Задача 1. Маски имён файлов

Строки **python_program** и **math_theorem** заканчиваются на одну и ту же букву “m”. Поэтому рассмотрим маску, заканчивающуюся на “*m”. **nuclear_synthesis** уже не подходит под эту маску, осталось выбрать в первых двух именах такую общую букву, которой нет в имени **biological_system**. Например, такой буквой является “h”, поэтому “*h*m” — подходящая маска.

Составить маску из трёх букв нельзя, например, если рассмотреть маску вида “*z*”, где “z” — какая-то буква, то эта буква должна встречаться в первых двух именах, но не должна встречаться в двух других именах. Такой буквы нет.

Ответ: *h*m

Задача 2. Городские кварталы

В городе есть $n + 1$ дорога длины m и $m + 1$ дорога длины n.

Ответ: $(n + 1) * m + (m + 1) * n$

Правильным является любое выражение, эквивалетное данному.

Задача 3. Пятибуквенные последовательности

Такие последовательности связаны с троичной системой счисления. Если заменить букву `L` на цифру 0, букву `R` на цифру 1, а букву `V` на цифру 2, то получится такая последовательность:

```
00000
00001
00002
00010
```

То есть в строке номер i записано представление в троичной системе счисления числа $i - 1$, дополненного нулями до пяти цифр.

Число, стоящее в строке 8, можно получить из числа, стоящего в строке 4, прибавлением 4, то есть при помощи увеличения на 1 двух последних цифр. Получим последовательность `LLLVR`.

Так как $81 = 3^4$, то первые 81 последовательность начинаются с буквы `L`, а следующие 4 буквы представляют собой все последовательности длины 4. Последовательность под номером 81 будет последней из них, ответ на этот вопрос — `LVVVV`.

Для ответа на третий вопрос (что записано в строке 98) можно использовать информацию о том, что в строке 100 записано `RLVLL`. Для получения ответа нужно уменьшить число на 2. Ответом будет строка `RLRVR`.

Для ответа на четвертый вопрос строку `RLVLL` нужно увеличить на 10. Ответом будет `RRLLR`.

Наконец, для ответа на последний вопрос (что записано в строке 179) переведем число 178 в троичную систему счисления. Получится число 20121, поэтому ответом на последний вопрос будет строка `VLRVR`.

Ответ:

```
LLLVR
LVVVV
RLRVR
RRLLR
VLRVR
```

Задача 4. Станция

Самое короткое решение состоит из 7 команд:

```
2 1 3
4 3 2
5 2 1
3 1 2
```

```
1 3 2
5 2 3
2 3 2
```

Это решение оценивается в 10 баллов. Более длинные решения оцениваются меньшим числом баллов.

Задача 5. Летоисчисление

Для начала просто прибавим к году первого события A число n. Мы получим ответ, не учитывающий отсутствие нулевого года. Чтобы учесть наличие нулевого года, необходимо в некоторых случаях прибавить к ответу 1 или вычесть из ответа 1.

```
a = int(input())
n = int(input())
b = a + n
if a < 0 and b >= 0:
    b += 1
elif a > 0 and b <= 0:
    b -= 1
print(b)
```

Задача 6. Прожектора

Если ресурс первого прожектора равен a, а ресурс остальных прожекторов неограничен, то прожектора смогут гореть $4a$ секунд.

Если ресурс второго прожектора равен b, а ресурсы остальных прожекторов неограничены, то время горения будет $2b + 1$.

Наконец, если ограничено только время горения третьего прожектора c, то ответ будет $4c + 2$.

Необходимо вывести наименьшее из этих величин.

```
a = int(input())
b = int(input())
c = int(input())
print(min(4 * a, 2 * b + 1, 4 * c + 2))
```

Задача 7. Ремонт забора

В задаче нужно последовательность из N единиц и нулей "накрыть" отрезками длины не более L так, чтобы были "накрыты" все единицы.

Это можно сделать простым жадным алгоритмом: найдём первую единицу и будем считать, что очередной отрезок длины L начинается с этой единицы, то есть можно пропустить следующие $L - 1$ элемент массива: если единица была встречена на позиции i, то можно продолжить просмотр последовательности начиная со значения $i + L$. В эффективном решении (на полный балл) задача решается за один проход по массиву, то есть с использованием одного цикла. Если в решении есть вложенные циклы, то оно может не уложиться по времени в ограничение 1 секунда и наберёт неполный балл.

Вот пример правильного и эффективного решения:

```
L = int(input())
N = int(input())
A = []
for i in range(N):
    A.append(int(input()))
ans = 0
i = 0
```

```
while i < len(A):
    if A[i] == 1:
        ans += 1
        i += L
    else:
        i += 1
print(ans)
```

Задачу можно решать и без использования массива, то есть необязательно сохранять в памяти все считанные числа. Достаточно запоминать номер элемента последовательности, являющийся началом последнего отрезка длины L (`last_start`). Тогда следующий отрезок начинается, если была считана единица и номер текущего считанного числа больше или равен, чем `last_start`$+L$. Пример такого решения:

```
L = int(input())
N = int(input())
ans = 0
last_start = -10 ** 6
for i in range(N):
    fence_elem = int(input())
    if fence_elem == 1 and i >= last_start + L:
        ans += 1
        last_start = i
print(ans)
```