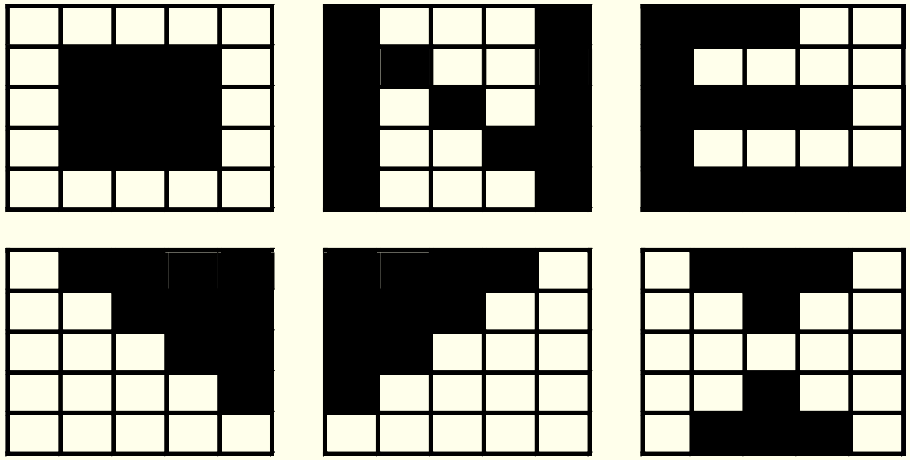
Exemplos de soluções para exercícios

Exercício 1

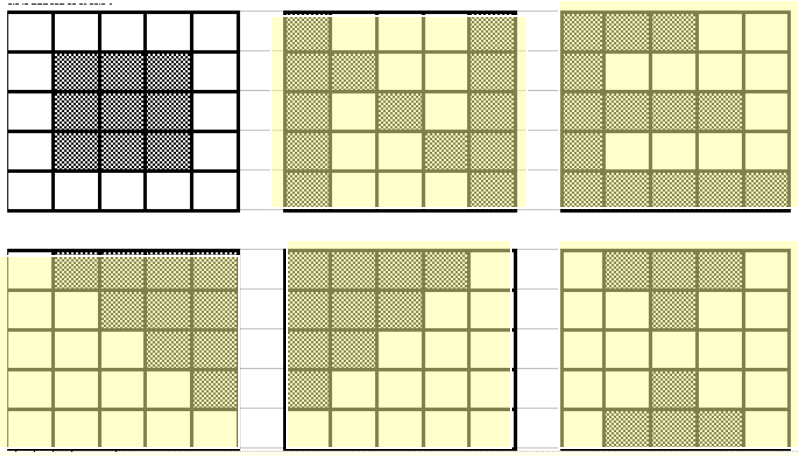
Preencher por leitura uma matriz M (5,5). Em seguida, calcular e imprimir a média dos elementos das áreas assinaladas.



Leitura e escrita da matriz m (5,5)

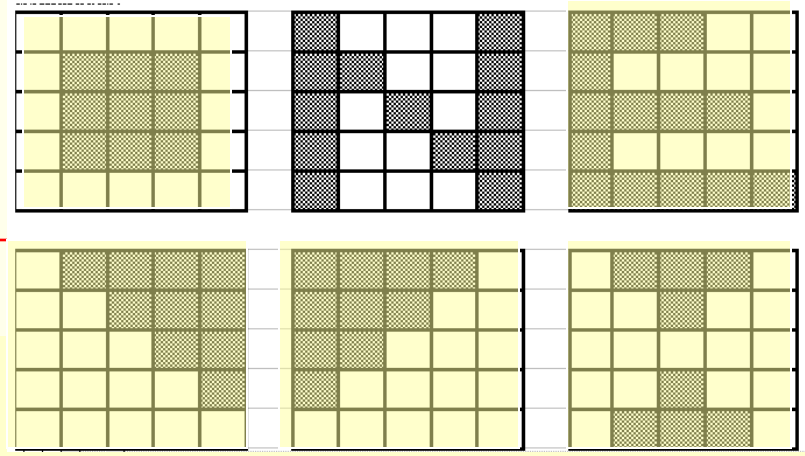
```
//Calcula media sobre elementos variados de
//uma matriz bidimensional
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#define  MAX  5
int main( )
{
  float m[MAX] [MAX];
  //somatorios para cada caso
  float soma1, soma2, soma3, soma4, soma5, soma6;
  //medias para cada caso
  float media1, media2, media3, media4, media5, media6;
  int  lin, col, contaelem, limite;
  //leitura dos valores da matriz m
  for (lin = 0; lin < MAX; lin++)
     for (col = 0; col < MAX; col++)
        {
          printf ("\nInforme o valor [%d] [%d]: " , lin, col );
          scanf ("%f", &m[lin][col]);
        }
  //escrita da matriz m
  printf("\n\nMatriz lida: \n");
  for (lin = 0; lin < MAX; lin++)
      {
        for (col = 0; col < MAX; col++)
             printf ("%6.2f    ",m[lin][col] );
        printf("\n");
      }

...
```

Calcular e imprimir a média dos elementos da Figura 1:



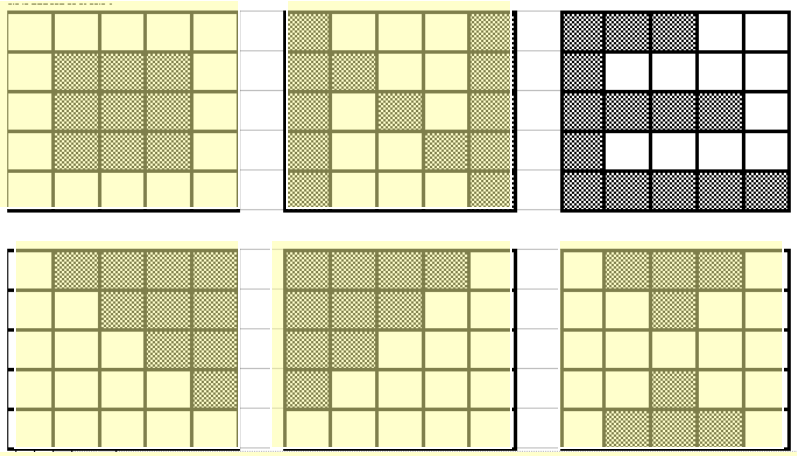
```
//primeira figura
soma1 = 0; contaelem  = 0;
for (lin = 1; lin < MAX - 1; lin++)
   for (col = 1; col < MAX -1; col++)
       {
          soma1 = soma1 + m[lin][col];
          contaelem++;
       }
media1 = soma1 / contaelem;
printf("\nMedia da figura 1: %6.2f (%d elementos considerados) \n",
                  media1, contaelem );
...
```

Calcular e imprimir a média dos elementos da Figura 2:



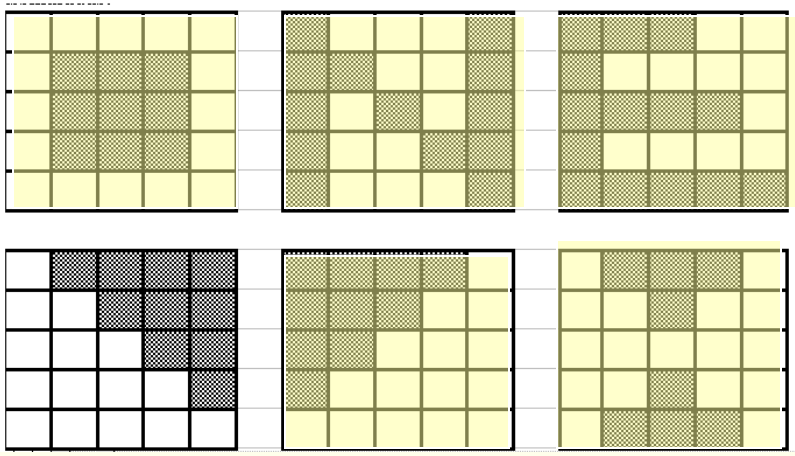
```
//segunda figura
 soma2 = 0; contaelem = 0;
 for (lin = 0; lin < MAX; lin++)
   switch (lin)
     {
         case 0: //linha 0 e MAX - 1
         case MAX - 1:   soma2 = soma2 + m[lin][0] + m[lin][MAX - 1];
                         contaelem = contaelem + 2;
                         break;
         case 1: // linhas 1, 2 e 3
         case 2:
         case 3: soma2 = soma2 + m[lin][0] + m[lin][MAX - 1] + m[lin][lin];
                  contaelem = contaelem + 3;
                 break;
     }
  media2 = soma2 / contaelem;
  printf("\nMedia da figura 2: %6.2f (%d elementos considerados) \n",
                      media2, contaelem );
...
```

Calcular e imprimir a média dos elementos da Figura 3:



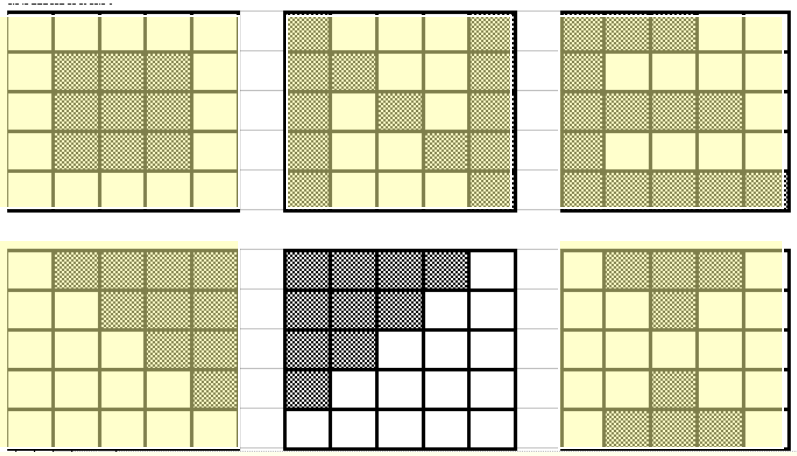
```
//terceira figura
soma3 = 0; contaelem = 0;
limite = MAX - 2;
for (lin  = 0; lin < MAX; lin++)
  switch (lin)
   {
       case 0: //linhas 0, 2 e MAX - 1
       case 2:
       case MAX - 1: for (col = 0;  col < limite ; col++ )
                                {
                                 soma3 = soma3 + m[lin][col];
                                 contaelem++;
                                }
                     limite++;          //limite da coluna varia de 2 a 4
                     break;
       case 1: //linhas 1 e 3
       case 3: soma3 = soma3 + m[lin][0];
               contaelem++;
               break;
   }
media3 = soma3 / contaelem;
printf("\nMedia da figura 3: %6.2f (%d elementos considerados) \n",
          media3, contaelem );
...
```

Calcular e imprimir a média dos elementos da Figura 4:



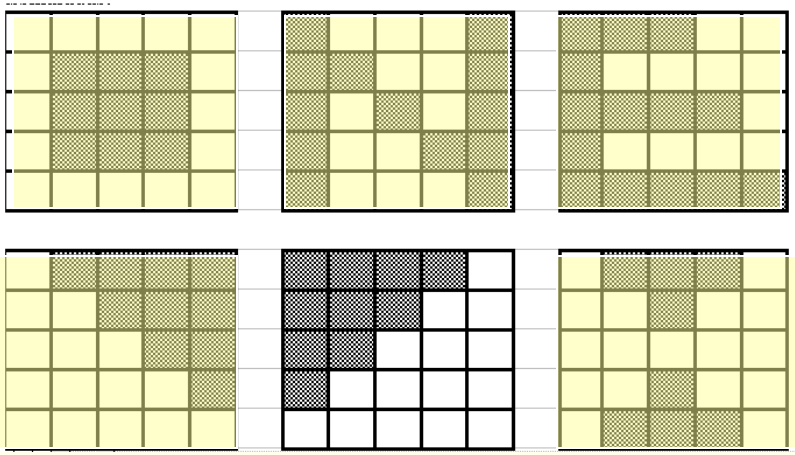
```
//quarta figura
soma4 = 0; contaelem = 0;
for (lin = 0; lin < MAX; lin++)
   for (col = lin + 1; col < MAX; col++)
   //valor para a coluna varia de linha + 1 até MAX - 1
       {
         soma4 = soma4 + m[lin][col];
         contaelem++;
       }
media4 = soma4 / contaelem;
printf("\nMedia da figura 4: %6.2f (%d elementos considerados) \n",
          media4, contaelem );
```

Calcular e imprimir a média dos elementos da Figura 5: versão 1



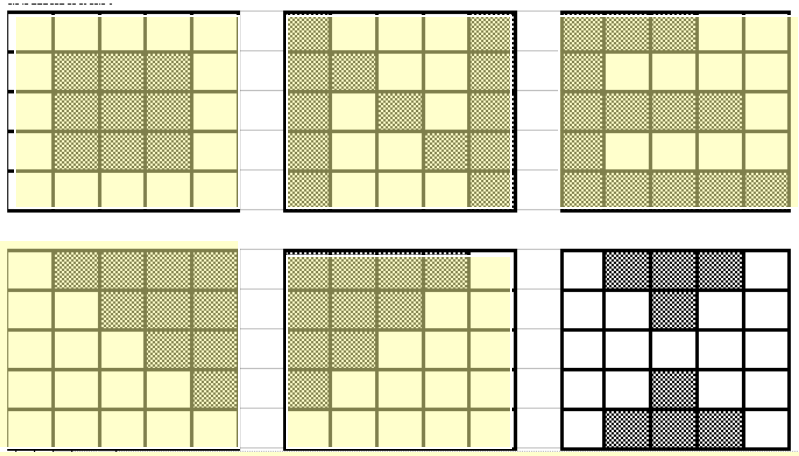
```
//quinta figura - versao 1
soma5 = 0; contaelem = 0;

for (lin = 0 ; lin < MAX - 1 ; lin++)
  for (col = 0; col < (MAX - 1)- lin; col++) //limite para a coluna: de 3 a 0
        {
          soma5 = soma5 + m[lin] [col];
          contaelem++;
         }
media5 = soma5 / contaelem;
printf
 ("\nMedia da figura 5: %6.2f (%d elementos considerados) - versao 1 \n",
       media5, contaelem );
...
```

Calcular e imprimir a média dos elementos da Figura 5: versão 2



```
//quinta figura - versao 2
soma5 = 0; contaelem = 0;
limite = MAX - 1;
for (lin = 0 ; lin < MAX - 1 ; lin++)
    {
     for (col = 0; col < limite; col++)
       {
          soma5 = soma5 + m[lin] [col];
                contaelem++;
        }
     limite--;     //limite para a coluna varia de MAX - 1  a 0,
                   // dependendo da linha
    }
media5 = soma5 / contaelem;
printf("\nMedia da figura 5: %6.2f (%d elementos considerados) - versao 2 \n",
          media5, contaelem );
...
```

Calcular e imprimir a média dos elementos da Figura 6:



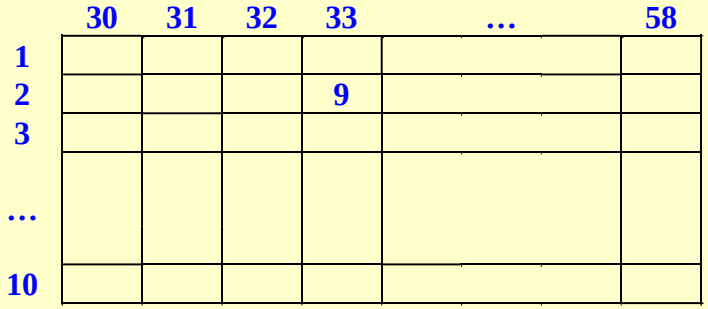
```
//sexta figura
soma6 = 0; contaelem = 0;
for (col = 1; col < MAX - 1; col++)
    switch (col)
      {
       case 2:  for (lin = 0; lin < MAX; lin++)
                //na coluna 3, acessa as linhas 0, 1, 3 e 4
                if (lin != 2)       //ignora linha 2
                  {
                    soma6 = soma6 + m[lin] [col];            //coluna 2
                    contaelem++;
                   }
                break;
       case  1:
       case  MAX - 2: //nas colunas 1 e MAX - 2, acessa as linhas 0 e MAX - 1
                      soma6 = soma6 + m[0][col] + m[MAX - 1][col] ;
                      contaelem = contaelem + 2;
                      break;

      }
 media6 = soma6 / contaelem;
 printf("\nMedia da figura 6: %6.2f (%d elementos considerados) \n",
           media6, contaelem );
system("pause");
return 0;
}
```

```
Matriz lida:
  1.00      2.00      3.00      4.00      5.00
  6.00      7.00      8.00      9.00     10.00
 11.00     12.00     13.00     14.00     15.00
 16.00     17.00     18.00     19.00     20.00
 21.00     22.00     23.00     24.00     25.00

Soma 1 = 117.00

Media da figura 1:  13.00 (9 elementos considerados)

Soma 2 = 169.00

Media da figura 2:  13.00 (13 elementos considerados)

Soma 3 = 193.00

Media da figura 3:  13.79 (14 elementos considerados)

Soma 4 =  90.00

Media da figura 4:   9.00 (10 elementos considerados)

Soma 5 =  70.00

Media da figura 5:   7.00 (10 elementos considerados) - versao 1

Soma 6 = 104.00

Media da figura 6:  13.00 (8 elementos considerados)
Pressione qualquer tecla para continuar. . .
```

```
Matriz lida:
  1.00      1.50      2.00      2.50      3.00
  3.50      4.00      4.50      5.00      5.50
  6.00      6.50      7.00      7.50      8.00
 10.00     10.50     11.00     11.50     12.00
 12.50     13.00     13.50     14.00     14.50

Soma 1 =  67.50

Media da figura 1:   7.50 (9 elementos considerados)

Soma 2 =  98.50

Media da figura 2:   7.58 (13 elementos considerados)

Soma 3 = 112.50

Media da figura 3:   8.04 (14 elementos considerados)

Soma 4 =  51.50

Media da figura 4:   5.15 (10 elementos considerados)

Soma 5 =  41.50

Media da figura 5:   4.15 (10 elementos considerados) - versao 1

Soma 6 =  62.00

Media da figura 6:   7.75 (8 elementos considerados)
Pressione qualquer tecla para continuar. . . _
```

[Ciência dos Computadores - Tremblay Bunt] Uma análise de acidentes de trânsito está sendo executada na ilha de Manhattan, cidade de New York. Por conveniência as ruas e avenidas são representadas por uma matriz como segue:

Exercício 2



onde as linhas indicam as avenidas da Primeira Avenida até a Décima Avenida, e as colunas indicam as ruas da Rua 30 à Rua 58. Os elementos da matriz indicam o número de acidentes ocorridos nas proximidades no período pré-fixado. Por exemplo, no caso mostrado, nove acidentes ocorreram próximos à intersecção da Segunda Avenida com a Rua 33. Um número desconhecido de dados de acidentes é lido. Cada acidente é dado por um par de números que descrevem sua localização. Por exemplo, o par 7, 42 designa um acidente ocorrido nas vizinhaças da Sétima Avenida com a Rua 42. Formular um algoritmo para ler estas informações e preparar uma matriz da forma mostrada. Utilizar um teste de fim de arquivo para determinar o fim dos dados. Incorporar um teste para verificar se os dados não caem antes da Primeira Avenida ou após a Décima Avenida, nem antes da Rua 30 ou após a Rua 58. Rejeitar qualquer dado inválido com uma mensagem. Após ler e armazenar os dados, produzir uma listagem das dez intersecções mais perigosas.

Inicialmente é apresentado o código inicial, onde estão a leitura e escrita da matriz original.

A seguir são apresentadas duas soluções para a determinação das intersecções mais perigosas

```
//Parte inicial da solucao do problema dos acidentes em Nova York.
//Leitura e escrita da matriz.
//O usuario fornece avenida entre 1 e 10 e rua entre 30 e 58.
//Internamente sao processadas avenidas entre 0 e 9 e ruas entre 0 e 29
#include <stdio.h>
#include <stdlib.h>
#define MAXAVENIDAS 10
#define MINRUAS 30
#define MAXRUAS 59
#define INTERSECPERIGO 10
int main (  )
{
   int rua, avenida, i, j, k, maior, ave , ru, intersec, seguir, cont_apres = 0;
   int matriz[MAXAVENIDAS] [MAXRUAS];
   for (i = 0; i < MAXAVENIDAS; i++)
     for (j= 0; j < MAXRUAS; j++)
        matriz[i] [j] = 0;
   //leitura da matriz
   do
   {
     printf("Avenida (1 a 10, -1 para parar): ");
     scanf("%d", &avenida);
     if ((avenida < 1 || avenida > MAXAVENIDAS) && avenida != -1)
       printf("\nAvenida invalida, deve ser 1 a 10 ou -1 para parar!\n");
   }
   while ((avenida < 1 || avenida > MAXAVENIDAS) && avenida != -1);
   ...
```

```
while (avenida != -1)
  {
    do
    {
      printf("Rua (30 a 58): ");
      scanf("%d", &rua);
      if (rua < MINRUAS || rua > MAXRUAS - 1)
         printf("\nRua deve ser de %d a %d\n", MINRUAS, MAXRUAS - 1);

    }
    while (rua < MINRUAS || rua > MAXRUAS - 1);
    do
    {
      printf("Numero acidentes do cruzamento avenida %d com rua %d: \n", avenida, rua);
      scanf("%d", &matriz[avenida - 1] [rua - 30]);
      if (matriz[avenida - 1] [rua - 30] < 1)
         printf("\nNumero de acidentes deve ser maior que zero!\n");
     }
    while (matriz[avenida - 1] [rua - 30] < 1);
    do
    {
      printf("Avenida (1 a 10, -1 para parar): ");
      scanf("%d", &avenida);
      if ((avenida < 1 || avenida > MAXAVENIDAS) && avenida != -1)
         printf("\nAvenida invalida, deve ser 1 a 10 ou -1 para parar!\n");
    }
    while ((avenida < 1 || avenida > MAXAVENIDAS) && avenida != -1);
  }

  ...
```

```
//escrita da matriz
for (i = 0; i < MAXAVENIDAS; i++)
     {
       printf("\n");
       for (j= 0; j < MAXRUAS - 30; j++)
         printf("\%2d ", matriz[i] [j]);
     }
   printf("\n\n");

  printf("\n\n");
  system("pause");
  return 0;
}
```

Solução 1

problema dos acidentes em Nova York, trecho que determina as intersecções mais perigosas

A matriz original é percorrida sucessivas vezes.

A cada varredura da matriz determina-se uma intersecção mais perigosa, seu valor é apresentado e em seguida substituído na matriz por -1 (para que não seja mais considerado nas varreduras subseqüentes).

O processo encerra-se ou pela apresentação de todas as intersecções mais perigosas solicitadas, ou por terminarem as intersecções com acidentes.

Se nenhuma intersecção com acidentes for informada, uma mensagem é apresentada.

```
intersec = 0;
 do // corresponde ao while (intersec < INTERSECPERIGO && maior != -99);
  {
   intersec ++;
   maior = -99;
   i = 0;
   j = 0;
   seguir = 1;
   do // corresponde ao while (seguir)
   {
      if (matriz[i] [j] > 0)
         {
          maior = matriz[i] [j];
          ave = i;
          ru = j;
          seguir = 0;
         }
      else
         {
          if (i <MAXAVENIDAS)
             {
               if (j < MINRUAS - 1)
                   j++;
               else
                  {
                   i++;
                   if (i < MAXAVENIDAS)
                       j = 0;
                   else
                       seguir = 0;
                  }
             }
          else
               seguir = 0;
         }
   }
   while (seguir); // fim do trecho de determinacao de um maior
```

Neste ponto é determinado o maior para uma nova varredura da matriz.
Só são considerados valores maiores que 0, uma vez que zero significa que
não houve acidentes no cruzamento e -1 é o valor atribuído a uma posição de intersecção mais perigosa já apresentada.
maior = -99 significa que não há mais intersecções na matriz com valores acima de zero, ou seja todas as intersecções com acidentes já foram consideradas.

Solução 1
problema dos acidentes em Nova York,
trecho que determina
as intersecções mais perigosas

Cont.

Solução 1
problema dos acidentes em Nova York,
trecho que determina
as intersecções mais perigosas

```
if (maior != -99)
      {
       cont_apres++;
       for (i = 0; i < MAXAVENIDAS; i++)
           for (j= 0; j < MAXRUAS - 30; j++)
               if (maior < matriz[i] [j])
                  {
                   ave = i;
                   ru = j ;
                   maior = matriz[i] [j];
                  }
       printf
("\nInterseccao mais perigosa de numero %d na avenida %d rua %d com acidentes
%d\n",
               intersec, ave + 1 , ru + 30, maior );
       matriz[ave] [ru] = -1;
      }
  } while (intersec < INTERSECPERIGO && maior != -99);
  //o trecho a seguir so eh ativado se nenhum acidente foi informado
  //para nenhum cruzamento, ou se houve um numero de interseccoes com
  //acidentes abaixo do numero que se procurava apresentar.
  if (cont_apres > 0)
    {
     if (maior == -99)
         printf("\nTodas as interseccoes com acidentes foram apresentadas!");
     }
 else
```

Solução 2

problema dos acidentes em Nova York, trecho que determina as intersecções mais perigosas

A partir da matriz original é gerada uma matriz condensada apenas com as interseçções não nulas.

A matriz condensada é classificada e após, a partir dela, são apresentadas as intersecções mais perigosas.

Se nenhuma intersecção com acidentes for informada, uma mensagem é apresentada.

Solução 2
problema dos acidentes em Nova York,
trecho que determina
as intersecções mais perigosas

```
  linha = 0;
   for (i = 0; i < MAXAVENIDAS; i++)
     {
       for (j= 0; j < MAXRUAS - 30; j++)
         if ( matriz[i] [j] > 0)
            {
              cont_com_acidentes++;
              matriz_condensada[linha] [0] = matriz[i] [j];
              matriz_condensada[linha] [1] = i+ 1;
              matriz_condensada[linha] [2] = j + 30;
              linha++;
            }
     }
   printf("\n\n");
   for (i = 0; i < linha; i++)
       printf("\n%6d %6d %6d", matriz_condensada[i] [0], matriz_condensada[i] [1],
matriz_condensada[i] [2]);
```

Cont.

Solução 2
problema dos acidentes em Nova York,
trecho que determina
as intersecções mais perigosas

```
trocou = 1;
   m = linha - 1;
   k = 1;
   while (trocou)
   {
     trocou = 0;
     for (i = 0 ; i < m; i++)
        if (matriz_condensada[i][0] > matriz_condensada[i + 1] [0])
           {
             aux1 =  matriz_condensada[i][0];
             aux2 =  matriz_condensada[i][1];
             aux3 =  matriz_condensada[i][2];
             matriz_condensada[i][0] = matriz_condensada[i + 1] [0];
             matriz_condensada[i][1] = matriz_condensada[i + 1] [1];
             matriz_condensada[i][2] = matriz_condensada[i + 1] [2];
             matriz_condensada[i + 1][0] = aux1;
             matriz_condensada[i + 1][1] = aux2;
             matriz_condensada[i + 1][2] = aux3;
             k = i;
             trocou = 1;
           }
     m = k;
   }
```

Cont.

Solução 2
problema dos acidentes em Nova York,
trecho que determina
as intersecções mais perigosas

```
if (cont_com_acidentes)
      {
        for (i = 0; i < linha && i <= INTERSECPERIGO; i++)
       printf("\nInterseccao mais perigosa de numero %d na avenida %d rua %d com acidentes %d\n",
              i + 1, matriz_condensada[i] [1] , matriz_condensada[i] [2], matriz_condensada[i] [0]);
        if (linha  < INTERSECPERIGO)
            printf("\nNao ha mais interseccoes com acidentes a serem apresentadas!");
      }
  else
       printf("\nNenhuma interseccao com acidentes foi informada!");
```

Exercício 3

Dada uma matriz M (10, 20), preenchê-la por leitura, escrevê-la e apresentar:

- o maior elemento de cada linha da matriz;
- a média dos elementos de cada coluna;
- para cada valor lido, até que o usuário informe que deseja parar, apresentar, em ordem decrescente, todos os valores (acompanhados de sua posição (linha-coluna) na matriz original) que forem maiores que o valor lido considerado no momento. Se em algum caso não houver valores na matriz que preencham esta condição, dar apenas uma mensagem a respeito.

Solução para a terceira tarefa

Tarefa 3: n valores são lidos e para cada um são apresentados, em ordem decrescente, os valores presentes na matriz de entrada que são maiores que o valor.

Solução adotada: criação de uma matriz de apoio, com os valores maiores que o pesquisado, mais suas coordenadas de linha e coluna. Essa matriz de apoio é classificada com um *bubblesort*.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#define  MAX1 2
#define  MAX2 4
int main( )
{
  float m[MAX1] [MAX2], pesq, aux;
  float somaiores[MAX1 * MAX2] [3];
  int lin, col, ind, i, lim, trocou, k;
  float soma, media, maior;
  system("color f1");
  //leitura dos valores da matriz m
  for (lin = 0; lin < MAX1; lin++)
     for (col = 0; col < MAX2; col++)
        {
          printf ("\nInforme o valor [%d] [%d]: " , lin, col );
          scanf ("%f", &m[lin][col]);
        }
```

Nos testes as dimensões da matriz estão reduzidas

```
//escrita da matriz m
printf("\n\nMatriz lida: \n");
for (lin = 0; lin < MAX1; lin++)
    {
      for (col = 0; col < MAX2; col++)
           printf ("%6.2f    ",m[lin][col] );
      printf("\n");
    }
//maior elemento de cada linha
for (lin = 0; lin < MAX1; lin++)
    {
      maior = m[lin] [0];
      for (col = 0; col < MAX2; col++)
          if (m[lin] [col] > maior)
             maior = m[lin] [col];
      printf("\nMaior elemento da linha %d = %6.2f\n", lin, maior);
      printf("\n");
    }
```

```
//media dos elementos de cada coluna
for (col = 0; col < MAX2; col++)
    {
      soma = 0;
      for (lin = 0; lin < MAX1; lin++)
          soma += m[lin] [col];
      media = soma / (MAX1);
      printf("\nMedia dos elementos da coluna %d = %6.2f\n", col, media);
      printf("\n");
    }
//para cada valor informado, lista em ordem
//decrescente os valores da matriz maiores
//que o valor informado
printf("\nValor a pesquisar (0 para parar): ");
scanf("%f", &pesq);
```

```
while (pesq)
   { // inicio do while (pesq)
     //criacao da matriz de apoio para a classificacao dos valores
     //existentes na matriz que sao maiores que o valor pesquisado
     ind = -1;
     for (lin = 0; lin < MAX1; lin++)
       for (col = 0; col < MAX2; col++)
          if (m[lin] [col] > pesq)
            {
                ind++;
                somaiores[ind] [0] = m[lin][col];
                somaiores[ind] [1] = lin;
                somaiores[ind] [2] = col;
            }
```

```
if (ind < 0)
      printf("\nNao ha valor maior que %6.2f no arranjo\n", pesq);
  else
      if (ind == 0)
         printf("\nvalor %6.2f na linha %6.0f na coluna %6.0f\n",
              somaiores[0][0], somaiores[0] [1], somaiores[0] [2]);
      else
       { //ha mais de um valor superior ao pesquisado, entao classifica e
         //apresenta os valores
```

Rotina de classificação:

bubblesort

```
trocou = 1; lim = ind; k = 0;
while (trocou)
 {
   trocou = 0;
   for (i = 0; i < lim; i++)
     if (somaiores[i][0] < somaiores[i + 1][0])
        {
          aux = somaiores[i][0];
          somaiores[i] [0]= somaiores[i + 1] [0];
          somaiores[i + 1][0] = aux;
          aux = somaiores[i][1];
          somaiores[i] [1]= somaiores[i + 1] [1];
          somaiores[i + 1][1] = aux;
          aux = somaiores[i][2];
          somaiores[i] [2]= somaiores[i + 1] [2];
          somaiores[i + 1][2] = aux;
          k = i;
          trocou = 1;
        }
   lim = k;
 }
for (lin = 0; lin <= ind; lin++)
  printf("\nvalor %6.2f na linha %6.0f na coluna %6.0f\n",
     somaiores[lin][0], somaiores[lin] [1], somaiores[lin] [2]);
}
```

```
    printf("\nValor a pesquisar (0 para parar): ");
    scanf("%f", &pesq);
  } //fim do while(pesq)
  system("pause");
  return 0;
}
```