuts += find(grid, x + 1, y, k - 1, home_x, home_y, visited)  # 向下移动
        nuts += find(grid, x, y - 1, k - 1, home_x, home_y, visited)  # 向左移动
```

```
        nuts += find(grid, x, y + 1, k - 1, home_x, home_y, visited)  # 向右移动

        # 在递归调用结束后将当前位置标记为未访问，使其他路径可以经过此位置
        visited[x][y] = False

        # 在每次递归调用结束后将小松鼠的位置重置回当前位置
        grid[x][y] = 1

    else:
        nuts += find(grid, x - 1, y, k - 1, home_x, home_y, visited)  # 向上移动
        nuts += find(grid, x + 1, y, k - 1, home_x, home_y, visited)  # 向下移动
        nuts += find(grid, x, y - 1, k - 1, home_x, home_y, visited)  # 向左移动
        nuts += find(grid, x, y + 1, k - 1, home_x, home_y, visited)  # 向右移动

    return nuts

mn_str = input("请输入M和N的值：")
m, n = map(int, mn_str.split(','))

xy_str = input("请输入小松鼠家的位置：")
x, y = map(int, xy_str.split(','))

k = int(input("请输入最多可以移动的小格子数："))
```

```
  grid = []
  visited = [[False] * n for j in range(m)]  # 用于标记位置是否已经访问过

  #调用函数进行计算
  for i in range(m):
    row_str = input("请输入第{}行小格子的值：".format(i+1))
    row = list(map(int, row_str.split(',')))
    grid.append(row)

  nuts = find(grid, x - 1, y - 1, k, x - 1, y - 1, visited)  # 将家的位置作为参数传递给 find 函数
  print("小松鼠最多可以储藏的坚果数量为：", nuts)
```

三、 实验结果

（运行结果，截图）

一、数字判断

```
>>>
=== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\Num_judgment.py ===
请输入数字，以逗号分隔，其中有一个数字只出现一次，其他数字均出现两次:1,2,5,2,5,1,3
3
>>>
=== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\Num_judgment.py ===
请输入数字，以逗号分隔，其中有一个数字只出现一次，其他数字均出现两次:1,5,1,3,5,2,3
2
>>>
```

二、分数

```
>>>
====== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\Fraction.py =====
请输入分子和分母，用空格隔开：1 3
0.(3)
>>>
====== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\Fraction.py =====
请输入分子和分母，用空格隔开：2 3
0.(6)
>>>
====== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\Fraction.py =====
请输入分子和分母，用空格隔开：3 7
0.(428571)
>>>
```

三、分糖果

```
>>>
====== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\candies.py
2,4,6
2
>>>
====== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\candies.py
6,4,2
2
>>>
```

四、寻找坚果

```
>>>
======== RESTART: D:\MineP\Program\Python.py\Modern_Program\Task5\nuts.py
请输入M和N的值：7,6
请输入小松鼠家的位置：3,4
请输入最多可以移动的小格子数：3
请输入第1行小格子的值：0,0,1,0,1,0
请输入第2行小格子的值：0,0,0,0,1,0
请输入第3行小格子的值：0,1,0,2,0,0
请输入第4行小格子的值：1,0,0,1,0,1
请输入第5行小格子的值：0,0,1,0,0,0
请输入第6行小格子的值：0,0,0,0,0,1
请输入第7行小格子的值：1,0,0,1,0,0
小松鼠最多可以储藏的坚果数量为： 7
```

四、 实验心得

（本次实验遇到的问题，解决过程，有什么收获等）

1.熟悉了字典的使用场景。

2.加强了对整型、浮点型数据的处理能力以及分数类问题的理解。

3.熟悉了 def 定义函数的操作以及使用。

4.搜索资料学习了数组的各种操作。

5.学习了 def 函数的各种参数传入与使用场景。

深圳大学学生实验报告用纸

成绩评定：

实验过程（60 分）	实验结果（30 分）	心得体会（10 分）	总分（100 分）

指导教师签字：　　年　月　日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。