Разбор задач

Задача 1. Творческая натура

Первый ряд плитки займет места: $k + n + k$. Каждый следующий ряд плитки «включается» в предыдущий ряд на глубину k, поэтому каждый следующий ряд добавляет $n + k$ к высоте зеркала. Исходя из этого можем записать формулу для t рядов зеркала: $H = (n + k) * t + k$

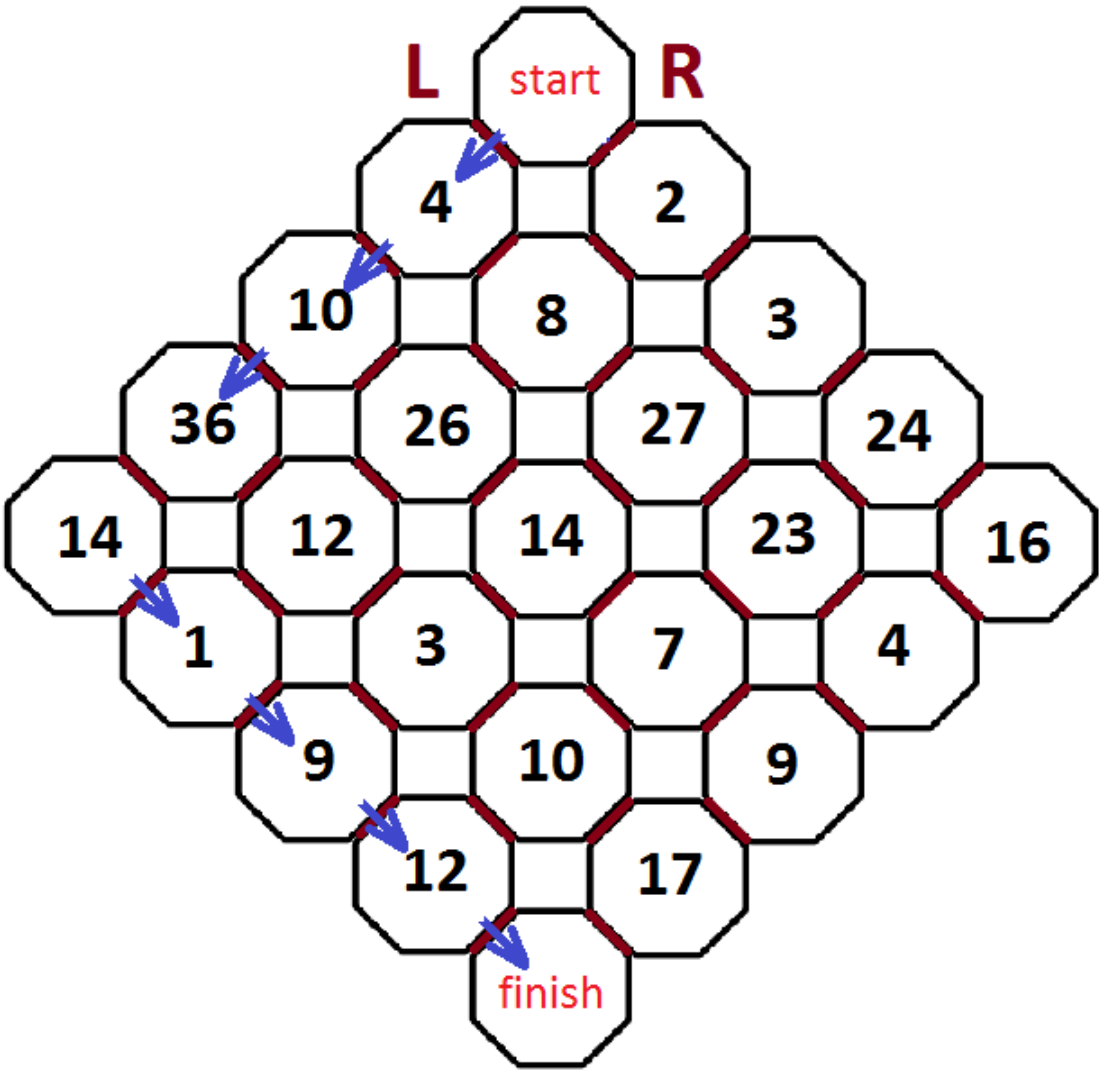
Так как по условию количество рядов всегда чётное, то количество плиток в первых двух рядах $m + m - 1$. Тогда количество плиток, составляющих всю зеркальную поверхность можем посчитать по формуле: $B = (2 * m - 1) * t/2$

Ответы:

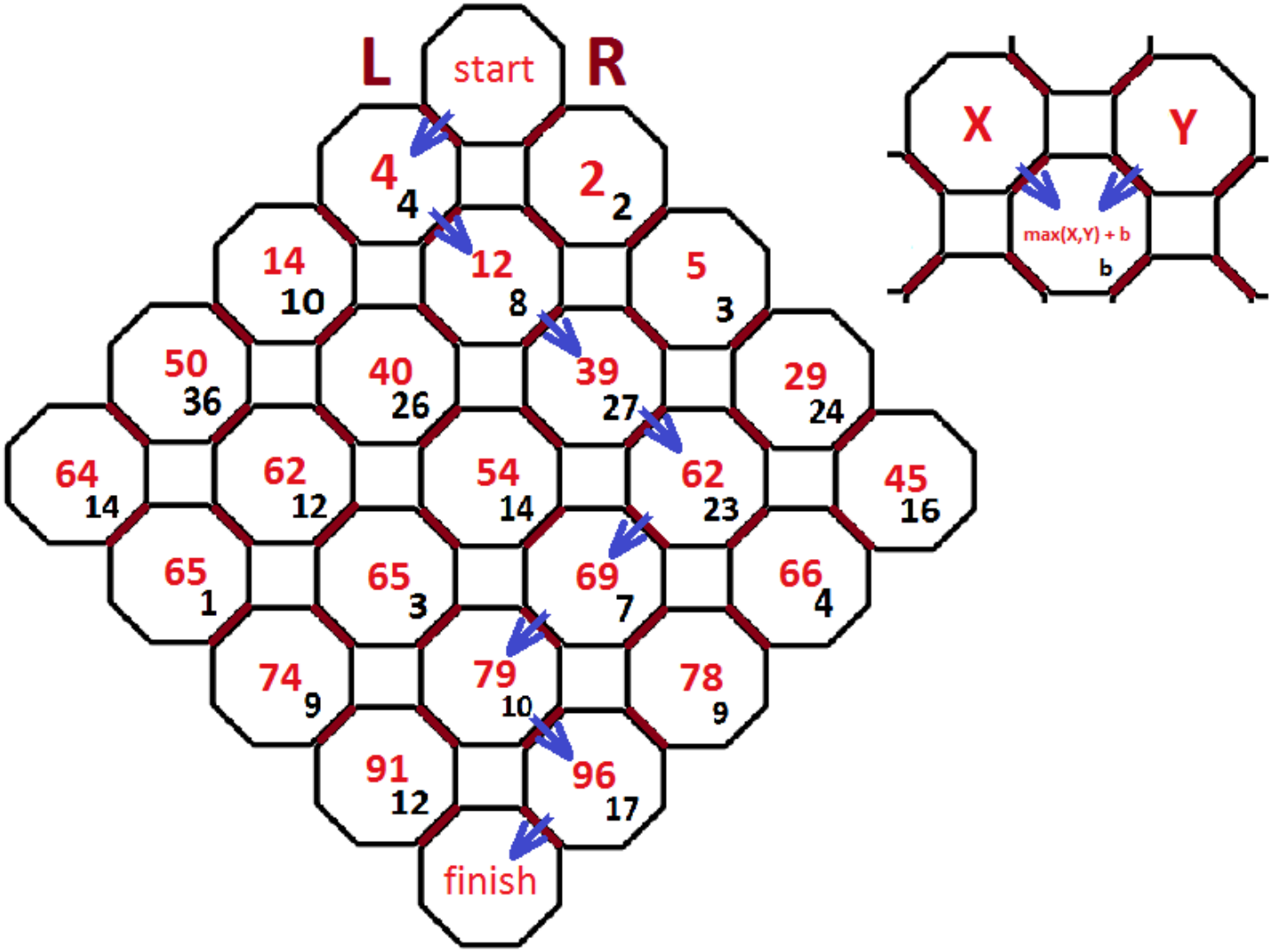
n	k	t	m	H	B
10	5	4	5	65	18
17	7	8	10	199	76
64	16	14	15	1136	203
99	33	20	25	2673	490
49	7	50	49	2807	2425

Задача 2. Лягушонок Пепе

Жадная стратегия – движение в сторону большего количества биткоинов – позволит набрать в данной задаче 50 баллов



Эффективное решение – метод динамического программирования – будем хранить для каждой комнаты **максимальную сумму**, которую можем получить двигаясь от старта до этой комнаты. При пересчете выбираем каждый раз из двух вариантов, соответствующих способам попасть в комнату, максимальный. Для восстановления ответа – пройдем от конца (последней комнаты) к началу в обратном порядке – каждый раз «двигаясь в сторону» максимального значения



Задачу можно решать простым перебором вариантов – в данном случае всего 70 вариантов, но, очевидно, более менее разумных не так много.

Ответ:

96

LRRRLLRL

Задача 3. Водолей

Перед тем, как решать задачу необходимо попробовать получить различный уровень воды в сосудах A и B.

Чтобы получить 6 литров в сосуде B необходимо дважды наполнить сосуд A, а затем перелить его содержимое в сосуд B.

Чтобы получить 4 литра в сосуде B необходимо наполнить сосуд B, а затем перелить его содержимое в пустой сосуд A. В сосуде B останется 4 литра.

Чтобы получить 1 литр в сосуде B необходимо наполнить сосуд B, а затем перелить его содержимое в пустой сосуд A, опустошить A и еще раз перелить воду в пустой сосуд A.

Чтобы получить 2 литра в сосуде A необходимо получить 6 литров в сосуде B, набрать воду в пустой сосуд A и перелить ее в сосуд B. В сосуде A останется 2 литра.

Заметим также, что:

- команда «наполнить A» не изменит состояние сосудов, если сосуд A полный
- команда «опустошить A» не изменит состояние сосудов, если сосуд A пустой
- команда «перелить в A» не изменит состояние сосудов, если сосуд A полный
- команда «перелить в A» не изменит состояние сосудов, если сосуд B пустой

Рассмотрим получение правильной последовательности для случая $A = 2, B = 7$. Для того, чтобы получить заданное количество воды нужно выполнить команды: «наполнить A», «перелить в B», «наполнить A», «перелить в B», «наполнить A», «перелить в B». Расставим полученные команды в заданной последовательности (в скобках укажем объем воды в каждом сосуде, после выполнения команды):

1. наполнить A $(A = 3, B = 0)$
2. перелить в B $(A = 0, B = 3)$
3. опустошить ...
4. наполнить A $(A = 3, B = 3)$
5. перелить в B $(A = 0, B = 6)$
6. опустошить ...
7. перелить в ...
8. опустошить ...
9. наполнить A $(A = 3, B = 6)$
10. перелить в B $(A = 2, B = 7)$

Оставшиеся позиции нужно заполнить так, чтобы количество воды в сосудах не менялось. В результате получаем: $ABAABABAAB$

Для остальных случаев ответ получаем аналогично.

Задача 4. Тайна Карбофоса

Заметим два важных факта:

- нет смысла поворачивать ручку несколько раз: любое нечётное количество поворотов ручки соответствует одному повороту, любое чётное количество поворотов равносильно состоянию, когда ручку не поворачивали
- порядок поворота ручек не важен

Закодируем положение ручек 0 - горизонтальное, 1 - вертикальное. Тогда начальное положение обозначим последовательностью 011010101. Построим таблицу, в которой первая строка соответствует начальному положению, а каждая следующая соответствует повороту одной из ручек (первой, второй, третьей и так далее)

Ручка	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
-	0	1	1	0	1	0	1	0	1
1	1					1			1
2		1	1					1	
3			1			1	1		1
4		1	1	1					
5	1			1	1	1			
6		1				1	1		
7				1	1		1	1	
8			1					1	1
9	1			1					1

Для получения ответа нужно выбрать строки таким образом, чтобы в каждом столбце получилось чётное количество единиц для выбранных строк. Поиск ответа начнем с ручки с номером 5, так как исходно она повернута вертикально и ее поворота можно добиться только одним из двух способов: непосредственным поворотом ручки 5 или поворотом ручки 7. Таким образом, в правильном ответе (если он существует) **обязательно** должна быть или ручка 5 или ручка 7, но не обе эти ручки.

Поворот ручки 5 приводит к положению 111101101. Ручки 5 и 7 нужно вычеркнуть, так как их больше поворачивать нельзя.

Ручка	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
-	1	1	1	1	0	1	1	0	1
1	1					1			1
2		1	1					1	
3			1			1	1		1
4		1	1	1					
6		1				1	1		
8			1					1	1
9	1			1					1

Следующая ручка, которую будем рассматривать - это ручка 1, так как она в настоящий момент повернута и при этом есть только два варианта, чтобы повернуть ее горизонтально: непосредственно повернуть ее или повернуть ручку 9. Попробуем повернуть ручку 1, тогда получим:

Ручка	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
-	0	1	1	1	0	0	1	0	0
2		1	1					1	
3			1			1	1		1
4		1	1	1					
6		1				1	1		
8			1					1	1

Очевидно, что ручку 4 поворачивать обязаны - это единственный вариант. После ее поворота получаем:

Ручка	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
-	0	0	0	0	0	0	1	0	0
2		1	1					1	
3			1			1	1		1
6		1				1	1		
8			1					1	1

Теперь требуется повернуть только ручку 7, это можно сделать либо повернув ручку 3, либо повернув ручку 6, однако, при этом оставшимися в рассмотрении ручками (2 и 8) развернуть все ручки в горизонтальное положение не получится.

Вернемся к шагу, на котором поворачивали ручку 1 и вместо этого повернем ручку 9. Получим:

Ручка	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
-	0	1	1	0	0	1	1	0	0
2		1	1					1	
3			1			1	1		1
4		1	1	1					
6		1				1	1		
8			1					1	1

В этом случае ручку 4, очевидно, также можно исключить, так как ее нельзя поворачивать. Для поворота ручки 2 есть два варианта: повернуть ее непосредственно или повернуть ручку 6. Повернем ручку 2 и получим:

Ручка	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
-	0	0	0	0	0	1	1	1	0
3			1			1	1		1
8			1					1	1

Повернув оставшиеся две ручки с номерами 3 и 8 получим горизонтальное положение для каждой ручки чемодана. Таким образом, получили один из двух вариантов верных ответов.

Варианты верных ответов (порядок ручек не влияет на баллы): 23589 или 23467

Замечания:

- для того, чтобы получить второй верный ответ, необходимо в качестве первой поворачиваемой ручки выбрать ручку с номером 7, а дальше действовать аналогично
- в этой задаче можно получить частичные баллы, если подобрать последовательность, в результате применения которой количество ручек, повернутых вертикально не будет больше 3
- задачу можно решить методом перебора всех вариантов на компьютере с использованием языка программирования, в этом случае потребуется перебрать 512 вариантов
- ускорить процесс подбора нужных строк можно с использованием электронных таблиц MS Excel (функции СУММ и ОСТАТ)

Задача 5. Фермер

Прочитаем данные и будем перебирать каждый килограмм кукурузы от 1 до n в цикле. Если номер очередного килограмма не делится на k, то добавляем к ответу стоимость этого килограмма a.

```
n = int(input())
a = int(input())
k = int(input())

ans = 0
for i in range(1, n+1):
  if i % k != 0:
    ans += a
print(ans)
```

Решение циклом можно ускорить. Для этого на каждом шаге будем рассматривать не по одному килограмму, а по k килограмм. Из каждых таких k килограмм к ответу будем прибавлять $(k-1)$ килограмм. Оставшееся количество килограмм (меньше k) добавим к ответу

```
n = int(input())
a = int(input())
k = int(input())

ans = 0
while n >= k:
    ans += a * (k - 1)
    n -= k

print(ans + a * n)
```

Последнее решение нетрудно превратить в формулу:

```
n = int(input())
a = int(input())
k = int(input())

print( (n // k * (k - 1) + n % k) * a )
```

В более компактном виде формула выглядит следующим образом: $(n - n//k) * a$. Здесь $n//k$ — количество килограмм, которые достанутся Джону бесплатно

Задача 6. Боинг

Для решения задачи необходимо получить номер места в ряду и номер ряда.

Чтобы получить номер ряда можно использовать формулу $(n+3)//6+1$ — считаем что в каждом ряду по 6 мест со смещением 3 от начала.

Для того, чтобы по номеру места получить номер буквы в латинском алфавите можно воспользоваться формулой: $(n+3)\%6$ — используется всего 6 букв со смещением 3 относительно начала алфавита.

Случаи, когда n принимает значения $117, 118, 119$ обработаем отдельно с использованием условного оператора.

```
n = int(input())
if n == 119:
  print("full")
elif n == 118:
  print("21E")
elif n == 117:
  print("21D")
else:
  s = "ABCDEF"
  j = (n + 3) % 6
  print((n + 3) // 6 + 1, s[j], sep="")
```

Задача 7. Интересное подмножество

Для неэффективного решения данной задачи необходимо написать перебор различных вариантов подмножества, например, с использованием рекурсии или битовых масок.

Отдельно отметим случай, в котором размер исходного множества равен 5 — в этом случае для получения решения можно написать 5 циклов для поиска ответа.

```
n = int(input())
ans = 0

if n == 5:
    a = [int(input()) for i in range(5)]

    for i0 in False, True:
      for i1 in False, True:
        for i2 in False, True:
          for i3 in False, True:
            for i4 in False, True:
              b = []
              if i0:
                b.append(a[0])
              if i1:
                b.append(a[1])
              if i2:
                b.append(a[2])
              if i3:
                b.append(a[3])
              if i4:
                b.append(a[4])
              for x in b:
                if x <= len(b):
                  break
              else:
                if len(b) > ans:
                  ans = len(b)
print(ans)
```

Для решения задачи на полный балл необходимо учесть, что все элементы множества попарно различны и подаются на вход в отсортированном по возрастанию виде.

Заметим, что для того, чтобы найти размер наибольшего интересующего нас подмножества, достаточно найти первый (наименьший) элемент исходного множества такой, что количество элементов больших его по значению в исходном множестве, меньше чем значение этого элемента.

```
n = int(input())
a = [int(input()) for i in range(n)]
for i in range(n):
  if a[i] > n - i:
    print(n - i)
    break
else:
  print(0)
```