Решение:

Идея решения сводится к полному перебору всех возможных расстановок букв в квадратной матрице 4 х 4. Для упрощения получения расстановок мы заменяем двумерный массив на одномерный. А именно, мы полагаем, что вся работа выполняется с массивом длинной в 16 элементов, элементы которого в процессе работы переводятся в двумерный.

Процедура получения очередной расстановки в свою очередь сводится к процедуре получения двоичного числа. Иными словами расстановка кодируется двоичным числом. Продемонстрируем это примером. Пусть дано множество из 3-х элементов (а, в, с) и дано некоторое двоичное число (трехзначное) например 010. Это число кодирует расстановку (в). Двоичное число 110 кодирует расстановку (а в). То есть если в двоичном числе в разряде с номером К стоит 1 то элемент множества с номером К в расстановке участвует иначе нет.

Из этого ясно, что механизм получения очередной расстановки сводится к получению очередного двоичного числа из уже имеющегося. А очередное число получается из текущего, прибавлением единицы по правилам двоичной арифметики.

Примечание

Однако для прибавления двоичной 1 можно воспользоваться более простым приемом:

- найдём первый младший нулевой элемент множества
- заменим его на единицу
- Все элементы стоящие до него заменим нулями.

Такой прием возможен если речь идет о прибавлении одной единицы. Поясним это утверждение примером:

1	0	1	1	1
				1
1	1	0	0	0

Действительно при прибавлении 1 к очередном разряду возникает ситуация переполнения, что приводит к записи нуля в очередном разряде результата, а 1 появляется только тогда, когда в очередном разряде слагаемого встречается ноль. В этом случае единица (которая называется «на ум пошло») переходит в разряд результата.

Таким образом мы воспользовались тем фактом, что между множеством двоичных чисел длины N и множеством расстановок из N элементов можно установить взаимно однозначное соответствие. Следовательно, задача поиска расстановок, полностью эквивалентна задаче построения двоичных чисел соответствующей длины. Поэтому вычислительный процесс состоит из двух компонентов:

1) Выполняется расчет очередного двоичного числа , которое собственно и представляет собой расстановку.
2) Выясняется, соответствует ли полученная расстановка критериям задачи.

Процесс завершается по получении последней расстановки которая в нашей терминологии соответствует наибольшему двоичному числу

Алгоритм

Пока не достигнуто наибольшее двоичное число **Выполнить**
Рассчитать очередное двоичное число
Если в числе 4 единицы, **То**

- Переводим одномерный массив в двумерный
- Выполняем проверку на соответствие очередной расстановки условиям задачи
- **Если** условия задачи выполнены **То**
  - Распечатываем очередную расстановку

Примечание

Для проверки количеств единиц, в программе созданы счетчики для каждой строки, каждого столбца и каждой большой диагонали. Условия задачи считаются выполненными, если значения всех счетчиков будут равны 1.

Текст программы на компонентном Паскале

```
MODULE Z1;
 IMPORT StdLog,In;
VAR
 a:ARRAY 100 OF INTEGER;
 b:ARRAY 100,100 OF INTEGER;
n:INTEGER;
PROCEDURE proverka1():BOOLEAN;
VAR
     f:BOOLEAN;
     i:INTEGER;
BEGIN
     f:=FALSE;
     FOR i:=0 TO n-1 DO
      IF a[i]=0 THEN f:=TRUE;
      END;
     END;
 RETURN f;
END proverka1;

PROCEDURE dvymeriz();
VAR
 i,k,j:INTEGER;
BEGIN
i:=0;
FOR k:=0 TO 3 DO
 FOR j:=0  TO 3 DO
   b[k,j]:=a[i];
    i:=i+1;
   END;
 END;
END dvymeriz;
```

```
PROCEDURE proverka2():BOOLEAN;
VAR
flag2:BOOLEAN;
q,i:INTEGER;
BEGIN
q:=0;
flag2:=FALSE;
 FOR i:=0 TO n-1 DO
  IF a[i]=1 THEN q:=q+1;
  END;
END;
IF q=4 THEN flag2:=TRUE;
END;
RETURN flag2;
END proverka2;

PROCEDURE work():BOOLEAN;
VAR
     flag:BOOLEAN;
     i,sum4,sum1,sum2,sum3,sum5,sum6,sum7,sum8,sum9,sum10,w:INTEG
ER;
BEGIN
     flag:=FALSE;
     sum1:=0;
     sum2:=0;
     sum3:=0;
     sum4:=0;
     sum5:=0;
     sum6:=0;
     sum7:=0;
     sum8:=0;
     sum9:=0;
     sum10:=0;
   FOR i:=0 TO 3 DO
       IF b[i,0]=1 THEN sum1:=sum1+1;
       END;
       IF b[i,1]=1 THEN sum2:=sum2+1;
       END;
       IF b[i,2]=1 THEN sum3:=sum3+1;
       END;
       IF b[i,3]=1 THEN sum4:=sum4+1;
       END;
       IF b[0,i]=1 THEN sum5:=sum5+1;
       END;
       IF b[1,i]=1 THEN sum6:=sum6+1;
       END;
       IF b[2,i]=1 THEN sum7:=sum7+1;
       END;
       IF b[3,i]=1 THEN sum8:=sum8+1;
       END;
       IF b[i,i]=1 THEN sum9:=sum9+1;
       END;
       IF b[i,3-i]=1 THEN sum10:=sum10+1;
```

```
       END;
 END;
  IF (sum1=1) & (sum2=1) & (sum3=1) & (sum4=1) & (sum5=1) &
(sum6=1) & (sum7=1) & (sum8=1) & (sum9=1) & (sum10=1) THEN
flag:=TRUE;
  END;
RETURN flag;
END work;


PROCEDURE raspechatka();
VAR
l,k:INTEGER;
BEGIN
FOR k:=0 TO 3 DO
 FOR l:=0  TO 3 DO
  StdLog.Int(b[k,l]);
StdLog.String("   ");
END;
  StdLog.Ln;
END;
  StdLog.Ln;
END raspechatka;

PROCEDURE main*;
     VAR
     i,h:INTEGER;
 BEGIN
     In.Open;
     In.Int(n);
     FOR i:=0 TO n-1 DO
       a[i]:=0;
     END;
   WHILE proverka1() DO
     i:=0;
     WHILE a[i]=1 DO i:=i+1;
     END;
     a[i]:=1;
     FOR h:=0 TO i-1 DO
     a[h]:=0;
     END;
  IF proverka2() THEN
   dvymeriz();
  IF work() THEN
   raspechatka;
  END;
  END;
END;
END main;
END Z1.
```

Текст программы на C++

```
#include <iostream.h>
#include <conio.h>
#include <stdlib.h>
int b[100][100],n,a[100];
int work();
int proverka1();
int proverka2();
void dvymeriz();
void raspechatka();
void main()
{    cin>>n;
          for (int i=0;i<n;i++) a[i]=0;
while  (proverka1())  //выполняется  расчет  очередного  двоичного
числа
{i=0;//
      while (a[i]==1)
      i++;
      a[i]=1;
      for (int h=0;h<=i-1;h++) a[h]=0;
 if (proverka2())
 {dvymeriz();
 if (work())
 {raspechatka();
 cout<<"\n";
 }
 }
}
}

int proverka1()
{
 for (int i=0;i<n;i++)
      if (a[i]==0) return 1;
return 0;
}

int proverka2() //проверяем колличество единиц в числе
{int q=0;
 for(int i=0;i<n;i++)
  if (a[i]==1) q++;
  if (q==4) return 1;
return 0;
}
void dvymeriz()
{int q=0;
 for (int i=0;i<=3;i++)
  for (int k=0;k<=3;k++)
  {
  b[i][k]=a[q];
   q++;

  }
}
```

```
int work() //Выяснение, соответствует ли полученная расстановка
критериям задачи
{int sum1=0,sum2=0,sum3=0,sum4=0,
     sum5=0,sum6=0,sum7=0,sum8=0,
     sum9=0,sum10=0;
 for (int i=0;i<=3;i++)
 {
  //проверка строк
  if (b[i][0]==1) sum1++;
  if (b[i][1]==1) sum2++;
  if (b[i][2]==1) sum3++;
  if (b[i][3]==1) sum4++;
  //проверка столбцов
  if (b[0][i]==1) sum5++;
  if (b[1][i]==1) sum6++;
  if (b[2][i]==1) sum7++;
  if (b[3][i]==1) sum8++;
  //проверка диагоналей
  if (b[i][i]==1) sum9++;
  if (b[i][3-i]==1) sum10++;
 }
  if (sum1==1 && sum2==1 &&
       sum3==1 && sum4==1 &&
       sum5==1 && sum6==1 &&
       sum7==1 && sum8==1 &&
      sum9==1 && sum10==1) return 1;
return 0;
}
void raspechatka() //распечатываем верные расстановки
{
 for (int i=0;i<=3;i++)
 {
  for (int k=0;k<=3;k++)
  {
  cout << b[i][k];
  }
  cout<<"\n";
 }
}
```