Решения заданий

Задача А. Текстовый фрактал

Рассмотрим, как будут получаться созданные Мишей фракталы:

1.A
2.AAB
3.AABAABC
4.AABAABCAABAABCD
5.AABAABCAABAABCDAABAABCAABAABCDE

И так далее.

Можно заметить, что длина каждой строки получается по формуле $2^n - 1$, где n – номер шага. При этом символы в начале строки остаются неизменными на каждом следующем шаге.

Пользуясь формулой длины строки, понимаем, что символы на позициях 129 – 135 появятся уже на 8 шаге и в дальнейшем будут оставаться неизменными (новые символы будут приписываться в конце).

Эти символы можно получить либо выписав их в явном виде, либо написав простой программный код, либо рассуждениями.

На 6 шаге мы получим строку длиной 63 буквы (последняя – F)

На 7 шаге – длиной 127 букв (последняя E)

Значит, дальше на 8 шаге к полученной ранее строке будет приписан её повтор, начинающийся с AABAABCAABAABCD, причем номера этих символов будут

128	**129**	**130**	**131**	**132**	**133**	**134**	**135**	136	137
A	**A**	**B**	**A**	**A**	**B**	**C**	**A**	A	B

Значит, нужные нам символы: **ABAABCA**

Пример программы на языке Python для получения ответа

```
s = 'A'
for i in range(2, 21):
    s = s + s + chr(ord(s[-1]) + 1)
print('answer', s[128:135])
```

Пример решения на языке Pascal ABC.NET

```
var s:string;
    i: integer;
begin
  s := 'A';
  for i := 2 to 10 do
    s := s + s + chr(ord(s[length(s)])+1);
  writeln(s[129:136]);
end.
```

Задача B. Робот-мусорщик

Для решения нужно вычислить, какое максимальное количество мусора может собрать Робот, выполняя только указанные команды. Для каждой клетки таблицы вычислим, какое максимальное количество мусора может собрать Робот, дойдя до данной клетки, а затем по полученным числам восстановить маршрут его движения.

Для определения количества мусора, которое соберет Робот, двигаясь до нужной клетки, будем на каждом шаге определять, из какой клетки наиболее выгодно переместиться – из клетки слева или из клетки сверху, а затем прибавлять к уже ранее накопленному количеству мусора то, что хранится в текущей клетке.

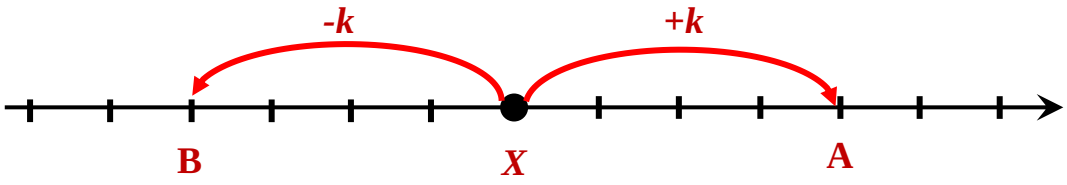
Таблица расчетов и соответствующий маршрут движения Робота:

9	20	26	38	53	64	74	79
18	28	40	50	64	77	89	102
33	**43**	**52**	**65**	70	88	98	107
45	57	64	**80**	**95**	**107**	122	130
50	67	73	91	105	**115**	128	140
56	74	88	106	114	**130**	140	146
70	79	98	120	129	**145**	**159**	166
77	85	108	129	142	156	**173**	**179**

Тогда алгоритм движения Робота задается последовательностью команд:
22111211222121

Задача C. Кузнечик

По условию, Кузнечик прыгнул из точки X на одно и то же расстояние k влево и вправо:



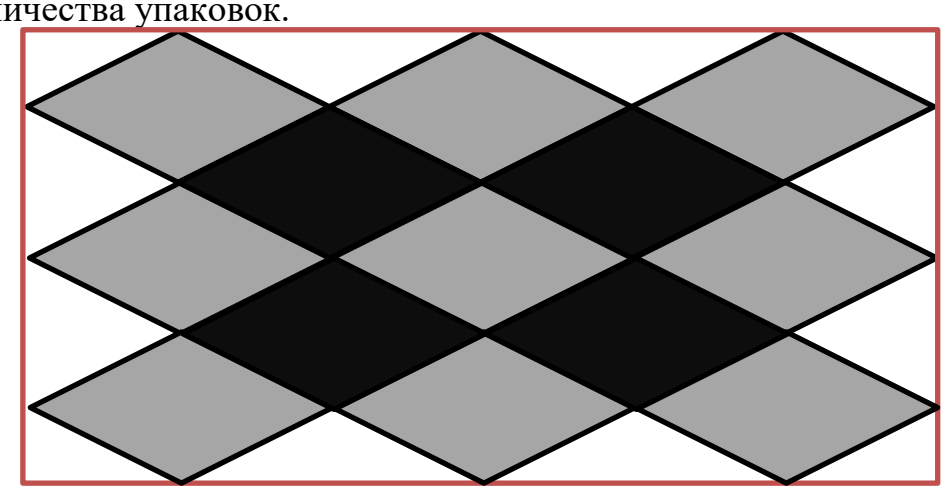
По рисунку видно, что точка X – это не что иное, как середина отрезка AB, а длина прыжка – половина длины отрезка AB.

Пример решения на языке Pascal ABC.NET

```
var a, b: integer;
begin
  read(a);
  read(b);
  writeln((a + b) div 2, ' ', max(a, b)-(a + b) div 2);
end.
```

Задача D. Укладка плитки

Выведем формулу для подсчета количества плитки, а затем для подсчета количества упаковок.



Посчитаем количество плитки в рядах, выделенных на рисунке серым цветом. Количество плитки в каждом ряду $\frac{N}{a}$, таких рядов – $\frac{M}{b}$, поэтому плиток в серых рядах будет $\frac{N}{a} \cdot \frac{M}{b}$

Можно заметить, что в рядах, выделенных черным, на одну плитку меньше, чем в серых. И самих рядов также на один меньше. Поэтому количество плиток в черных рядах $\left(\frac{N}{a} - 1\right) \cdot \left(\frac{M}{b} - 1\right)$

Тогда общее количество плитки:

$$\frac{N}{a} \cdot \frac{M}{b} + \left(\frac{N}{a} - 1\right) \cdot \left(\frac{M}{b} - 1\right)$$

А количество коробок плитки можно посчитать, разделив эту величину на K (округлив значение вверх до ближайшего целого)

Пример решения на языке Pascal ABC.NET

```
var n, m, a, b, k, tile, container: longint;
begin
   read(n, m, a, b, k);
   tile := (n div a)*(m div b)+(n div a -1)*(m div b - 1);
   container := (tile + k - 1) div k;
   write(container);
end.
```

Задача E. Бегуны

Формально задача сводится к подсчету количества элементов, которые меньше предыдущего ровно на k (тогда время второго бегуна вместе со временем ожидания своего старта равно времени первого спортсмена).

Пример решения на языке Pascal ABC.NET

```
var k, i, count, prev, next: integer;
begin
  read(k);
  count := 0;
  read(prev);
  while prev <> 0 do
  begin
    read(next);
    if prev - next = k then
      count := count + 1;
    prev := next;
  end;
writeln(count);
end.
```