

Задача А. Часы (15 баллов)

Представим себе окружность, на которой отмечено $24 \cdot 60 = 1440$ точек, и каждая точка соответствует некоторой минуте в сутках. Пронумеруем точки по часовой стрелке от 0 до 1439, где точка 0 — это время 00:00, а точка 1439 — это время 23:59. Пусть время на часах соответствует точке x , а правильное время — точке y .

Заметим, что кнопка А увеличивает номер текущей точки на 1, а кнопка В — на 60.

Найдём d — расстояние между точками по часовой стрелке. Если $x \leq y$, то $d = y - x$, иначе $d = (y + 1440) - x$.

Мы хотим как можно быстрее продвинуться на расстояние d , используя прыжки вперёд на расстояния 1 и 60. Легко понять, что нам нужно использовать $d \bmod 60$ прыжков первого типа и $d \div 60$ прыжков второго типа. Значит, ответ на задачу равен $d \bmod 60 + d \div 60$.

Можно было и не использовать при решении задачи конструкцию с окружностью, описанную выше, а просто промоделировать процесс, если догадаться, что сначала нужно увеличивать минуты, пока они не станут правильными, а потом — увеличивать часы.

Ниже представлено решение на Python 3.

```
def t():
    s = input()
    return int(s[0:2]) * 60 + int(s[3:5])

t1, t2 = t(), t()
d = t2 - t1
if d < 0:
    d += 1440
print(d % 60 + d // 60)
```

Задача В. Калькулятор (18 баллов)

В этой задаче требовалось сделать ровно то, что написано в условии. Сначала вывести первую строчку всех цифр, потом вторую и так далее. Что бы это было проще сделать, можно заметить, что есть всего 4 типа различных строчек, которые бывают во всех цифрах. А именно строчки «*...», «...*», «*...*» и «*****». Поэтому для каждой из цифр можно было сделать массив из 7 чисел, определяющие, какая из этих строчек первая строчка цифры, вторая и так далее. И тогда для каждой из 7 строчек вывода можно просто выводить одну из возможных строчек для каждой цифры числа.

Ниже представлено решение на Python 3.

```
arr = ['...*', '*...', '*...*', '*****'];
mas = [[4, 3, 3, 3, 3, 3, 4], [1, 1, 1, 1, 1, 1, 1], [4, 1, 1, 4, 2, 2, 4], [4, 1, 1, 4, 2, 2, 4]];

v = input()
n = len(v)
for i in range(7):
    for j in range(n):
        if (j > 0):
            print('.', end = '')
        print(arr[mas[int(v[j])][i]], end = '')
    print('')
```

Задача С. Формула 2 (20 баллов)

Переформулируем задачу, требуется найти все такие k , что среди первых k элементов нет таких, которые больше чем k , то есть встречаются все числа от 1 до k . Чтобы решить эту задачу будем проходить по массиву и поддерживать максимальное число на префиксе. Если максимальное число на префиксе равно длине префикса, то это означает что все числа на префиксе не больше k , следовательно это ответ.

Ниже представлено решение на Python 3.

```
n = int(input())
a = list(map(int, input().split()))
ans = []
ma = 0
for i in range(n):
    ma = max(ma, a[i])
    if i + 1 == ma:
        ans.append(i + 1)
print(len(ans))
print(*ans)
```

Задача D. Подпалиндромы (22 балла)

Удалим все пробелы, так как они не влияют на количество палиндромов.

Переберем центр палиндрома. Тогда первая буква будет совпадать с последней. Переберем какой она может быть от **a** до **z**. Чтобы узнать сколько у нас палиндромов с таким центром и зафиксированной буквой, нам нужно знать сколько раз встречается буква левее центра палиндрома и сколько раз правее и перемножить эти количества. Для того чтобы узнать сколько раз встречается левее и сколько раз правее, воспользуемся техникой частичных сумм.

Ниже представлено решение на Python 3.

```
import string

s = input().replace("_", "")
n = len(s)
cnt = [[0]*26 for _ in range(n + 1)]

for i in range(n):
    cnt[i + 1] = cnt[i].copy()
    cnt[i + 1][ord(s[i]) - 97] += 1
ans = 0
for i in range(1, n - 1):
    for c in range(0, 26):
        ans += cnt[i][c] * (cnt[n][c] - cnt[i + 1][c]);

print(ans)
```