Программа, позволяющая заполнить и вывести на экран двухмерный массив A.

```
PROGRAM example1;
CONST N = 8;
M = 10;
VAR
a : ARRAY [ 1 .. N , 1 .. M ] of INTEGER;
i, j : INTEGER;
BEGIN
 FOR i := 1 TO N DO
      FOR j := 1 TO M DO
                a[ i, j ] := random(10) ;
WRITELN('Полученный массив:' );
FOR i := 1 TO N DO
   BEGIN
        FOR j := 1 TO M DO
            WRITE( a[ i,j ] ,'  ' );
WRITELN(");
END;
END.
```

Программа, позволяющая заполнить двухмерный массив А и вычислить сумму элементов данного массива.

```
PROGRAM example;
uses crt;
CONST N = 8;  M = 10;
 VAR  a : ARRAY [ 1 .. N , 1 .. M ] of INTEGER;
 i,j : INTEGER;
 sum: INTEGER;
BEGIN
Randomize;
sum := 0;
FOR i := 1 TO N DO
     FOR j := 1 TO M DO
A[i,j] := random(10) ;
WRITELN( 'Полученный массив: ‘ ) ;
FOR i := 1 TO N DO
          BEGIN
FORj := 1 TO M DO
               WRITE( a[i,j],' ' ) ;
 WRITELN( '' ) ;
END;
FOR i := 1 TO N DO
     FOR j := 1 TO M DO
              sum := sum+a[i,j];
WRITELN('Сумма элементов массива: ',sum);
END.
```

Алгоритм вычисления суммы элементов всего двумерного массива.

```
S:=0;
For i:=1 to m do
For j:=1 to n do
S:=S+a[i,j];
```

Вычисление суммы элементов каждой строки.

Нужно использовать базовый алгоритм вычисления суммы элементов одномерного массива (это будут строки) и повторить его столько раз, сколько имеется строк.

Совет: чтобы не ошибиться, **напишите сначала базовый алгоритм вычисления суммы для строки, а потом повторяйте его нужное количество раз**.

```
S:=0;
For j:=1 to 4 do
S:=S+a[индекс строки,j]
{ для того, чтобы
перемещаться по строке
мы меняем индекс j  }
```

Теперь повторим алгоритм 3 раза

```
for i:=1 to 3 do begin
S:=0;
for j:=1 to 4 do
S:=S+a[ i, j ];
writeln(S,i)
end;
```

Задача поиска максимального (минимального) элемента (считая, что элемент единственный) для каждой строки

Задача сводится к повторению аналогичного базового алгоритма для одномерных массивов m раз, если мы ищем максимальный элемент каждой строки.

Для i-той строки алгоритм выглядит следующим образом:

```
max:=a[номер_строки, 1];
For j:=1 to n do
if a[номер_строки,j]>max then
max:=a[номер_строки,j]
```

Повторим алгоритм m раз

```
for i:=1 to m do
begin
max:=a[i,1];
For j:=1 to n do
if a[i,j]>max then
max:=a[i,j]
 end;
```

Алгоритм поиска минимального элемента для каждого столбца:

```
for j:=1 to n do {перемещаемся по столбцу}
min:=a[1,j];
for i:=1 to m do {начинаем движение по i-тому столбцу}
   if a[i,j]<min then
         min:=a[i,j];
end;
  writeln('min=',j,' столбца ',min)
end;
```

Алгоритм для случая, когда надо сохранить индексы максимального элемента.

```
for i:=1 to m do
begin
max:=a[i,1];
ind_L:=i; {сохраняем номер строки}
ind_C:=1; {заносим номер 1—первый столбец}
for j:=1 to n do
if a[i,j]>max then
begin
max:=a[i,j];
ind_C:=j {сохраняем номер j-ого столбца} end;
writeln('max строки=', i, max) end;
```

Алгоритм поиска минимального элемента и его индексов для каждого столбца:

```
for j:=1 to n do {перемещаемся по столбцу}
begin
min:=a[1,j];
ind_L:=1; {сохраняем номер строки}
ind_C:=j; {сохраняем номер столбца}
for i:=1 to m do {начинаем движение по i-тому столбцу}
   if a[i,j]<min then
   begin
       min:=a[i,j];
       ind_L:=i {сохраняем номер i-той строки}
   end;
  writeln('min=',j,' столбца ',min)
end;
```

Алгоритм поиска минимального элемента и его индексов для всего массива.

```
min:=a[1,1];
ind_L:=1;
 ind_C:=1;
 for i:=1 to m do
    for j:=1 to n do
       if a[i,j]<min then
         begin
           min:=a[i,j];
           ind_L:=i;
           ind_C:=j
         end;
```