# 2023年NOC复赛–Python初中组

## 1. 出租车费用

时间限制：1s

内存限制：128mb

（注：input()括号中不允许添加任何提示语）

根据乘坐出租车的里程，计算应该支付的费用。

出租车根据行驶里程阶梯式计价，具体规则如下表：

| | 行驶里程范围 | 单价 |
|---|---|---|
| 第一阶梯 | 10公里及以内 | 2元/公里 |
| 第二阶梯 | 11–20公里 | 1.5元/公里 |
| 第三阶梯 | 21公里及以上 | 1元/公里 |

请补全下面程序，使程序实现如下功能：

（1）程序开始运行后，输入一个正整数，表示乘坐出租车的里程数；

（2）根据规则计算并输出乘坐出租车应该支付的费用，结果保留一位小数。

```
d = int(input())
if __________:
    cost = d * 2
__________
    cost = __________
else:
    cost = __________
print('%.1f' % cost)
```

**参考代码：**

```
d = int(input())
if d <= 10:
    cost = d * 2
elif d <= 20:
    cost = 10 * 2 + (d - 10) * 1.5
else:
    cost = 10 * 2 + 10 * 1.5 + (d - 20)
print('%.1f' % cost)
```

## 2. 和的结果数

时间限制：1s
内存限制：128mb
（注：input()括号中不允许添加任何提示语）

请补全下面程序，使程序实现如下功能：
（1）输入n（2≤n≤100）个整数，整数之间以空格隔开，并保证n个数各不相同；
（2）从这n个数中任选两个，计算并输出任意两个数的和共有多少种不同结果。

例如：
输入一组整数1 3 4 6，任意两数和可能为4（1+3）、5（1+4）、7（1+6 或 3+4）、9（3+6）、10（4+6），因此，输出结果应为5。

```
ls = [int(i) for i in input().split()]
results = []
for i in range(_______):
    for j in range(_______):
        s = ls[i] + ls[j]
        if s not in results:
            ____________

print(len(results))
```

参考代码：

```
ls = [int(i) for i in input().split()]
results = []
for i in range(0, len(ls) - 1):
    for j in range(i + 1, len(ls)):
        s = ls[i] + ls[j]
        if s not in results:
            results.append(s)

print(len(results))
```

## 3. 因数和个数

时间限制：1s
内存限制：128mb
（注：input()括号中不允许添加任何提示语）

把一个数的因数全加在一起，叫做这个数的因数和。例如，6的因数有1、2、3、6，则它的因数和为1 + 2 + 3 + 6 = 12。

请编写一段程序，输入一个正整数n（n≤1000），判断因数和为n的正整数有多少个，并依次输出这些正整数。

**输入描述：**
输入一个正整数n（n≤1000）
**输出描述：**
第一行输出一个正整数，表示因数和为n的正整数的数量，如不存在，则输出0
如果存在，则第二行由小到大输出所有符合条件的数，数字之间以空格隔开

**输入样例1：**
12
**输出样例1：**
2
6 11
**样例说明1：**
因数和为12的正整数有2个，分别是6和11

**输入样例2：**
10
**输出样例2：**
0
**样例说明2：**
因数和为10的正整数不存在，所以仅输出0

**参考代码：**

```python
n = int(input())
def add(n):
    if n == 1:
        return 1
    s = 0
    for i in range(1, n):
        if i ** 2 > n:
            break
        if n % i == 0:
            s += i + n // i
            if i ** 2 == n:
                s -= i
    return s
cnt = 0
ls = []
for i in range(1, n + 1):
    if add(i) == n:
        cnt += 1
        ls.append(i)
print(cnt)
for i in ls:
    print(i, end = ' ')
```

## 4. 单词变复数

**时间限制：1s**

**内存限制：128mb**

（注：input()括号中不允许添加任何提示语）

英语单词在变成复数形式时，有以下几种常见情况（不完全）：

（1）常规情况下结尾直接加s；

（2）以s、sh、ch、x结尾的单词，加es；

（3）以辅音字母加上y结尾的单词，去掉y加ies；

（注：英文字母中，除了a、e、i、o、u这5个元音字母外，其他都是辅音字母）

请编写一段程序，输入n（1≤n≤1000）个英文单词（单数，仅包含小写字母），并按以上规则变成复数形式。

**输入描述：**

输入n（1≤n≤1000）个英文单词（单数，仅包含小写字母），单词之间以空格隔开

**输出描述：**

输出这组单词按以上规则变成的复数形式，单词之间以空格隔开

**输入样例：**

teacher box butterfly

**输出样例：**

teachers boxes butterflies

**参考代码：**

```
lsw = input().split()
def plural(word):
    if word[-1] in ['s', 'x'] or word[-2:] in ['sh', 'ch']:
        return word + 'es'
    elif word[-1] == 'y' and word[-2] not in ['a', 'e', 'i', 'o', 'u']:
        return word[:-1] + 'ies'
    else:
        return word + 's'
for w in lsw:
    print(plural(w), end=' ')
```

## 5. 啤酒兑换

**时间限制：1s**

**内存限制：128mb**

（注：input()括号中不允许添加任何提示语）

某啤酒品牌正在举办一次促销优惠活动。凭3个啤酒瓶可以再换一瓶啤酒，凭5个瓶盖也可以再换一瓶啤酒，换来的啤酒可以继续换，但不允许赊账。

请编写一段程序，计算一个人初始买入n（1≤n≤100）瓶啤酒后，最终他最多能得到多少瓶啤酒，并将这个结果输出。

**输入描述：**
输入一个正整数n，表示初始买入的啤酒瓶数（1≤n≤100）
**输出描述：**
输出一个正整数，表示最终能得到的啤酒数

**输入样例：**
10
**输出样例：**
19

**参考代码：**

```
n = int(input())
total = n
bottles = n
caps = n
while bottles >= 3 or caps >= 5:
    bottles_t = bottles // 3
    caps_t = caps // 5
    total += bottles_t + caps_t
    bottles = bottles - bottles_t * 3 + bottles_t + caps_t
    caps = caps - caps_t * 5 + bottles_t + caps_t
print(total)
```

## 6. 反转递增串

**时间限制：1s**
**内存限制：128mb**
（注：input()括号中不允许添加任何提示语）

编写一段程序，输入一个n（2≤n≤1000）位正整数，将其中所有递增数字子串进行反转。

例如，正整数12387645包含两个递增数字子串1238和45，将这两部分进行反转，得到最终的83217654。

**输入描述：**
输入一个n（2≤n≤1000）位正整数
**输出描述：**
输出一行，为将输入中所有递增数字子串进行反转的结果

**输入样例1：**
12387645
**输出样例1：**
83217654

**输入样例2：**
87654321
**输出样例2：**
87654321

**参考代码：**

```
n = input()
result = ''
i = 0
while i < len(n):
    j = i + 1
    while j < len(n) and int(n[j]) > int(n[j - 1]):
        j += 1
    result += n[i: j][::-1]
    i = j
print(result)
```

## 7. 包围黑子块

**时间限制：**1s
**内存限制：**128mb
（注：input()括号中不允许添加任何提示语）

小张发明了一种新式的棋类游戏——战斗棋。在“战斗棋”中，棋子分为黑、白两色。有一条重要的规则叫做“包围败地”。即，横竖相连的同色棋子算成一块棋，一块棋被对方棋子横竖包围起来就算一块“败地”，棋盘边角也算是包围。

现有一片战斗棋区域，包含N * M（1≤N≤100，1≤M≤100）个落子点，所有落子点均摆满了棋子，其中1代表黑子、0代表白子。

请编写一段程序，计算并输出这片区域中被包围的黑色败地数量。

**输入描述：**
第一行输入2个正整数N和M（1≤N≤100，1≤M≤100），N表示区域的行数，M表示区域的列数，正整数之间以一个空格隔开
接下来的N行每行包括M个数字（数字只能为1或0），1表示黑子，0表示白子，数字之间以一个空格隔开
**输出描述：**
输出一个正整数，表示N * M的区域中被包围的黑色败地数量

**输入样例：**
5 3
1 0 1
0 1 0
0 1 0

1 0 1
0 0 0
输出样例：
5

参考代码：

```
n, m = [int(i) for i in input().split( )]
ls = [[int(i) for i in input().split( )] for i in range(n)]
v = [[0] * m for i in range(n)]
def dfs(x, y):
    v[x][y] = 1
    for dx, dy in [(1, 0), (-1, 0), (0, 1), (0, -1)]:
        xx = x + dx
        yy = y + dy
        if 0 <= xx < n and 0 <= yy < m and ls[xx][yy] == 1 and v[xx][yy] ==
            dfs(xx, yy)
cnt = 0
for i in range(n):
    for j in range(m):
        if ls[i][j] == 1 and v[i][j] == 0:
            dfs(i, j)
            cnt += 1
print(cnt)
```