1. Две доминошки

(Решение задачи — *Киндер М.И.*)

Две доминошки $(a\,;b)$ и $(c\,;d)$ можно поставить рядом только, когда одно из чисел в паре $(a\,;b)$ совпадает с одним из чисел в паре $(c\,;d)$, то есть когда $a = c$ или $a = d$ или $b = c$ или $b = d$. В каждом из этих случаев легко восстанавливается порядок расположения доминошек. Если не выполняется ни одно из этих условий, выводим –1.

```
if  a = c  then  write(b, ' ', a, ' ', a, ' ', d)  else
if  a = d  then  write(b, ' ', a, ' ', a, ' ', c)  else
if  b = c  then  write(a, ' ', b, ' ', b, ' ', d)  else
if  b = d  then  write(a, ' ', b, ' ', b, ' ', c)  else
write('–1');
```

2. Красивое число

(Автор задачи и решения — *Киндер М.И.*)

Если число n удовлетворяет условию, то чётные и нечётные цифры в его записи обязательно чередуются. Поэтому если в записи числа первая слева цифра нечётная (чётная), то все цифры на нечётных местах тоже должны быть нечётными (чётными), а все цифры на чётных местах — чётными (нечётными). Другими словами, каждые две *соседние* цифры в записи числа n имеют разные остатки при делении на 2. Эту проверку можно сделать за $O(n)$ операций.

```
flag := 1;
for  k := 1  to  n – 1  do
   if  m[k] mod 2  =  m[k + 1] mod 2  then flag := 0;
if  flag = 0  then  write('no');
if  flag = 1  then  write('yes');
```

3. Очередь

(Автор задачи и решения — *Киндер М.И.*)

Приведем решение подзадачи 1. Отметим очевидный факт: за фанатом, стоящим в конце очереди, никто не стоит. Поэтому для каждого i-ого фаната (первый элемент в i-ой строке) пытаемся найти фаната (среди вторых элементов), который стоит за ним. Как мы знаем, для фаната, стоящего в конце, такого нет. Действуя таким образом, за $O(n^2)$ операций мы найдем номер первого с конца очереди фаната. По этому номеру восстанавливаем всю очередь.

Приведем решение подзадачи 2. Очередь фанатов образует перестановку чисел от 1 до n, поэтому можно подсчитать количество вхождений каждого числа и определить два номера — начало и конец очереди, которые входят по одному разу. Одновременно в специальном массиве запоминаем номера на футболках фанатов, стоящих перед каждым из них. Для фаната, стоящего в конце очереди, этот номер (по умолчанию) равен 0. Определив номер последнего фаната, восстанавливаем всю очередь, используя специальный массив стоящих впереди фанатов. Итоговое время работы — $O(n)$.

4. Две дроби

(Автор задачи и решения — *Киндер М.И.*)

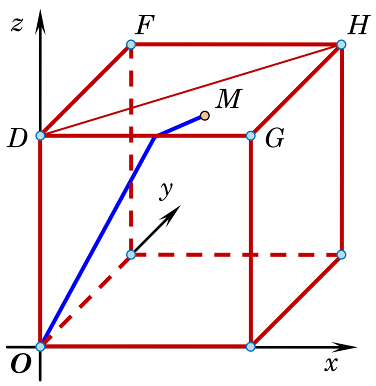
Приведем одно из возможных решений. Пусть $a\,/\,b$ — исходная, а $c\,/\,d$ — требуемая дробь. Сначала покажем, как из любой дроби можно получить число 1. Если $a > b$, применяем операцию B вычитания единицы. Каждая такая операция, очевидно, уменьшает числитель дроби на величину знаменателя. Используя эту операцию несколько раз (а именно, a div b), придём к дроби $a'\,/\,b$, у которой числитель a' будет уже меньше знаменателя b. Применяя операцию C, заменим эту дробь на обратную $b\,/\,a'$, у которой числитель b опять будет больше знаменателя a'. Вновь повторяем операцию B по уменьшению числителя до тех пор, пока числитель не будет меньше знаменателя, и так далее. Поскольку числитель и знаменатель дроби не может уменьшаться безгранично, через несколько шагов придем к дроби с числителем 1. Используя операцию C, получим дробь со знаменателем 1, то есть целое число. После нескольких операций вычитания приходим к 1. (Приведенный алгоритм, по сути, совпадает с алгоритмом Евклида вычисления наибольшего общего делителя чисел a и b.)

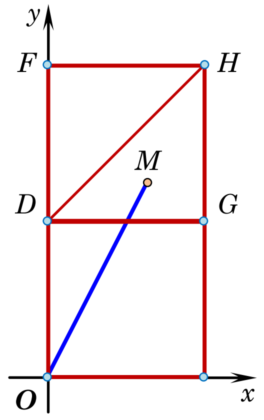
С помощью описанной процедуры любую несократимую дробь a / b можно привести к 1. И наоборот, из 1 можно получить несократимую дробь a / b. Для этого последовательность операций от a / b к 1 нужно записать в *обратном* порядке, причём каждую операцию B в этой последовательности заменить обратной к ней операцией A. (Обратная для операции C — сама операция C, поэтому заменять её не нужно.)

Теперь процесс преобразования дроби a / b в дробь c / d можно описать схемой $a / b \to 1 \to c / d$. Сложность алгоритма — $O(a + b + c + d)$.

5. Муравей на кубе

(Решение задачи — *Попов А.А.*)





Рассмотрим развёртку двух граней куба. Путь по поверхности куба перейдет в путь по развёртке. Ясно, что наименьшая длина достигается в случае, если путь представляет собой отрезок, соединяющий точки A и M.

Разберём подробно один из возможных случаев, когда точка M находится на верхней грани куба. (Если точка M лежит на боковых гранях куба, решение аналогично. В случае, когда точка M на нижней грани, длина кратчайшего пути легко вычисляется по теореме Пифагора.)

Если муравей находится внутри треугольника DGH (т.е. $x > y$), то $OM^2 = x^2 + (a + y)^2$.

Если муравей находится внутри треугольника DFH (т.е. $x < y$), то $OM^2 = y^2 + (a + x)^2$.

Если $x = y$, то справедлива любая из этих формул.

6. Факторизации

(Автор задачи и решения — *Киндер М.И.*)

Решение задачи основано на использовании идеи динамического программирования. Пусть $F(n, m)$ — количество неупорядоченных факторизаций числа n, у которых наибольший множитель не превосходит m. Расположим все делители числа n в невозрастающем порядке и рассмотрим произвольное разложение числа $n = d_1 \cdot d_2 \cdot \ldots \cdot d_k$ на множители с наибольшей частью $d_1 \le m$. Заметим, что все остальные множители в произведении $d_2 \cdot d_3 \cdot \ldots \cdot d_k = n \text{ div } d_1$ не превосходят d_1, поэтому количество таких факторизаций с *наибольшей частью* $d_1 \le m$ равно $F(n \text{ div } d_1, d_1)$. В качестве d_1 можно выбрать любой делитель d числа n, не превосходящий m. Суммируя по всем таким d, получим рекуррентное соотношение:

$$F(n,m) = \sum_{\substack{d \mid n \\ d \le m}}^{n} F(n \text{ div } d,\, d).$$

(Здесь сумма вычисляется во всем делителям d числа n.) Например,

$$F(10,10) = F(1,10) + F(2,5) + F(5,2) + F(10,1).$$

Техническая реализация этого алгоритма сравнительно несложная. Сначала находим список делителей $d[1]$, $d[2]$, ... , $d[j]$ числа n:

```
j := 0;
for i := 1 to m do
    if  n mod i = 0  then begin inc(j);  d[j] := i; end;
```

Требуемое количество факторизаций равно значению рекурсивной функции *Fact*(*n*, *j*):

```
Function Fact(n: integer; m: integer): integer;
if  m = 1 then begin Result := 0; exit; end;
if   n = 1 then begin Result := 1; exit; end;
s :=  0;
for  i := 1  to m  do
     if  n mod d[i] = 0  then inc(s, Fact(n div d[i], i));
Result := s;
end;
```

Председатель жюри М.И.Киндер