

Аргов Д.И.

Муниципальный этап
Всероссийской олимпиады
ШКОЛЬНИКОВ
по информатике

Рыбинск, 2016 г.

Задача А. Допинг (12 баллов)

- В Кабачке у Болванщика до утра горел свет – работала санэпидемстанция и антидопиновый комитет. Первая искала на кухне тараканов, второй – мельдоний в чае. Чтобы никто не догадался об уровне замеров (кроме комиссии WADA), их зашифровали. Для этого цифры числа перепутали и разбросали по строке, перемешав с различными буквами латинского алфавита. «Помоги мне, Алиса», - грустно сказал Болванщик, - «нужно выяснить уровень мельдония в крови у моих тараканов. Иначе СЭС потребует их усыпить, а я член партии защитников окружающей среды».
- **Помогите Алисе и Болванщику, напишите программу, которая выделит из заданной строки все цифры и соберет из них наибольшее и наименьшее число.**

- **Формат входного файла input.txt:**
- Единственная строка содержит последовательность символов (длина не более 255). Строка может содержать любые латинские буквы и цифры ($1 \leq \text{количество цифр} \leq 255$).
- **Формат входного файла output.txt:**
- Первая строка должна содержать наибольшее число, которое можно собрать из всех имеющихся цифр исходной строки.
- Вторая строка должна содержать наименьшее число, которое можно собрать из всех имеющихся цифр исходной строки. Ведущих нулей в числе быть не может, исключение – число 0, которое может содержать несколько 0 (например, 0000).

input.txt	output.txt
a4FF653RR00SD1a	6543100 1003456
A0B0c0d0	0000 0000

- `var f:text; i:integer; c:char;`
- `m:array['0'..'9'] of integer;`

	'0'	'1'	'2'	'3'	...	'9'
m	2	2	1	2	...	0

S `'a3FF213RR00SD1a'`

`'3321100'`

`'1001233'`

2 этап:
Составим наибольшее число.
Для этого выведем
имеющиеся цифры в
обратном порядке

3 этап:
Составим наименьшее число.
Для этого выведем
имеющиеся цифры в прямом
порядке. (!) **Ведущие нули!**

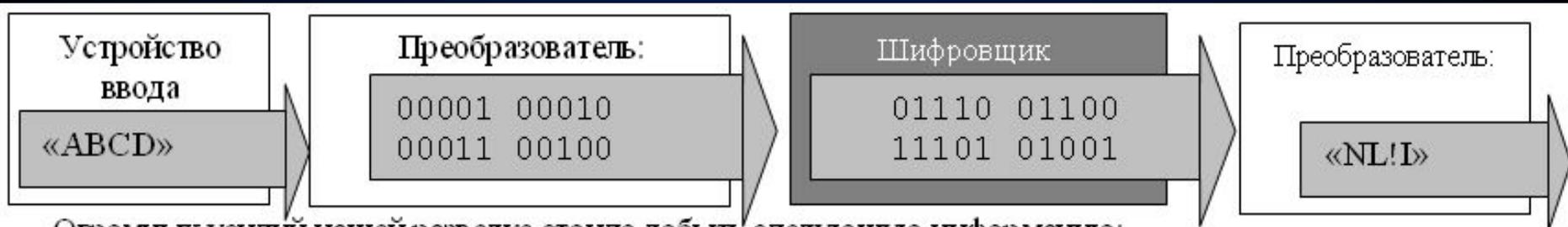
```

• assign(f, 'input.txt'); reset(f);
fillchar(m, SizeOf(m), 0);
• while not seekeof(F) do
• begin
• read(f, c); if c in ['0'..'9'] then inc(m[c])
• end;
• close(f);
• assign(f, 'output.txt'); rewrite(f);
• for c:='9' downto '0' do
• for i:=1 to m[c] do
• write(f, c);
• writeln(f);
• for c:='1' to '9' do
• if m[c]<>0 then break;
• if m[c]<>0
• then begin
• write(f, c); dec(m[c])
• end;
• for c:='0' to '9' do
• for i:=1 to m[c] do
• write(f, c);
• close(f);
• end.

```

Задача В. Черный ящик (15 баллов)

- Идущая сотни лет непрерывная война между силами Тьмы и Света требовала от Алисы определенных усилий. Первое, что необходимо было сделать, это выяснить, где Черная Королева хранит печеньки, приготовленные для раздачи на Озерной площади. Выяснив это, печеньки нужно будет незамедлительно уничтожить, то есть сорвать планы госдепа ЧК. Проблема состояла в том, что все данные о месте хранения печенок были зашифрованы при помощи шифровального аппарата под кодовым названием «Черный ящик». Его задача состоит в шифровке самых секретных сведений. На вход «черного ящика» поступает текстовая информация (обычно с клавиатуры компьютера), на выход – тоже текстовая информация, но уже в зашифрованном бессмысленном виде.
- «Черный ящик» состоит из преобразователей, которые переводят текст из буквенного вида в двоичную последовательность 0 и 1 и обратно, а так же шифровщика, который шифрует последовательность нулей и единиц по некому правилу.



Огромных усилий нашей разведке стоило добыть следующую информацию:

Исходный текст	Последовательность кодов			Зашифрованная последовательность			Выходной закодированный текст
BE OR NOT TO BE?	00010	00101	00000	01100	11000	01111	LXOWT;VWQ;AWO,X?
	01111	10010	00000	10111	10100	11111	
	01110	01111	10100	10110	10111	10001	
	00000	10100	01111	11111	00001	10111	
	00000	00010	00101	01111	11100	11000	
	11110			11110			

- Известно, что во входном тексте могут использоваться заглавные латинские буквы, а также пробелы и знаки препинания. Благодаря деятельности сверхсекретного агента Соня Мышь, нашей разведке удалось узнать алгоритм работы ЧЯ.

' '	'A'	'B'	'C'	'D'	...	'Z'	'.'	','	'!'	'?'	'.'
00000	00001	00010	00011	00100	...	11010	11011	11100	11101	11110	11111

- **Алгоритм работы шифровщика:**
 1. Первый бит в исходной и зашифрованной последовательностях сохраняется;
 2. начиная со второго бита, движемся далее, и сравниваем текущий $s[i]$ бит и предыдущий $s[i-1]$. Если они равны, то в выходную последовательность записываем 1, если нет, то - 0.
 3. Полученная последовательность бит разбивается на группы по 5 штук, преобразуется в символы и сохраняется в зашифрованный файл.
- **Задание: Помогите Алисе расшифровать файл, добытый нашей разведкой.**
- **Формат входного файла input.txt:**
 - В единственной строке входного файла содержится последовательность заглавных латинских букв, знаков препинания и пробелов, заканчивающаяся символом '#'
($1 \leq \text{длина строки} \leq 1000$).
- **Формат выходного файла output.txt:**
 - Выведите в единственной строке выходного файла расшифрованный текст сообщения без символа '#'

input.txt	output.txt
DNDNOTBE?OD?DPY#	MAMA MILA RAMU.
SBUIWDORUOQFWSBF.OSWSOURQAUXOCH OQFWSY#	WISDOM IS KNOWING HOW LITTLE WE KNOW.

```
function DecToBin(x:integer):string;
var s:string; i:integer;
begin
  s:='';
  for i:=1 to 5 do
    begin
      s:=chr(ord('0')+x mod 2)+s;
      x:=x div 2
    end;
  DecToBin:=s
end;
begin
  Ch[' ']:='00000';Ch['.']:='11011';Ch[',']:='11100';
  Ch['!']:='11101';Ch['?']:='11110';Ch[';']:='11111';
  i:=0;
  for c:='A' to 'Z' do
    begin
      inc(i);
      Ch[c]:=DecToBin(i);
    end;
end;
```

Преобразует десятичное
число в двоичное

Построим массив кодов всех
СИМВОЛОВ

```
assign(f, 'input.txt'); reset(f);
n:=0;
repeat
  read(f, c);
  if c<>'#'
  then begin
    s:=Ch[c];
    for i:=1 to 5 do
      begin
        inc(n); rs[n]:=s[i];
      end;
    end;
until c='#';
close(f);

r[1]:=rs[1];
for i:=2 to n do
  if rs[i]='1'
  then r[i]:=r[i-1]
  else if r[i-1]='1'
    then r[i]:='0'
    else r[i]:='1';
```

Считаем входной текст,
преобразуем его в двоичные
коды

Расшифруем его.
Если очередной бит равен 1,
то сохраняем его, если – 0,
то меняем на
противоположный

```
assign(f, 'output.txt');rewrite(f);
s:='';
for i:=1 to n do
begin
  s:=s+r[i];
  if length(s)=5
  then begin
    c:=' ';
    while Ch[c]<>s do
      inc(c);
    s:='';
    write(f,c);
  end;
end;
close(f);
```

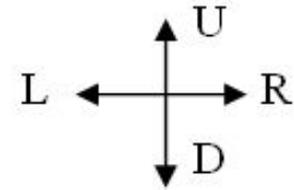
Разобьем двоичный код на группы по 5 бит, найдем их буквенный эквивалент

Задача С. Битва Змеек (20 баллов)

- Несколько лет назад тот, чье имя нельзя произносить, решил проникнуть из волшебного мира Гарри Поттера в обыденность Зазеркалья и уничтожить живые краски зазеркального мира. Для этого он послал в Зазеркалье свою змею Нагайну, которая должна покусать всех жителей или, в крайнем случае, обшипеть. Мир, как всегда, спасла Алиса, вырастив Волшебную Змейку. С тех пор в Зазеркалье получила популярность игра «Битва Змеек». В начале игры каждому игроку давалась только что вылупившаяся Змейка. Чтобы ее вырастить, нужно накормить ее волшебными яблоками, которые падают на землю в саду Черной Королевы. Сад представляет собой прямоугольное поле, разбитое на клетки. Каждая клетка может быть пустой, содержать камень или яблоко одного из трех цветов. Змейка умеет ползать только вперед и есть яблоки, встречающиеся на ее пути. Чтобы управлять Змейкой существует программа, состоящая из простых команд L, R, U, D, которые означают повернуть голову Змейки влево, вправо, вверх и вниз соответственно, а затем сместить каждое звено тела Змейки на одну позицию вперед. Например, выполняя команду L, Змейка поворачивает голову влево, смещает ее в клетку игрового поля, расположенную слева от текущей. Затем каждое звено тела Змейки смещается в клетку, которую занимало предыдущее звено.

Игра прекращается, если:

- обе Змейки выполнили все команды программы;
- голова любой Змейки вышла за пределы игрового поля;
- голова любой Змейки встретила на своем пути камень;
- голова Змейки врезалась в собственное тело или любую часть Змейки противника.



.
.	.	@	.	1	.
.	.	#	.	.	.
.	.	#	#	#	.
.	.	.	.	#	.
.	.	*	.	.	.

Исходное состояние игрового поля

.
.	@	#	.	1	.
.	.	#	.	.	.
.	.	#	#	#	.
.
.	.	*	.	.	.

После выполнения команды L

.
.	.	#	@	1	.
.	.	#	.	.	.
.	.	#	#	#	.
.
.	.	*	.	.	.

После выполнения команды R

.	.	@	.	.	.
.	.	#	.	1	.
.	.	#	.	.	.
.	.	#	#	#	.
.
.	.	*	.	.	.

После выполнения команды U

- После того, как голова Змейки переместилась в клетку с яблоком (съела очередное яблоко любого цвета), тело остается на месте, то есть длина Змейки увеличивается на одну клетку.

- Игроки могут управлять своей Змейкой, заставляя ее перемещаться в выбранном направлении (при помощи команд L, R, D, U). Выиграет тот игрок, у кого после окончания игры Змейка окажется длиннее. Обычно игроки ходят по очереди, но если Змейка съела яблоко, ее хозяин получает право на дополнительные ходы. Количество дополнительных ходов определяется цветом яблока:
 - зеленое яблоко дает 1 дополнительный ход;
 - желтое – 2 дополнительных хода;
 - красное – 3 дополнительных хода;
- **Задание: По имеющейся карте игрового поля и списку команд управления Змейками, определите победителя игры или сообщите о том, что произошла ничья.**

- **Формат входного файла input.txt:**
- Первая строка содержит два числа N и M , разделенные пробелом. ($2 \leq N$, $M \leq 100$) - размеры игрового поля.
- Вторая строка содержит четыре числа – координаты Змеек первого и второго игрока (номер строки и номер столбца). Змейки могут располагаться только в пустых клетках поля и их первоначальные координаты не совпадают.
- В следующих N строках по M символов в каждой задается само поле (**нет пробелов!**):
- символ '.' (точка) означает, что это пустая клетка;
- символ '@' обозначает голову Змейки;
- '#' (только на рисунках) – обозначает тело змеи, первоначально длина тела (хвоста) всегда 0 звеньев – Змейка только что вылупилась из яйца, но в процессе поедания яблок длина хвоста будет расти;
- '1', '2', '3', - значит, что в этой клетке лежит яблоко зеленого, желтого или красного цвета соответственно;
- '*' - означает камень.
- В следующей строке дается число K ($1 \leq K \leq 2000$) - количество команд, которые заданы змейкам. K может быть больше количества команд, которые выполнят змейки.
- В последней строке задается строка из K символов – последовательность команд, задающих передвижение Змеек. Первый ход принадлежит первому игроку, второй – как первому, так и второму, это определяется тем, съела Змейка игрока яблоко или нет. Если Змейка первого игрока съела красное яблоко, то она получает право на +3 дополнительных хода. Если в течение этих ходов она съест еще яблоки, то дополнительные ходы суммируются.
- Гарантируется, что входные данные корректны и две последовательные команды алгоритма не будут заставлять Змейку двигаться в противоположном направлении.

- **Формат выходного файла output.txt:**
- В первой строке вывести единственное число – номер победителя (1 или 2), если произошла ничья, то вывести 0.
- Во второй строке вывести два числа, разделенных пробелом – длины змеек первого и второго игроков.
- **Особенности тестов:**
- не менее 5 тестов только с зелеными яблоками;
- порядка 15 тестов с тремя типами яблок;
- 10 тестов с программой длиной не более 255 команд, 10 тестов – не более 2000.

input.txt	output.txt
<pre> 4 5 1 1 4 5 @.3.* .1.2.*1.@ 14 RLRRDLLLLUULR </pre>	<pre> 1 4 2 </pre>
<pre> 4 5 1 1 4 5 @.31* .111. .1... .*11@ 14 RLURRDLLLDURR </pre>	<pre> 1 6 3 </pre>
<pre> 5 6 1 1 5 6 @11.2. ..13.2 2.*... 1.1.1@ 19 RRLLRRDLLLLLDLLDRR </pre>	<pre> 1 8 3 </pre>

```
Procedure Load(NameF:string;Var a: tMatr; var
  Z,N,M,K:integer; Var Pr:tPr);
Var i,j:integer;
  f:text;
begin
  assign(f,NameF);reset(f);
  readln(f,N,M);
  readln(f,Zi[0], zj[0], zi[1], zj[1]);
  for i:=1 to N do
    begin
      for j:=1 to M do
        begin
          read(f,a[i,j]);
          end;
        readln(f);
      end;
    readln(f,K);
    for i:=1 to K do read(f,Pr[i]);
    close(f);
  end;
```

```
Procedure Solve (Var a: tMatr; Z,N,M,K:integer; Pr:tPr);

Var i,j, xod:integer;
    XVi,XVJ:byte;
    f:text;

procedure PutQ(xi,xj:byte);
begin
    xv[Gamer]:=xv[Gamer] mod 10000 +1;
    Q[Gamer][Xv[Gamer]].i:=Xi;  Q[Gamer][Xv[Gamer]].j:=Xj;
    inc(L[Gamer]);
end;

procedure GetQ(var xi,xj:byte);
begin
    xi:=Q[Gamer][g[Gamer]].i;  xj:=Q[Gamer][g[Gamer]].j;
    dec(L[Gamer]);
    g[Gamer]:=g[Gamer] mod 10000 +1;
end;
```

```

Gamer:=0;  xod:=1;
g[0]:=1;xv[0]:=0;l[0]:=0;  g[1]:=1;xv[1]:=0;l[1]:=0;
for i:=1 to k do
begin
  case Pr[i] of
    'L': if zj[Gamer]=1 then break
        else begin
            di[Gamer]:=0;dj[Gamer]:=-1;
            end;
    'R': if zj[Gamer]=M then break
        else begin
            di[Gamer]:=0;dj[Gamer]:=1;
            end;
    'U': if zi[Gamer]=1 then break
        else begin
            di[Gamer]:=-1;dj[Gamer]:=0;
            end;
    'D': if zi[Gamer]=N then break
        else begin
            di[Gamer]:=1;dj[Gamer]:=0;
            end;
  end;
end;

```

```

case a[zi[Gamer]+di[Gamer],zj[Gamer]+dj[Gamer]] of
  '*','#','@': break;
  '.': begin
    a[zi[Gamer],zj[Gamer]]:='#';
    PutQ(zi[Gamer],zj[Gamer]);
    zi[Gamer]:=zi[Gamer]+di[Gamer];
    zj[Gamer]:=zj[Gamer]+dj[Gamer];
    a[zi[Gamer],zj[Gamer]]:='@';
    GetQ(XVi,XVj);
    a[XVi,XVj]:='.';
  end;
  '1'..'3':begin
    xod:=xod+(ord(a[zi[Gamer]+di[Gamer],
      zj[Gamer]+dj[Gamer]])-ord('0'));
    a[zi[Gamer],zj[Gamer]]:='#';
    PutQ(zi[Gamer],zj[Gamer]);
    zi[Gamer]:=zi[Gamer]+di[Gamer];
    zj[Gamer]:=zj[Gamer]+dj[Gamer];
    a[zi[Gamer],zj[Gamer]]:='@';
  end;
end;
dec(xod);
if Xod=0
then begin   xod:=1; Gamer:=1-Gamer;   end;
end;

```

```
    assign(f, 'output.txt');  
    rewrite(f);  
    if l[0]>L[1] then writeln(f,1)  
    else if L[1]>L[0] then writeln(f,2)  
        else writeln(f,0);  
    Writeln(f,l[0]+1,' ',l[1]+1);  
    close(f);  
end;  
BEGIN  
    Load('input.txt',a,Z,N,M,K,Pr);  
    Solve(a,Z,N,M,K,Pr);  
END.
```

Задача D. Волшебный замОк (30 баллов)

Как выяснила наша разведка, печенки для подкупа зазеркальной оппозиции Черная Королева хранила в огромном сейфе-холодильнике. Попробовав такую печеньку, обычный житель Зазеркалья полностью терял разум, у него начинались галлюцинации (видел то, чего не было), и он начинал скакать, как заведенный. Прыжки продолжались до тех пор, пока у жителя не отказывали ноги. Алиса решила уничтожить столь опасное «угощение» и, тем самым, спасти жителей Зазеркалья. Доступ к недрам сейфа охранял не огнедышащий дракон, не злобный крокодил Гена, а волшебный замок. На его дисплее выводились два натуральных числа X и Y . Чтобы открыть замок, пользователь должен был расставить между цифрами числа X знаки арифметических операций (+, -, *) таким образом, чтобы в результате вычислений арифметического выражения получилось число Y .

Помогите Алисе вскрыть сейф Черной Королевы, напишите программу, которая по данным числам X и Y выведет все возможные варианты

- **Формат входного файла input.txt:**
- В единственной строке файла находятся два натуральных числа X и Y ($9 < X \leq 2\,000\,000\,000$, $1 \leq Y \leq 2\,000\,000\,000$).
- **Формат выходного файла output.txt:**
- Выходной файл должен содержать арифметическое выражение, состоящее из цифр числа X (порядок цифр должен быть сохранен), знаков арифметических операций (+, -, *). Если можно получить несколько правильных вариантов, то выведите все в лексикографическом порядке. Если арифметическое выражение построить невозможно, то вывести сообщение no solution. Число 0 может состоять только из одной цифры, 00 – нельзя использовать в выражении, ведущие нули в операнде запрещены (например, 05, 021).

input.txt	output.txt
22 4	2*2 2+2
22222 66	2*22+22 22*2+22 22+2*22 22+22*2
51 7	no solution
11456 16	1*1+4+5+6 1+1*4+5+6 1+1+4*5-6 1+14-5+6 11+4-5+6

- **Примечание:** лексикографический порядок – это порядок расположения символов в строке по возрастанию их кодов в кодовой таблице. Две строки располагаются в лексикографическом порядке, если первая строка меньше второй. Первая строка считается меньше второй, если первый несовпавший символ двух строк меньше по коду, при их равенстве – более короткая строка.
 - '*' < '+' < '-' < '0' < '1' < ... < '9'

```
type tRes=array[1..2550] of string[50];
var StIn,StOut:string;
    X,Y:longint;
    N,RN:integer;
    Res:tRes;
    Err:boolean;
    fIn, fOut:text;
    c:longint;
Const Max=5;
    Zn:array[1..5] of char=('*','+','-','(',')');
```

```
Function Calc(S:string):longint;
var i, code:integer;
```

Вычисляет значение строки S

```
function Numb(s:string):boolean;
var i:integer;
begin
    Numb:=true;
    for i:=1 to length(s) do
        if not(s[i] in ['0'..'9'])
            then Numb:=false;
```

Получает числовое значение
строки S

```
end;
```

```
function Mul:longint; Forward;  
function Factor:longint; Forward;
```

Описание процедур,
которые
реализуются ниже

```
function Add:longint; {суммирует слагаемые}  
var q,res:longint;    c:char;  
Begin    res:=Mul; {первое слагаемое}  
    While s[i] in ['+', '-'] do    Begin  
        c:=s[i]; i:=i+1; q:=Mul; {очередное слагаемое}  
        case c of  
            '+':res:=res+q;  
            '-':res:=res-q;  
        End  
    End; {While}  
    Add:=res  
End;
```

```
function Mul:longint; {перемножает множители}
var q,res:longint;      c:char;
Begin
  res:=Factor;{первый множитель}
  While (s[i] in ['*','/'])and(i<=length(s)) do
  Begin
    c:=s[i];i:=i+1;q:=Factor;{очередной множитель}
    case c of
      '*':res:=res*q;
      '/':If q=0 then Begin
          writeln('деление на 0'); halt End
        else res:=res div q
      End {case}
    End; {While}
  Mul:=res
End;
```

```

function Number:longint;{}
var res:longint;k:integer;
Begin  res:=0; k:=i;
  While (i<=length(s)) and (s[i] in ['0'..'9']) do Begin
    res:=res*10+(ord(s[i])-ord('0')); i:=i+1
  End;
  if (res=0)and(i-k>1)or(s[k]='0')and(i-k>1) then Err:=true;
  Number:=res
End;

```

```

function Factor:longint; {выделяет множитель}
var q:longint; c:char;
Begin
  case s[i] of
    '0'..'9':Factor:=Number;
    '(':Begin i:=i+1;Factor:=Add; i:=i+1;{пропустили ')'}End;
    '-': Begin i:=i+1; Factor:=-Factor; End
    else Begin Err:=true End
  End {case}
End;

```

```
Begin {основная подпрограмма Calc}  
  if Numb(s)  
  then Val(StOut,C,code)  
  else begin  
      i:=1; Err:=false ;C:=Add  
      end;  
  Calc:=c  
end;
```

```

Function ChRang(S:string):boolean;
var i,R,Z,k:integer; Zn:Boolean;
begin
  r:=0;ChRang:=true;z:=0;
  for i:=1 to Length(s) do
    case s[i] of
      '+','-','*':Zn:=true;
    end;
  ChRang:=(R=0); if r<>0 then exit;
  z:=0;
  for i:=1 to Length(s) do
    if (s[i]=')')and(s[i-1]='(')
    then begin
      k:=i-1; r:=1;
      while r<>0 do
        begin
          case s[k] of
            ')':inc(r);
            '(':dec(r);
          end;
          dec(k);
        end;
      if (S[k+1]='(')and(s[k+2]='(')
      then ChRang:=false
    end
  end;
end;

```

```
Function Check:boolean;
```

```
begin
```

```
    Check := (Calc (StOut) = Y) and (not  
    Err)
```

```
end;
```

```

procedure Rec(k,Z,R:integer);
var i,L:integer;
begin
if k>N
then begin
    if Check {если выражение корректно, то запомним его}
    then begin
        inc(RN);
        Res[RN]:=StOut;
    end
end
else begin
    StOut:=StOut+StIn[k]; L:= Length(StOut);
    Rec(k+1,0,R);
    if (Z=0) and (k<>1) and (StOut[L-1] in ['0'..'9'])
    then for i:=1 to 3 do {переберем все знаки}
        begin
            StOut[L]:=Zn[i];
            Rec(k,1,R)
        end;
    Delete(StOut,L,1);
end
end;
end;

```

```
begin
  Assign(fOut, 'input.txt');    reset(fOut);
  read(fOut, x, y); close(fOut);
  Str(X, StIn);
  Assign(fOut, 'output.txt');    reWrite(fOut);
  StOut := ''; N := Length(StIn);
  RN := 0;
  Rec(1, 0, 0);
  if RN = 0 then writeln(fOut, 'no solution')
  else begin
    Sort(Res, RN);
    for i := 1 to RN do
      Writeln(fOut, Res[i]);
    end;
  close(fOut);
end.
```

Задача Е. «Четыре площади» (25 баллов)

- «Всем привет», - сказала Алиса, заходя в кабачок Болванщика, ответа не последовало... Вместо ответного приветствия, в Алису полетела чайная чашка, которая со звоном разбилась о входную дверь. «Не братья мы больше», - вопил Мартовский Заяц, - «ты мухлюешь». «Да ты бредишь на всю свою больную голову», - кричал Болванщик, - «печенек, наверное, объелся!» «Давайте успокоимся, друзья, и спокойно во всем разберемся», - мудро предложила Алиса, - «рассказывайте, в чем причина ссоры». «Причина – в печеньках господпа, а ссора вышла из-за спора, кто победил», уже тише проговорил Болванщик, - «у меня и записи имеются».
- «Четыре площади» - очень древняя игра Зазеркалья, многие жители баловались ею длинными зимними вечерами. Игрокам дается прямоугольник $N \times M$ клеток, каждая клетка может быть белой или черной. Необходимо провести две линии (вертикальную и горизонтальную), которые разделят четырехугольник на 4 прямоугольника таким образом, чтобы произведение четырех «белых площадей» было максимальным. Под «белой площадью» понимается количество белых клеток в данном прямоугольнике.
- **Помогите Алисе решить столь непростую задачу.**

- **Формат входного файла input.txt:**
- Первая строка содержит два числа N и M, разделенные пробелом. ($2 \leq N$, $M \leq 450$) - размеры игрового прямоугольника.
- В следующих N строках записано по M символов: 1 – белая клетка, 0 – черная (нет пробелов!):
- **Формат выходного файла output.txt:**
- Вывести единственное число – максимальное произведение четырех белых площадей. Гарантируется, что такое число может быть получено.

Input.txt	Output.txt
<pre> 6 10 01101 00111 11000 01000 01111 10000 10001 01001 10000 11111 01100 00011 </pre>	<pre> 2025 {Пояснение: 9*5*5*9=2025 Граница областей проходит по 3 строке и 5 столбцу, пробелов в файле не будет} </pre>
<pre> 5 14 00100101100011 10001000110100 10111000101101 11110010100101 00110111001011 </pre>	<pre> 5544 {3 6 8*11*7*9=5544} </pre>

Особенности тестов:

- $2 \leq N$, $M \leq 10$ – 5 тестов;
- $2 \leq N$, $M \leq 100$ – 10 тестов;
- $2 \leq N$, $M \leq 450$ – 10 тестов.

```
const max=1000;
Type matr=array[1..max,1..max] of byte;
      Smatr=array[1..max,1..max] of longint;
var a:matr;
      s:smatr;
      n,m:integer;
procedure Load(var a: matr; var n,m:integer);
var f:text;      i,j:integer;      c:char;
begin
  assign(f,'input.txt');reset(f);
  readln(f,n,m);
  for i:=1 to n do
    begin
      for j:=1 to m do
        begin
          read(f,c);
          a[i,j]:=ord(c)-ord('0')
        end;
      readln(f);
    end;
  close(f);
end;
```

```

procedure Solve2;
var i,j,k,ii,jj,iii,jjj:integer;
    f:text;p1,p2,p3,p4,pm1,pm2,pm3,pm4,pm,max:int64;
begin
    Max:=1; k:=0;
    for i:=1 to n-1 do
        for j:=1 to m-1 do
            begin
                p1:=0; p2:=0; p3:=0; p4:=0;
                for ii:=1 to i do
                    for jj:=1 to j do
                        p1:=p1+a[ii,jj];
                for ii:=1 to i do
                    for jj:=j+1 to m do
                        p2:=p2+a[ii,jj];
                for ii:=i+1 to n do
                    for jj:=1 to j do
                        p3:=p3+a[ii,jj];
                for ii:=i+1 to n do
                    for jj:=j+1 to m do
                        p4:=p4+a[ii,jj];
                pm:=p1*p2*p3*p4;
                if pm>max
                then begin
                    pm1:=p1; pm2:=p2; pm3:=p3; pm4:=p4 ;           max:=pm; iii:=i;jjj:=j;
                    end
                else if pm=max           then inc(k);
                end;
            assign(f,'output.txt');rewrite(f);   writeln(f,max);   close(f);
        end;
    end;
end;

```

«Лобовое» решение, которое не успевает на больших тестах

```

procedure Solve;
var i,j,k,ii,jj:integer;    f:text;
    p1,p2,p3,p4,pm1,pm2,pm3,pm4,pm,max:int64;
begin
  s[1,1]:=a[1,1];
  for j:=2 to m do  s[1,j]:=s[1,j-1]+a[1,j];
  for i:=2 to n do  s[i,1]:=s[i-1,1]+a[i,1];
  for i:=2 to n do
    for j:=2 to m do
      s[i,j]:=s[i,j-1]+s[i-1,j]-s[i-1,j-1]+a[i,j];
  Max:=1; k:=0;
  for i:=1 to n-1 do
    for j:=1 to m-1 do
      begin
        p1:=s[i,j];          p2:=s[i,m]-s[i,j];
        p3:=s[n,j]-s[i,j];  p4:=s[n,m]-s[i,m]-s[n,j]+s[i,j];
        pm:=p1*p2*p3*p4;
        if pm>max
        then begin
          pm1:=p1; pm2:=p2; pm3:=p3; pm4:=p4 ; max:=pm; ii:=i;jj:=j;
        end
        else if pm=max      then inc(k);
      end;
  assign(f,'output.txt');rewrite(f);
  writeln(f,max);
  close(f);
end;

```

«Умное» решение,
которое успевает на
больших тестах

Задача F. Диадема Клеопатры (10 баллов)

- Аня — страстный любитель ювелирных изделий. Ее коллекция насчитывает множество бриллиантов, изумрудов и алмазов.
- *...Срочная новость! Бесценный змеиный рубин Клеопатры был украден!*
- *Три дня назад мир потрясло сенсационное известие: исследовательская экспедиция обнаружила в одном из храмов, построенных во времена великой египетской императрицы Клеопатры, потайную комнату. В ней кроме бронзовой статуи императрицы обнаружилась поразительной красоты диадема, ранее считавшаяся бесследно утерянной! Ученые сообщили, что диадема увенчана алым а-каратным рубином в форме змеиной головы. Однако буквально пару часов назад поступила новость, что бесценное украшение было изувечено: кто-то пробрался в камеру хранения диадемы и вырезал из нее рубин! Полиция устанавливает круг подозреваемых...*
- Услышав о краже рубина, Аня сразу бросилась исследовать информацию на черных рынках. В течение дня она обнаружила *Нобъявлений* о продаже рубина в форме змеиной головы, утверждающих что это именно украденная древняя ценность. Аня не простит себе, если она упустит такой бесценный экспонат для своей коллекции, поэтому она приказала своему помощнику Глебу срочно купить все эти камни, надеясь приобрести среди них настоящую реликвию.

Купив все N камней, Глеб тут же провел несколько пробных измерений, взвесив некоторые наборы из них, и отправил результаты Ане по электронной почте. Тем временем она проконсультировалась с известным исследователем старины Андрэ Шесто-Мерта по поводу украденной драгоценности и узнала, что по всем имеющимся историческим источникам рубин весил не a карат, как утверждали журналисты, а b карат! Зная результаты взвешиваний Глеба, и учитывая, что все поддельные камни весят a карат, и только настоящий змеиный рубин может весить b карат, определите, какие из купленных камней могут на самом деле являться потерянной реликвией великой императрицы прошлого.

Входные данные

В первой строке находятся четыре целых числа N , a , b и K ($1 \leq N \leq 10$, $1 \leq a, b \leq 1000$, $a \neq b$, $1 \leq K \leq 10$).

Далее идут K строк, описывающих взвешивания, проведенные Глебом.

Первое число в i -ом описании — w_i ($1 \leq w_i \leq 10000$), суммарный вес группы камней, участвовавших в i -ом взвешивании.

Второе число — m_i ($1 \leq m_i \leq N$) — количество камней, участвовавших в i -ом взвешивании.

Далее следуют m_i целых чисел, упорядоченных по возрастанию, — номера камней, участвовавших в i -ом взвешивании.

Выходные данные

Если среди купленных Глебом камней змеиного рубина точно нет, выведите строку "Fail" (без кавычек).

Если украденный рубин может присутствовать среди камней, то выведите в первой строке количество всех возможных кандидатур на роль древней реликвии, а второй строке — номера возможных вариантов в порядке возрастания.

Если же Глеб в некоторый момент ошибся в расчетах, и присланная им информация о взвешиваниях не может соответствовать действительности, выведите строку "Impossible" (без кавычек).

Ввод	Вывод
4 15 17 2 30 2 1 3 47 3 2 3 4	2 2 4
3 15 17 3 30 2 1 2 30 2 2 3 47 3 1 2 3	Impossible
2 1 2 2 1 1 2 1 1 1	Fail

- **Примечание**

- В первом тесте из первого взвешивания мы делаем вывод, что первый и третий камни гарантированно поддельные. С другой стороны, среди второго, третьего и четвертого камня точно есть настоящий. Значит, настоящим может оказаться второй или четвертый камень.
- Во втором тесте Глеб ошибся в измерениях, потому что из первых двух измерений следует, что все камни фальшивые, а из последнего — что настоящий камень, тем не менее, среди них присутствует.
- В третьем тесте из результатов явно следует, что оба приобретенных камня фальшивые.

```
var i,n,j,k,a,b,m,num,num1,count,count1,sum:longint;  
    flag,found,f:boolean;  
    p:array[0..201] of longint;  
    w:array [0..1001,0..201] of longint;
```

```
procedure proverka(num:longint;var fl:boolean);  
var j,sum:longint;  
begin  
    fl:=true;  
    for j:=1 to k do {переберем все взвешивания (столбцы)}  
    begin  
        sum:=w[j,0]*a; {масса кучки равна a*кол-во}  
        if w[j,num]=1 then sum:=sum+b-a;  
        if sum<>w[j,n+1] then fl:=false  
    end; {a, a, a, ..., a – нет рубина}  
end; {a, a, b, a, ..., a – есть один рубин}  
    {a, b, a, ..., b – есть несколько рубинов - некорректно}
```

```
begin
```

```
  read(n,a,b,k);
```

```
  for i:=1 to n do
```

```
    p[i]:=0;
```

```
  for i:=0 to k do
```

```
    for j:=0 to n+1 do
```

```
      w[i,j]:=0;
```

```
  flag:=true;      found:=false;      count:=0;
```

```
  for i:=1 to k do
```

```
    begin
```

```
      read(w[i,n+1],w[i,0]); {считаем суммарный вес и кол-во}
```

```
      if (w[i,n+1]<>w[i,0]*a) and (w[i,n+1]<>w[i,0]*a-a+b)
```

```
      then flag:=false; {если не все веса а или не один b,то ошибка}
```

```
      if w[i,n+1]=w[i,0]*a-a+b
```

```
      then found:=true; {если не все веса а и один b,то нашли}
```

```
      for j:=1 to w[i,0] do
```

```
        begin
```

```
          read(num); {отметим 1 все номера камней i-го взвешивания}
```

```
          w[i,num]:=1;
```

```
        end
```

```
  end;
```

```
a=15
```

```
b=10
```

	0	1	2	3	4	5	6	n+1
1	2	0	1	0	0	1	0	30
2	3	1	1	0	0	1	0	40
3	4	1	1	1	0	1	0	55

```
if flag {если не нашли противоречий}
then begin
  for i:=1 to n do {переберем все наборы взвешиваний}
  begin
    proverka(i,f); {если i-ая строка содержит рубин,}
    if f {то запомним ее номер}
    then begin inc(count);p[i]:=1 end;
  end;
  if found and (count=0)
  then writeln('Impossible')
  else if count=0
  then writeln('Fail')
  else begin
    writeln(count);
    for i:=1 to n do
      if p[i]=1 then write(i,' ')
    end
  end
end
else writeln('Impossible')
end.
```

Задача А. Панграмма (10 баллов)

- Слово или предложение на некотором языке называется *панграммой*, если в нем встречаются все символы алфавита этого языка *хотя бы один раз*. Панграммы часто используют в типографии для демонстрации шрифтов или тестирования средств вывода различных устройств.
- Вам дана строка, состоящая из маленьких и больших латинских букв. Проверьте, является ли эта строка панграммой. Считается, что строка содержит букву латинского алфавита, если эта буква встречается в верхнем или нижнем регистре.
- **Входные данные**
- В первой строке записано одно целое число n ($1 \leq n \leq 100$) — количество символов в строке.
- Во второй строке записана сама строка. Строка содержит исключительно строчные и заглавные латинские буквы.
- **Выходные данные**
- Выведите «Yes», если строка является панграммой, в противном случае выведите в нижнем регистре первую по алфавиту латинскую букву, которой нет во входной строке.

Ввод	Вывод
26 QwErTyUiOpLkJhGfDsAzXcVbNm	Yes
12 BestOlympiad	c
35 TheQuickBrownFoxJumpsOverTheLazyDog	Yes

```
var i,j,n:integer;      alp,s:string;      flag:boolean;

begin
  readln(n);          readln(s);
  flag:=true;
  alp:='abcdefghijklmnopqrstuvwxy';
  i:=0;
  repeat
    inc(i);
    if pos(alp[i],s)+pos(chr(ord(alp[i])-32),s)=0
    then flag:=false
until not flag or (i=26);

  if flag
  then writeln('Yes')
  else writeln(alp[i])
end.
```

Задача В. Множество (10 баллов)

- Множество натуральных чисел задается первыми двумя числами и правилом, порождающим все остальные элементы множества. Правило следующее: если взять любые (даже одинаковые) два элемента X и Y из множества, то элемент $X*Y+X+Y$ тоже принадлежит множеству. Других элементов, кроме начальных и включенных во множество при применении правила, во множестве нет. Например, если взять два первых элемента равными 1 каждый, то далее во множество войдут числа: $1*1+1+1=3$; $1*3+3+1=7$; $3*3+3+3=15$; $3*7+3+7=31$ и т.д.
- Требуется по первым двум числам множества A и B и третьему числу C определить, принадлежит ли число C множеству, порожденному числами A и B .
- **Входные данные**
- В единственной строке заданы через пробел три натуральных числа A, B и C . ($1 \leq A, B \leq 100$, $1 \leq C \leq 10^9$).
- **Выходные данные**
- Требуется вывести слово «Yes», если число C принадлежит множеству и «No», если не принадлежит.

Ввод	Вывод
1 1 1023	Yes
1 2 13	No

```
program Set_AB;  
var a,b,c,n:longint;  
begin  
    read(a,b,c);  
    c:=c+1;a:=a+1;b:=b+1;  
    while c mod a =0 do c:=c div a;  
    while c mod b =0 do c:=c div b;  
    if c=1 then writeln('Yes')  
        else writeln('No')  
end.
```

Задача С. Взаимные расстояния (10 баллов)

- На прямой расположены K различных точек, которые пронумерованы числами от 1 до K не обязательно в порядке их расположения на прямой. Назовем «крайними» две такие точки, что слева от одной из них и справа от другой из них нет ни одной из заданных точек. Для каждой пары точек измерили расстояние между ними. Требуется по этим данным определить для каждой точки расстояние до ближайшей из «крайних» точек.
- Входные данные
- В первой строке задано число точек K (от 2 до 8). В следующих $K*(K-1)/2$ строках записаны через пробел по 3 целых числа: номера каких-то двух точек i, j и результат измерения расстояния между ними d_{ij} ($1 \leq i, j \leq K; 1 \leq d_{ij} \leq 1000$). Гарантируется, что входные данные содержат ровно по одному разу все измеренные расстояния.
- Выходные данные.
- Выведите через пробел K чисел – расстояния от заданных точек до ближайшей «крайней» точки в порядке возрастания их номеров. Если

Ввод	Вывод
4 3 4 3 1 2 1 3 2 1 1 4 1 1 3 2 4 2 2	1 1 0 0 Примечание. Точки в примере расположены в порядке 3 2 1 4 (или наоборот) на расстоянии 1 друг от друга.

```
ar i,j,k,n,l,maxx,n1,n2:integer;
  x,ans:array [1..8] of integer;
  d:array [1..8,1..8] of integer;
  flag:boolean;
begin
  read(n); maxx:=0;
  for i:=1 to n do
    d[i,i]:=0;
  for i:=1 to (n*(n-1)) div 2 do
    begin
      read (j,k,l);
      d[j,k]:=1;    d[k,j]:=1;
      if d[j,k]>maxx
      then begin
          maxx:=d[j,k];
          n1:=j;    n2:=k
        end;
    end;
  end;
```

```
for i:=1 to n do
    x[i]:=d[n1,i];
flag:=true;
{check}
for i:=1 to n-1 do
    for j:=i+1 to n do
        if abs(x[i]-x[j])<>d[i,j]
            then flag:=false;
if flag
then for i:=1 to n do
    if abs(x[i]-x[n1])<abs(x[i]-x[n2])
        then write(abs(x[i]-x[n1]),' ')
        else write(abs(x[i]-x[n2]),' ')
    else writeln('Error')
end.
```

