Разбор задач

Задача 1. Три кубика

Сумма чисел на одном кубике равна $1+2+3+4+5+6 = 21$. У среднего кубика (как его не располагай) не будут видны две противоположные стороны, сумма на них равна 7. У двух крайних кубиков для максимизации результата скроем грани с числом 1. Итого $21 + 21 + 21 - 7 - 1 - 1 = 63 - 9 = 54$.

Задача 2. Садовые гномы

Аккуратным проходом по всем 25 клеткам получим, что сад будет выглядеть так:

```
00000
00101
10110
01111
00000
```

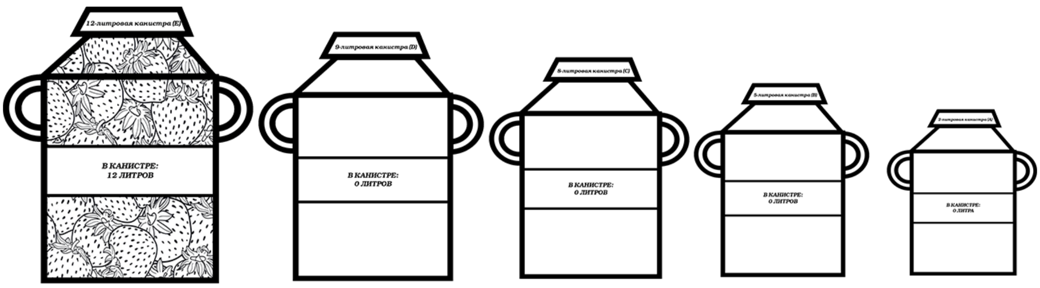
Задача 3. Земляничная поляна

Пример решения из пяти команд:

E>D
D>C
C>B
B>A
A>D

Можно показать, что более быстрого алгоритма не существует.

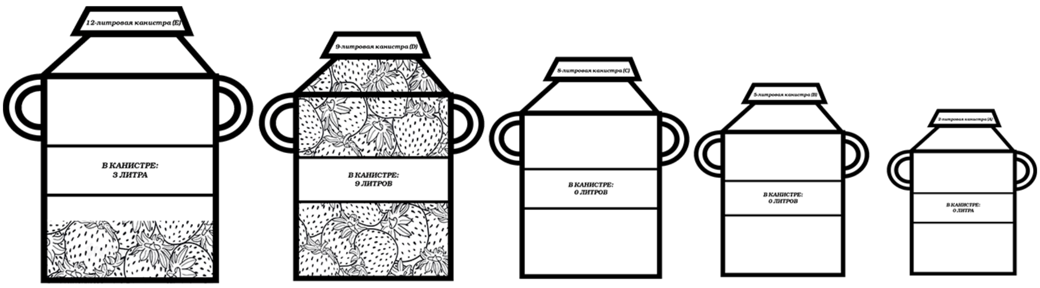
Решение:



Шаг 1.

Из 12-литровой канистры (E) необходимо пересыпать ягоду в 9-литровую канистру (D). После пересыпания в 12-литровой канистре (E) останется 3 литра земляники, исходя из условия, эта канистра достается одному из туристов и больше использоваться не может.

Решение:



Шаг 2.

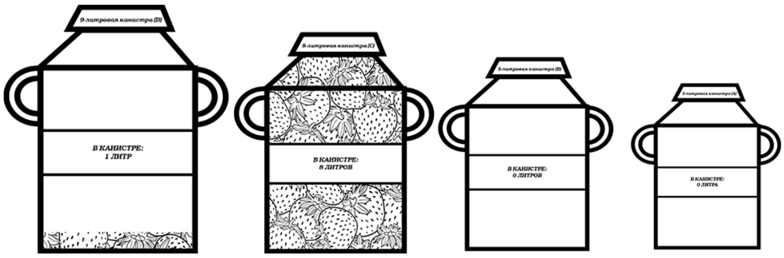
Из 9-литровой канистры (D) необходимо пересыпать в 8-литровую канистру (C).

Получается:

8 литров в 8-литровой канистре (C)

1 литр в 9-литровой канистре (D)

Решение:



Шаг 3.

Из 8-литровой канистры (C) необходимо пересыпать в 5-литровую канистру(B).

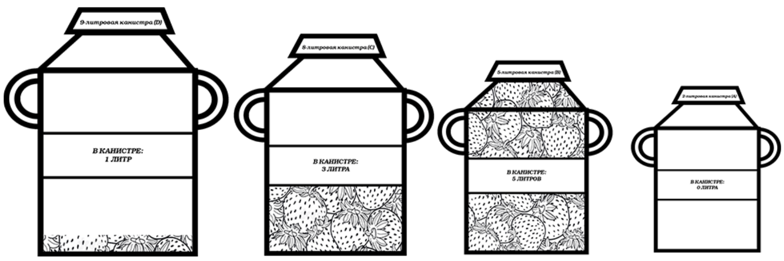
Получается:

1 литр в 9-литровой канистре (D)

5 литров в 5-литровой канистре (B)

3 литра в 8-литровой канистре (C). Значит 8-литровую канистру (C) забирает следующий турист и она больше не может участвовать в пересыпании.

Решение:



Шаг 4.

Из 5-литровой канистры (B) необходимо пересыпать в 2-литровую канистру (A).

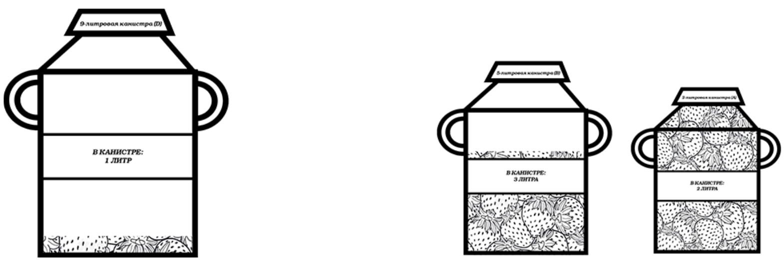
Получается:

1 литр в 9-литровой канистре (D)

2 литра в 2-литровой канистре (A)

3 литра в 5-литровой канистре (B). Значит 5-литровую канистру (B) забирает следующий турист и она больше не может участвовать в пересыпании.

Решение:



Шаг 5.

Из 2-литровой канистры (B) необходимо пересыпать в 9-литровую канистру (D), тем самым получаем последние 3 литра земляники для последнего туриста.

Решение:

Задача 4. Случай в лагере

Ответ:

MK
DT
AP
NG
LF

Решение: 1. Необходимо построить таблицу следующего вида:

		Номер тумбочки				
		1	2	3	4	5
Имена	Марина(M)					
	Даша (D)					
	Алёна (A)					
	Лера (L)					
	Наташа (N)					
Увлечение	Спортивная гимнастика (G)					
	Танго(T)					
	футбол(F)					
	Вязание(K)					
	Рисование(P)					

		Увлечение				
		Спортивная гимнастика (G)	Танго(T)	футбол(F)	Вязание(K)	Рисование(P)
Имена	Марина(M)					
	Даша (D)					
	Алёна (A)					
	Лера (L)					
	Наташа (N)					

Расставьте, согласно высказываниям, (+) (плюсы) на пересечении имени и увлечения, либо имени и номера тумбочки, либо увлечения и номера тумбочки, если они соответствуют друг другу, и (-) (минусы) в противном случае.

2. В первом высказывании сказано: «Тумбочка девочки, увлекающейся рисованием, находится посередине». Согласно этому можно сделать вывод: тумбочку под номером 3 заняла девочка, чьё увлечение - рисование. На пересечении номера тумбочки (в нашем случае - 3) и увлечения (рисование) необходимо поставить (+)(плюс). На пересечении остальных увлечений и третьей тумбочки поставьте (-) (минус). Аналогично, на пересечении 1,2,4 и 5 тумбочек и рисования, ставим (-)(минус).

		Номер тумбочки				
		1	2	3	4	5
Имена	Марина(M)					
	Даша (D)					
	Алёна (A)					
	Лера (L)					
	Наташа (N)					
Увлечение	Спортивная гимнастика (G)			-		
	Танго(T)			-		
	футбол(F)			-		
	Вязание(K)			-		
	Рисование(P)	-	-	+	-	-

		Увлечение				
		Спортивная гимнастика (G)	Танго(T)	футбол(F)	Вязание(K)	Рисование(P)
Имена	Марина(M)					
	Даша (D)					
	Алёна (A)					
	Лера (L)					
	Наташа (N)					

3. Исходя из второго высказывания у Марины тумбочка под номером 1. Её увлечение не связано ни со спортом, ни с танцами, значит она увлекается либо рисование, либо вязанием. Из первого высказывания известно, что девочка, которая увлекается рисование, заняла тумбочку под номером 3, следовательно это не Марина. Значит её увлечение - вязание. Аналогично первому примеру расставляем (+)(плюсы) и (-)(минусы).

		Номер тумбочки				
		1	2	3	4	5
Имена	Марина(M)	+	-	-	-	-
	Даша (D)	-				
	Алёна (A)	-				
	Лера (L)	-				
	Наташа (N)	-				
Увлечение	Спортивная гимнастика (G)	-		-		
	Танго(T)	-		-		
	футбол(F)	-		-		
	Вязание(K)	+		-		
	Рисование(P)	-	-	+	-	-

	Увлечение				
	Спортивная гимнастика (G)	Танго(T)	футбол(F)	Вязание(K)	Рисование(P)
Марина(M)	-	-	-	+	-
Даша (D)					
Алёна (A)					
Лера (L)					
Наташа (N)					

4. Из четвертого высказывание стало известно, что в тумбочке под номером 5 находится футбольный мяч, следовательно, эту тумбочку заняла девочка, которая увлекается футболом. Ставим на пересечении 5 тумбочки и футбола (+)(плюс), на пересечении остальных увлечений и 5 тумбочки (-) (минус). Так же (-) (минус) ставим на пересечении футбола и тумбочек под номерами 1,2,3,4.

		Номер тумбочки				
		1	2	3	4	5
Имена	Марина(M)	+	-	-	-	-
	Даша (D)	-				
	Алёна (A)	-				
	Лера (L)	-				
	Наташа (N)	-				
Увлечение	Спортивная гимнастика (G)	-		-		
	Танго(T)	-		-		
	футбол(F)	-		-		
	Вязание(K)	+		-		
	Рисование(P)	-	-	+	-	-

	Увлечение				
	Спортивная гимнастика (G)	Танго(T)	футбол(F)	Вязание(K)	Рисование(P)
Марина(M)	-	-	-	+	-
Даша (D)					
Алёна (A)					
Лера (L)	-				
Наташа (N)					

5. У Алёны нечетный номер тумбочки. К нечетным номерам относятся 1, 3 и 5 тумбочки. Уже известно из предыдущих высказываний, что первую тумбочку заняла Марина. В 5 тумбочка досталась футболистке, но Алёна, исходя из высказывания, не любит спорт, значит она не могла занять 5 тумбочку. Остается только тумбочка под номером 3, которую займет Алёна. Из первого высказывания следует, что девочка, занявшая 3 тумбочку, увлекается рисованием, так как мы выясняли, что Алёна заняла 3 тумбочку, значит она увлекается рисованием. Расставьте (+) и (-).

		Номер тумбочки				
		1	2	3	4	5
Имена	Марина(M)	+	-	-	-	-
	Даша (D)	-		-		
	Алёна (A)	-	-	+	-	-
	Лера (L)	-		-		
	Наташа (N)	-		-		
Увлечение	Спортивная гимнастика (G)	-		-		-
	Танго(T)	-		-		-
	футбол(F)	-	-	-	-	+
	Вязание(K)	+		-		-
	Рисование(P)	-	-	+	-	-

	Увлечение				
	Спортивная гимнастика (G)	Танго(T)	футбол(F)	Вязание(K)	Рисование(P)
Марина(M)	-	-	-	+	-
Даша (D)				-	-
Алёна (A)	-	-	-	-	+
Лера (L)	-			-	-
Наташа (N)				-	-

6. Из 6 высказывания следует, что Даша увлекается танго, так как она танцор. Тумбочка её будет находится не с краю, значит Дашиной тумбочкой может быть 2,3,4. Из предыдущего шага выясняли, что 3 тумбочка досталась Алёне. Значит Даша может занять или 2 или 4 тумбочку. Расставьте (+) и (-) в соответствующие ячейки.

		Номер тумбочки				
		1	2	3	4	5
Имена	Марина(M)	+	-	-	-	-
	Даша (D)	-		-		-
	Алёна (A)	-	-	+	-	-
	Лера (L)	-		-		
	Наташа (N)	-		-		
Увлечение	Спортивная гимнастика (G)	-		-		-
	Танго(T)	-		-		-
	футбол(F)	-	-	-	-	+
	Вязание(K)	+		-		-
	Рисование(P)	-	-	+	-	-

	Увлечение				
	Спортивная гимнастика (G)	Танго(T)	футбол(F)	Вязание(K)	Рисование(P)
Марина(M)	-	-	-	+	-
Даша (D)	-	+	-	-	-
Алёна (A)	-	-	-	-	+
Лера (L)	-	-		-	-
Наташа (N)		-		-	-

7. Из таблицы видно, что спортивной гимнастикой увлекается Наташа, т.к. напротив остальных девочек находятся (-). Это значит, что футбол остается Лере. Из 3 высказывания нам известно, что тумбочки Леры, которая увлекается футболом, и гимнастки, гимнастикой увлекается Наташа, находятся рядом. Тумбочка под номером 5 принадлежит футболистке (Лере), значит тумбочку номером 4 заняла гимнастка Наташа. Расставьте (+) и (-) в соответствующие ячейки

		Номер тумбочки				
		1	2	3	4	5
Имена	Марина(M)	+	-	-	-	-
	Даша (D)	-		-	-	-
	Алёна (A)	-	-	+	-	-
	Лера (L)	-	-	-	-	+
	Наташа (N)	-	-	-	+	-
Увлечение	Спортивная гимнастика (G)	-	-	-	+	-
	Танго(T)	-		-	-	-
	футбол(F)	-	-	-	-	+
	Вязание(K)	+	-	-	-	-
	Рисование(P)	-	-	+	-	-

	Увлечение				
	Спортивная гимнастика (G)	Танго(T)	футбол(F)	Вязание(K)	Рисование(P)
Марина(M)	-	-	-	+	-
Даша (D)	-	+	-	-	-
Алёна (A)	-	-	-	-	+
Лера (L)	-	-	+	-	-
Наташа (N)	+	-	-	-	-

8. В таблице осталась только одна незаполненная ячейка на пересечении Даши и тумбочки под номером 2, значит Даша занимает эту тумбочку.

Задача 5. Сегодняшнее число

Правильный ответ:
109012
1212020
222911
2
09122021

Комментарии.

1. В нашем наборе две цифры 0, две цифры 1, три цифры 2 и одна цифра 9. Для формирования палиндрома необходимо поставить цифру в начало палиндрома и такую же цифру в конец, то есть, нам необходимо две одинаковые цифры. Чтобы палиндром стал наибольшим из всех возможных, его первая цифра должна быть наибольшей из всех возможных. Из всех цифр, которые присутствуют как минимум два раза в наборе (это 0, 1 и 2), наибольшая цифра — 2. Поставим её в начало и конец палиндрома: 2*2. Набор цифр уменьшился, теперь в нем две цифры 0, две цифры 1, одна цифра 2 и одна цифра 9.

   Попробуем продолжить формирование палиндрома: снова найдем наибольшую цифру, встречающуюся как минимум два раза в наборе (это 0 и 1) — цифру 1 и поставим её на второе с начала и с конца. Теперь наш палиндром выглядит так: 21*12, а набор так — две цифры 0, одна цифра 2 и одна цифра 9.

   Попробуем продолжить формирование палиндрома: снова найдем наибольшую цифру, встречающуюся как минимум два раза в наборе — цифру 0 и поставим её на третье с начала и с конца. Теперь наш палиндром выглядит так: 210*012, а набор так — одна цифра 2 и одна цифра 9.

   У нас не осталось ни одной цифры в наборе, встречающейся как минимум дважды. Но в палиндром можно вставить еще одну цифру на центральную позицию. Из двух имеющихся у нас цифр выберем наибольшую — цифру 9. Окончательно наш палиндром выглядит так: 2109012, а в наборе осталась неиспользованной одна цифра 2.

2. На первую позицию поставим наибольшую цифру 9. На вторую позицию нужно поставить такую достаточно большую цифру, чтобы следующая цифра была ещё больше неё. На вторую позицию поставим цифру 1, на третью — 2. Получили число 912*. На четвертую и пятую позиции опять можно поставить такой же набор цифр. Получили число 91212*. Из оставшихся цифр (2, 0, 0) составим единственно возможное окончание числа: 91212020.

3. 22:29:11. Наибольшее число часов, которое может быть на индикаторе, равно 23. Из цифр нашего набора мы можем составить наибольшее подходящее число 22. В наборе останутся цифры 001129. Наибольшее число минут, которое может быть на индикаторе, равно 59. Из цифр нашего набора мы можем составить наибольшее подходящее число 29. В наборе останутся цифры 0011. Наибольшее число секунд, которое может быть на индикаторе, равно 59. Из цифр нашего набора мы можем составить наибольшее подходящее число 11.

4. Сформируем начала 4-х двузначных чисел, так, чтобы они отличались как можно меньше: 1*, 2*, 2 и 2*. Остались цифры (0, 0, 1 и 9). Подставим 9 в конец меньшего из начал: 19. Остальные 3 числа — 20, 20 и 21. Разность между 21 и 19 равна 2.

5. Текущая дата 29.10.2021. Попробуем «остаться» в текущем году и найти корректную пару «число-месяц» из цифр 0, 1, 2 и 9. Понятно, что в октябре такой даты уже не будет, для ноября не хватает еще одной единицы, а вот декабрь сформировать можно из цифр 1 и 2. Тогда на число месяца приходятся две цифры 0 и 9 из которых можно получить корректную дату 09. Итого 09 декабря 2021 года или 09122021.

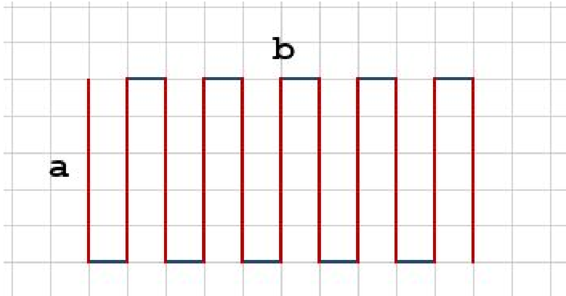
Разбор задач

Задача 1. Найди все палиндромы

Всего имеется 9 различных подстрок-палиндромов:
ABA
ABABA
ABBCBBA
BAB
BABAB
BABABAB
BB
BBCBB
BCB

Задача 2. Ночная смена

Полный путь охранника состоит из $(b + 1)$ вертикальных линий длины a и b горизонтальных линий длины 1.



Всего получается

$$(b + 1) * a + b * 1$$

что можно записать разными способами, например, если раскрыть скобки, то получится

$$a * b + a + b$$

Задача 3. Запертая комната

Ответ:

311
13
3312
133111
33312

Решение.

1. Найдем все числа, которые можно получить путем нажатия одной из кнопок (поскольку на индикаторе нечетное число, имеет смысл нажимать только первую и третью кнопку). Это числа 2 (1) и 5 (3). В скобках будем указывать последовательность нажатий кнопок для получения этого числа.

2. Найдем все числа, которые можно получить путем нажатия двух кнопок. Очевидно, что все такие числа возможно получить только из чисел, которые можно получить путем нажатия на одну кнопку (мы их нашли на предыдущем шаге) добавив в конец их кода 1, 2 или 3.

Из числа 2 можно получить числа 3 (11), 10 (13) и 1 (12). Последнее число уберем из списка — его мы можем получить вообще не нажимая ни одной кнопки (в общем случае будем добавлять в наш список только новые числа, поскольку если число уже встретилось нам ранее, очевидно, что оно имело более короткий код).

Из числа 5 получим числа 6 (31) и 25 (33). Выпишем все полученные на втором шаге числа в порядке возрастания (для облегчения дальнейшей работы): 3 (11), 6 (31), 10 (13) и 25 (33). Отметим, что число 10 присутствует в задании, ответ для этого числа — его код (13).

3. На следующих шагах будем поступать аналогично: из всех чисел, полученных на предыдущих шагах постараемся получить все возможные новые числа, сохраняя их коды. Вот числа, которые будут получены на третьем шаге (в порядке возрастания): 4 (111), 7 (311), 11 (131), 15 (113), 26 (331), 30 (313), 50 (133) и 125 (333). Отметим, что число 7 присутствует в задании, ответ для этого числа — его код (311).

4. Вот числа, которые будут получены на четвертом шаге (в порядке возрастания): 8 (3111), 12 (1311), 13 (3312), 16 (1131), 20 (1113), 27 (3311), 31 (3131), 35 (3113), 51(1331), 55(1313), 75(1133), 126(3331), 130(3313), 150(3133), 250(1333) и 625(3333). Отметим, что число 13 присутствует в задании, ответ для этого числа — его код (3312).

5. Этот алгоритм можно продолжать и дальше (в информатике подобный способ нахождения кратчайшего пути между двумя объектами носит название «волновой алгоритм» или «алгоритм Ли»), но чисел стало много, работать становится сложнее. К счастью, нам осталось найти коды всего для двух чисел. Попробуем зайти с обратной стороны.

   Число 63 из задания возможно получить только из чисел 62 (с помощью первой кнопки) и 126 (с помощью второй кнопки). Число 126 нам встретилось на 4 шаге, значит ответ для числа 63 равен коду для числа 126 с добавленной цифрой 2, то есть (33312).

   Число 53 из задания возможно получить только из чисел 52 (с помощью первой кнопки) и 106 (с помощью второй кнопки). Ни одно из этих чисел нам пока не встретилось, но можно заметить, что число 52 возможно получить из числа 51 (встретившимся нам на четвертом шаге). Тогда ответ для последнего числа из задания 53 будет (133111).

Также данную задачу возможно решить путем составления компьютерной программы и реализации алгоритма динамического программирования.

Задача 4. Земляничная поляна

Пример из 11 команд:

E>A
E>C
E>D
C>B
A>C
D>F
F>A
A>D
B>D
D>A
A>B

Пример решения задачи представлен в таблице:

шаг	20	3	5	6	7	30
1.	17	3	0	0	0	0
2.	11	3	0	6	0	0
3.	4	3	0	6	7	0
4.	выбыл	3	5	1	7	0
5.	выбыл	0	5	4	7	0
6.	выбыл	0	5	выбыл	0	7
7.	выбыл	3	5	выбыл	0	4
8.	выбыл	0	5	выбыл	3	выбыл
9.	выбыл	0	1	выбыл	7	выбыл
10.	выбыл	3	1	выбыл	4	выбыл
11.	выбыл	0	4	выбыл	выбыл	выбыл

Задача 5. Костяные войны

Частичное решение:

Переберем все возможные a и b, не превышающие P, и проверим выполнение равенства $2(a + b) = P$.

```
p = int(input())
count = 0
for a in range(1, p + 1):
    for b in range(1, p + 1):
        if 2 * (a + b) == p:
            count += 1
print(count)
```

Полное решение:

Очевидно, что если периметр нечётен, то способов нет ни одного, и нужно вывести ноль.

В чётном случае заметим, что один из них, пускай Марш, может пожертвовать отрезки длиной от 1 до $P/2 - 1$, а отрезки Копа определятся однозначно. Всего $P/2 - 1$ способ.

```
n = int(input())
if n % 2 == 1:
    print(0)
else:
    print(n // 2 - 1)
```

Задача 6. Потерявшееся число

Частичное решение:

Будем перебирать все числа, делящиеся на 6, получать их половину и треть и проверять, равна ли какая-нибудь пара чисел из этой тройки паре чисел, данной в условии. Если да — ответом будет третье число из тройки, если нет — переходим к следующему числу, делящемуся на 6, и проверяем его.

Полное решение:

Первый случай: остались карточки, на которых изначально были записаны исходное число и его половина, потерялась карточка с третью числа.

Признаком этого случая является следующее равенство: 2 * a = b. Значит, третье число можно вычислить по формуле b // 3.

Второй случай: остались карточки, на которых изначально были записаны исходное число и его треть, потерялась карточка с половиной числа.

Признаком этого случая является следующее равенство: 3 * a = b. Значит, третье число можно вычислить по формуле b // 2.

Третий случай: остались карточки, на которых изначально были записаны половина исходного число и его треть, потерялась карточка с самим числом.

Признаком этого случая является следующее равенство: 3 * a = 2 * b. Значит, третье число можно вычислить по формуле 3 * a.

```
a = int(input())
b = int(input())
if a * 2 == b:
    print(b // 3)
elif a * 3 == b:
    print(b // 2)
else:
    print(2 * b)
```

Задача 7. Два грузчика

Частные решения:

Первая подзадача (любую коробку может унести любой грузчик):

В этом случае каждой ходкой переносится по 2 коробки, поэтому...

Первый случай: если n — четное: $ans = n/2$.

Второй случай: если n — нечетное: $ans = n/2 + 1$ (последнюю коробку несет кто-то один).

В обеих случаях ответом будет $n/2$, округленное вверх до целого числа или $ans = \lfloor(n+1)/2\rfloor$, где операция $\lfloor.\rfloor$ — целая часть от результата деления.

```
b = int(input())
n = int(input())
ans = (n + 1) // 2
print(ans)
```

Вторая подзадача (нет коробок, которые нужно тащить вдвоем):

Пусть x — количество коробок, которые может перенести Шурик, y — количество коробок, которые может перенести Федя, но не может Шурик (считывая вес очередной коробки будем увеличивать на 1 соответствующую переменную).

Первый случай: если $x \geqslant y$. В этом случае Федя перетаскает все свои коробки не раньше Шурика и общее количество ходок получится таким же, как и в первой подзадаче: ответом будет $n/2$ округленное вверх до целого числа или $ans = \lfloor(n+1)/2\rfloor$.

Второй случай: если $y > x$. В этом случае Федя перетаскает все свои коробки позже Шурика и несколько последних ходок Шурик будет просто идти рядом с Федей с пустыми руками: $ans = y$.

```
b = int(input())
n = int(input())
L = list()
for i in range(n):
    L.append(int(input()))

x, y = 0, 0
for i in L:
    if i <= a:
```

```
        x += 1
    else:
        y += 1

ans = 0
if x <= y:
    ans += y
else:
    ans += (x + y + 1) // 2
print(ans)
```

Полное решение:

Пусть x — количество коробок, которые может перенести Шурик, y — количество коробок, которые может перенести Федя, но не может Шурик, z — количество коробок, которые они могут перенести только вдвоем.

Тогда ответ точно равен z (столько коробок они несут вдвоём) плюс...

Первый случай: если $x \leqslant y$, то к ответу добавим y (переходов Феди с грузом больше, чем у Шурика).

Второй случай: иначе добавим $(x+y)/2$ с округлением вверх (Шурик не ходят без груза, кроме, возможно, последнего перехода, если осталась одна коробка на двоих).

```
a = int(input())
b = int(input())
n = int(input())
x, y, z = 0, 0, 0
for i in range(n):
    p = int(input())
    if p <= a:
        x += 1
    elif p <= b:
        y += 1
    else:
        z += 1
ans = z
if x <= y:
    ans += y
else:
    ans += (x + y + 1) // 2
print(ans)
```

Разбор задач

Задача 1. Костяные войны

Частичное решение:

Переберем все возможные a и b, не превышающие P, и проверим выполнение равенства $2(a+b) = P$.

```
p = int(input())
count = 0
for a in range(1, p + 1):
    for b in range(1, p + 1):
        if 2 * (a + b) == p:
            count += 1
print(count)
```

Полное решение:

Очевидно, что если периметр нечётен, то способов нет ни одного, и нужно вывести ноль.

В чётном случае заметим, что один из них, пускай Марш, может пожертвовать отрезки длиной от 1 до $P/2 - 1$, а отрезки Копа определятся однозначно. Всего $P/2 - 1$ способ.

```
n = int(input())
if n % 2 == 1:
    print(0)
else:
    print(n // 2 - 1)
```

Задача 2. Потерявшееся число

Частичное решение:

Будем перебирать все числа, делящиеся на 6, получать их половину и треть и проверять, равна ли какая-нибудь пара чисел из этой тройки паре чисел, данной в условии. Если да — ответом будет третье число из тройки, если нет — переходим к следующему числу, делящемуся на 6, и проверяем его.

Полное решение:

Первый случай: остались карточки, на которых изначально были записаны исходное число и его половина, потерялась карточка с третью числа.

Признаком этого случая является следующее равенство: 2 * a = b. Значит, третье число можно вычислить по формуле b // 3.

Второй случай: остались карточки, на которых изначально были записаны исходное число и его треть, потерялась карточка с половиной числа.

Признаком этого случая является следующее равенство: 3 * a = b. Значит, третье число можно вычислить по формуле b // 2.

Третий случай: остались карточки, на которых изначально были записаны половина исходного число и его треть, потерялась карточка с самим числом.

Признаком этого случая является следующее равенство: 3 * a = 2 * b. Значит, третье число можно вычислить по формуле 3 * a.

```
a = int(input())
b = int(input())
if a * 2 == b:
    print(b // 3)
elif a * 3 == b:
```

```
    print(b // 2)
else:
    print(2 * b)
```

Задача 3. Два грузчика

Частные решения:

Первая подзадача (любую коробку может унести любой грузчик):

В этом случае каждой ходкой переносится по 2 коробки, поэтому...

Первый случай: если n — четное: $ans = n/2$.

Второй случай: если n — нечетное: $ans = n/2 + 1$ (последнюю коробку несет кто-то один).

В обеих случаях ответом будет $n/2$, округленное вверх до целого числа или $ans = \lfloor (n+1)/2 \rfloor$, где операция $\lfloor . \rfloor$ — целая часть от результата деления.

```
b = int(input())
n = int(input())
ans = (n + 1) // 2
print(ans)
```

Вторая подзадача (нет коробок, которые нужно тащить вдвоем):

Пусть x — количество коробок, которые может перенести Шурик, y — количество коробок, которые может перенести Федя, но не может Шурик (считывая вес очередной коробки будем увеличивать на 1 соответствующую переменную).

Первый случай: если $x \geqslant y$. В этом случае Федя перетаскает все свои коробки не раньше Шурика и общее количество ходок получится таким же, как и в первой подзадаче: ответом будет $n/2$ округленное вверх до целого числа или $ans = \lfloor (n+1)/2 \rfloor$.

Второй случай: если $y > x$. В этом случае Федя перетаскает все свои коробки позже Шурика и несколько последних ходок Шурик будет просто идти рядом с Федей с пустыми руками: $ans = y$.

```
b = int(input())
n = int(input())
L = list()
for i in range(n):
    L.append(int(input()))

x, y = 0, 0
for i in L:
    if i <= a:
        x += 1
    else:
        y += 1

ans = 0
if x <= y:
    ans += y
else:
    ans += (x + y + 1) // 2
print(ans)
```

Полное решение:

Пусть x — количество коробок, которые может перенести Шурик, y — количество коробок, которые может перенести Федя, но не может Шурик, z — количество коробок, которые они могут перенести только вдвоем.

Тогда ответ точно равен z (столько коробок они несут вдвоём) плюс...

Первый случай: если $x \leqslant y$, то к ответу добавим y (переходов Феди с грузом больше, чем у Шурика).

Второй случай: иначе добавим $(x+y)/2$ с округлением вверх (Шурик не ходят без груза, кроме, возможно, последнего перехода, если осталась одна коробка на двоих).

```
a = int(input())
b = int(input())
n = int(input())
x, y, z = 0, 0, 0
for i in range(n):
    p = int(input())
    if p <= a:
        x += 1
    elif p <= b:
        y += 1
    else:
        z += 1
ans = z
if x <= y:
    ans += y
else:
    ans += (x + y + 1) // 2
print(ans)
```

Задача 4. Любовь и двери

Далее под «правильным набором дверей» будем понимать такой набор, при котором изменять их уже не нужно, то есть в начале набора стоят двери одного типа, а в конце двери другого типа, например набор 2, 2, 2, 2, 1, 1, 1 будет правильным. Такую проверку можно сделать за одно прохождение по набору путем сравнения рядом стоящих дверей. Если не более одной пары рядом стоящих дверей различны, то такой набор является правильным.

Решение для тестов первой группы, в которых не больше трех золотых дверей и длина которых не больше 10. Понятно, что в этом случае ответ не превосходит 3. Попробуем перебрать все варианты изменения 0, 1 или 2 дверей и для каждого из них проверим, что полученный после текущего изменения набор дверей является правильным. После каждого изменения и проверки на правильность отменяем все изменения и проверяем следующий вариант. При этом вариант с 0 изменений (то есть без изменений) проверяется сразу, вариант с одним изменением проверяется одним циклом, а вариант с двумя изменениями проверяется двойным вложенным циклом. Максимальная сложность будет кубической с учетом проверки каждого варианта на правильность.

Решение для тестов второй группы, в которых длина набора не превосходит 100. В этом случае можно реализовать алгоритмы с тройным вложенным циклом. Например такой: переберем место встречи героев i (до позиции i невключительно мы хотим чтобы были двери одного типа, а после этой позиции включительно — двери другого типа). Далее для выбранной границы бежим по всем дверям и каждую дверь, которая не соответствует зафиксированному выше порядку дверей для позиции i изменяем, при этом увеличивая счетчик. Непосредственно после каждого изменения проверяем является ли набор после этого изменения правильным и если да — то сравниваем полученное до этого момента количество изменений с минимальным. Не забудем проверить два варианта расположения дверей (сначала двери типа 1, потом двери типа 2 и наоборот).

Для прохождения тестов третьей группы, в которых длина набора не больше 1000 нужно избавиться от одного вложенного цикла, то есть написать квадратичное решение. Можно заметить, что в предыдущем решении не обязательно проверять на правильность набор дверей после каждого изменения, достаточно это делать только после полного приведения дверей к порядку, зафиксированному при помощи позиции встречи i. Другой вариант с такой же сложностью (наиболее близкий к основному решению всей задачи) такой: мы для позиции встречи i и заданного этой позицией расположения дверей просто переберём все двери и подсчитаем количество дверей, которые нужно изменить слева, и количество дверей, которые нужно изменить справа, чтобы получить заданную расстановку дверей. Среди всех таких вариантов изменения найдем минимальный.

Для получения полного решения нужно написать линейное решение. После чтения последовательности подсчитаем сколько в ней цифр 1 и сколько цифр 2. Далее за один проход по набору получим ответ следующим образом: для каждой позиции i от 0 до длины набора минус один мы храним $left_1$ — сколько цифр 1 и $left_2$ — сколько цифр 2 было встречено на отрезке от начала строки до позиции $i - 1$ включительно (для $i = 0$ эти величины полагаем равными 0). Зная общее количество цифр по отдельности и величины $left_1$ и $left_2$, можно вычислить сколько раз встречается каждая цифра на отрезке от i до конца строки ($right_1$ и $right_2$ соответственно). Тогда для позиции i вычислим минимум из $left_1 + right_2$ и $left_2 + right_1$ — столько минимум нужно сделать замен, чтобы именно в этой позиции произошла встреча. Осталось выбрать самую маленькую по этому показателю позицию и вывести минимум для неё.

```
n = int(input())
a = [int(input()) for i in range(n)]
left_1 = 0
left_2 = 0
right_1 = a.count(1)
right_2 = a.count(2)
ans = min(right_1, right_2)
for c in a:
    if c == 1:
        right_1 -= 1
        left_1 += 1
    else:
        right_2 -= 1
        left_2 += 1
    ans = min(ans, left_1 + right_2, left_2 + right_1)
print(ans)
```

При написании и тестировании программы полезно проверить особые частные случаи.

Первый случай — когда двери изначально уже расположены так, что их можно открыть без изменений (тогда ответом будет число 0). Например, задана последовательность 2, 1, 1, 1. Очевидно, что если выдать тому кто у первой двери серебряный ключ, а тому кто у последней двери золотой, то можно открыть сразу все двери без их изменения. Важным подслучаем здесь является вариант, когда все двери уже изначально имеют один тип, например 2, 2, 2, 2, 2 или 1, 1, 1. Эти примеры тоже обязательно нужно протестировать.

Второй частный случай связан с вариантом, когда оптимально изменять только двери одного типа, например серебряные на золотые, и окончательный набор дверей так же является последовательностью дверей одного типа (в данном случае золотых). Например, последовательность 1, 1, 1, 1, 2, 1, 1, 1, 1. Для неё лучшим вариантом будет изменить за одну минуту дверь номер 5.

Задача 5. Долгое вычитание, Карл!

Частные решения.

Первая подзадача.

Если число n — однозначное, то ответ равен самому числу. Если число n — двузначное четное, то потребуется $\lfloor (n - 8)/2 \rfloor$ операций (через $\lfloor \cdot \rfloor$ обозначается целая часть от деления), чтобы число стало равно 8. Итого `ans = (n - 8) // 2 + 8`. Если число n — двузначное нечетное, то потребуется $\lfloor (n - 9)/2 \rfloor$ операций, чтобы число стало равно 9. Итого `ans = (n - 9) // 2 + 9`.

```
n = int(input())
if n % 2:
    ans = (n - 9) // 2 + 9
else:
    ans = (n - 8) // 2 + 8
if n < 10:
    ans = n
print(ans)
```

Вторая подзадача.

Моделирование процесса вычитания «в чистом виде»: пока число не обратится в 0, вычитать из него его длину, подсчитывая количество операций.

```
n = int(input())
ans = 0
while n:
    ans += 1
    n -= len(str(n))
print(ans)
```

Полное решение.

Разберем конкретный пример. Пусть $n = 123456$.

Найдем первое пятизначное число, встретившееся нам в процессе уменьшения n. Очевидно, оно имеет такой же остаток от деления на 6, что и n. Возьмем число 99999 и будем уменьшать его на 1, пока остатки не равны. Получим новое $n = 99996$. При этом мы использовали $\lfloor(123456 - 99996)/6\rfloor = 3910$ операций.

Далее аналогично. Найдем первое четырехзначное число, встретившееся нам в процессе уменьшения n. Очевидно, оно имеет такой же остаток от деления на 5, что и n. Возьмем число 9999 и будем уменьшать его на 1, пока остатки не равны. Получим новое $n = 9996$. При этом мы использовали $\lfloor(99996 - 9996)/5\rfloor = 18000$ операций (всего 21910).

Найдем первое трехзначное число, встретившееся нам в процессе уменьшения n. Очевидно, оно имеет такой же остаток от деления на 4, что и n. Возьмем число 999 и будем уменьшать его на 1, пока остатки не равны. Получим новое $n = 996$. При этом мы использовали $\lfloor(9996 - 996)/4\rfloor = 2250$ операций (всего 24160).

Найдем первое двузначное число, встретившееся нам в процессе уменьшения n. Очевидно, оно имеет такой же остаток от деления на 3, что и n. Возьмем число 99 и будем уменьшать его на 1, пока остатки не равны. Получим новое $n = 99$. При этом мы использовали $\lfloor(996 - 99)/3\rfloor = 299$ операций (всего 24459).

Наконец, найдем первое однозначное число, встретившееся нам в процессе уменьшения n. Очевидно, оно имеет такой же остаток от деления на 2, что и n. Возьмем число 9 и будем уменьшать его на 1, пока остатки не равны. Получим новое $n = 9$. При этом мы использовали $\lfloor(99 - 9)/2\rfloor = 45$ операций (всего 24504).

Не забудем, что n нужно уменьшить до 0, поэтому добавим к ответу еще 9. Итого 24513.

Запрограммируем этот процесс: будем последовательно находить наибольшие числа длины меньше, чем исходное на 1, которое встретится нам в процессе вычитания (и будет последним, полученным при вычитании длины исходного числа), пока исходное число не уменьшится до однозначного.

```
n = int(input())
ans = 0
while n > 9:
    new_n = int('9' * (len(str(n)) - 1))
    while new_n % len(str(n)) != n % len(str(n)):
```

```
            new_n -= 1
        ans += (n - new_n) // len(str(n))
        n = new_n
ans += n
print(ans)
```