**Universidade Federal de Ouro Preto – UFOP**
**Instituto de Ciências Exatas e Biológicas – ICEB**
**Departamento de Computação – DECOM**
**Disciplina: Algoritmos e Estrutura de Dados I – CIC102**
**Professor: David Menotti (menottid@gmail.com)**

## Lista de Exercícios 05 – Estruturas de Dados Homogêneas - Matrizes

1) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva os elementos da diagonal principal.

```
algoritmo L5P01;
constante
  N = 10
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia (M[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      se ( I = J ) então
        imprima(M[I,J]);
      fim-se
    fim-para
  fim-para
fim
```

```
program L5P01;
const
  N = 10;
var
  I, J : integer;
  M : array[1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Entre com o elemento (',I,'x',J,') da matriz: ');
      readLn(M[I,J]);
    end;
  for I := 1 to N do
  begin
    for J := 1 to N do
      if ( I = J ) then
        write(M[I,J],chr(9))
      else
        write(chr(9));
    writeLn('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    if ( I == J )
      fprintf(1,'%d\t',M(I,J));
    else
      fprintf(1,'\t');
    end
  end
  fprintf(1,'\n');
end
```

2) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva todos os elementos, exceto os elementos da diagonal principal.

```
algoritmo L5P02;
constante
  N = 10;
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      se ( I <> J ) então
        imprima(M[I,J]);
      fim-se
    fim-para
  fim-para
fim
```

```
program L5P02;
const
  N = 10;
var
  MAT : array [1..N,1..N] of integer;
  I, J : integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(MAT[I,J]);
    end;
  writeLn('Elementos da diagonal principal: ');
  for I := 1 to N do
  begin
    for J := 1 to N do
      if ( I <> J ) then
        write(MAT[I,J],chr(9))
      else
        write(chr(9));
    writeLn('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    MAT(I,J) = input(sprintf('Digite o elemento (%dx%d ) da matriz: ',I,J));
  end
end
disp('Matriz sem elementos da diagonal principal: ');
for I = 1 : N
  for J = 1 : N
    if ( I ~= J )
      fprintf(1,'%d\t ',MAT(I,J));
    else
      fprintf(1,'\t');
    end
  end
  fprintf(1,'\n');
end
```

3) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva somente os elementos acima da diagonal principal.

```
algoritmo L5P03;
constante
  N = 10;
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até 10 faça
    para J de 1 até 10 faça
      se ( I < J ) então
        imprima(M[I,J]);
      fim-se
    fim-para
  fim-para
fim
```

```
program L5P03;
const
  N = 10;
var
  I, J : integer;
  M : array [1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento(',I,'x',J,') da matriz: ');
      readLn(M[I,J]);
    end;
  writeLn('Elementos acima da diagonal principal:');
  for I := 1 to N do
  begin
    for J := 1 to N do
      if ( I < J ) then
        write(M[I,J],chr(9))
      else
        write(chr(9));
    writeLn('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
disp('Elementos acima da diagonal principal:');
for I = 1 : N
  for J = 1 : N
    if ( I < J )
      fprintf(1,'%d\t',M(I,J));
    else
      fprintf(1,'\t');
    end
  end
  fprintf(1,'\n');
end
```

4) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e imprima a soma dos elementos que estão acima da diagonal principal:

```
algoritmo L5P04;
constante
  N = 10;
var
  inteiro   : I, J, S, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  S <- 0;
  para I de 1 até N faça
    para J de 1 até N faça
      se ( I < J ) então
        S <- S + M[I,J];
      fim-se
    fim-para
  fim-para
  imprima(S);
fim
```

```
program L5P04;
const
  N = 10;
var
  I, J, S : integer;
  M : array[1..N,1..N] of integer;
begin
  for I:=1 to N do
    for J:=1 to N do
    begin
      writeln('Digite o elemento (',I,',',J,') da matriz: ');
      readln(M[I,J]);
    end;
  S:=0;
  for I := 1 to N do
    for J := 1 to N do
      if ( I < J ) then
        S := S + M[I,J];
  writeLn('a soma eh:',S);
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
S = 0;
for I = 1 : N
  for J = 1 : N
    if ( I < J )
      S = S + M(I,J);
    end
  end
end
fprintf(1,'A soma eh: %d\n',S);
```

5) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva somente os elementos abaixo da diagonal principal.

```
algoritmo L5P05;
constante
  N = 10;
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 2 até N faça
    para J de 1 até I-1 faça
      imprima(M[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P05;
const
  N = 10;
var
  I, J : integer;
  M : array [1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite Elemento (',I,'x',J,') da matriz: ');
      readLn(M[I,J]);
    end;
  writeLn('Elementos abaixo da Diagonal Principal');
  for I := 2 to N do
  begin
    for J := 1 to I-1 do
      write(M[I,J],chr(9));
    writeLn('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
disp('Elementos abaixo da Diagonal Principal');
for I = 2 : N
  for J = 1 : I-1
    fprintf(1,'%d\t',M(I,J));
  end
  fprintf(1,'\n');
end
```

6) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e imprima o produto dos elementos que estão abaixo da diagonal principal.

```
algoritmo L5P06;
constante
  N = 10;
var
  inteiro   : I, J, P, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  P <- 1;
  para I de 1 até N faça
    para J de 1 até N faça
      se ( I > J ) então
        P <- P * M[I,J];
      fim-se
    fim-para
  fim-para
  imprima(P);
fim
```

```
program L5P06;
const
  N = 10;
var
  I, J, P : integer;
  M : array[1..N,1..N] of integer;
begin
  for I:=1 to N do
    for J:= 1 to N do
    begin
      write('Entre com o elemento ',I,'x',J);
      readLn(M[I,J]);
    end;
  P := 1;
  for I := 1 to N do
    for J := 1 to N do
      if ( I > J ) then
        P := P * M[I,J];
  writeLn(P);
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemnto (%dx%d): ',I,J));
  end
end
P = 1;
for I = 1 : N
  for J = 1 : N
    if ( I > J )
      P = P * M(I,J);
    end
  end
end
disp(P);
```

7) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva os elementos da diagonal secundária.

```
algoritmo L5P07;
constante
  N = 10;
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      se ( J = (N - I + 1) ) então
        imprima(M[I,J]);
      fim-se
    fim-para
  fim-para
fim
```

```
program L5P07;
const
  N = 10;
var
  M : array [1..N,1..N] of integer;
  I, J : integer;
begin
  for I :=1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,'): ');
      readLn(M[I,J]);
    end;
  writeLn('Elementos da diagonal secundaria');
  for I := 1 to N do
  begin
    for J := 1 to N do
      if ( J = (N - I + 1) ) then
        write(M[I,J],chr(9))
      else
        write(chr(9));
    writeLn('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J)=input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
disp('Elementos da Diagonal Secundaria:');
for I = 1 : N
  for J = 1 : N
    if ( J == (N - I + 1) )
      fprintf(1,'%d\t',M(I,J));
    else
      fprintf(1,'\t');
    end
  end
  fprintf(1,'\n');
end
```

8) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva todos os elementos exceto os elementos da diagonal secundária.

```
algoritmo L5P08;
constante
  N = 10;
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N  faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      se ( I <> N - J + 1 ) então
        imprima(M[I,J]);
      fim-se
    fim-para
  fim-para
fim
```

```
program L5P08;
const
  N = 10;
var
  I, J : integer;
  M : array [1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(M [I,J]);
    end;
  writeLn('Elementos acima da diagonal principal');
  for I := 1 to N do
    for J := 1 to N do
      if ( I <> N - J + 1) then
        writeln(M[I,J]);
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
disp('Elementos acima da diagonal principal');
for I = 1 : N
  for J = 1 : N
    if ( I ~= N - J + 1 )
      fprintf(1,'%d\t',M(I,J));
    else
      fprintf(1,'\t');
    end
  end
  fprintf(1,'\n');
end
```

9) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva somente os elementos acima da diagonal secundária.

```
algoritmo L5P09;
constante
  N = 10;
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N faça
    para  J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      se ( I < N - J + 1 ) então
        imprima(M[I,J]);
      fim-se
    fim-para
  fim-para
fim
```

```
program L5P09;
const
  N = 10;
var
  I, J : integer;
  M : array[1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      writeln('Digite o elemento (',I,',',J,') da matriz: ');
      readln(M[I,J]);
    end;
  writeLn('Elementos acima da diagonal secundaria');
  for I := 1 to N do
  begin
    for J := 1 to N do
      if ( I < N - J + 1) then
        write(M[I,J],chr(9))
      else
        write(chr(9));
    writeLn('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
disp('Elementos acima da diagonal secundaria');
for I = 1 : N
  for J = 1 : N
    if ( I < N - J + 1)
      fprintf(1,'%d\t',M(I,J));
    else
      fprintf(1,'\t');
    end
  end
  fprintf(1,'\n');
end
```

10) Criar um algoritmo que leia os elementos de uma matriz inteira 10 x 10 e escreva somente os elementos abaixo da diagonal secundária.

```
algoritmo L5P10;
constante
  N = 10;
var
  inteiro   : I, J, M[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 2 até N faça
    para J de N-I+2 até N faça
      imprima(M[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P10;
const
  N = 10;
var
  I, J : integer;
  M : array [1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(M[I,J]);
    end;
  writeLn('Elementos abaixo da Diagonal Secundaria');
  for I := 2 to N do
  begin
    for J := 1 to N-I+1 do
      write(chr(9));
    for J := N-I+2 to N do
      write(M[I,J],chr(9));
    writeLn('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
disp('Elementos abaixo da Diagonal Secundaria');
for I = 2 : N
  for J = 1 : N-I+1
    fprintf(1,'\t');
  end
  for J = N-I+2 : N
    fprintf(1,'%d\t',M(I,J));
  end
  fprintf(1,'\n');
end
```

11) Entrar com valores para uma matriz $A_{3x4}$. Gerar e imprimir uma matriz B que é o triplo da matriz A.

```
algoritmo L5P11;
var
  inteiro : I, J;
  real    : MA[1..3,1..4], MB[1..3,1..4];
início
  para I de 1 até 3 faça
    para J de 1 até 4 faça
      leia(MA[I,J]);
    fim-para
  fim-para
  para I de 1 até 3 faça
    para J de 1 até 4 faça
      MB[I,J] <- 3 * A[I,J];
    fim-para
  fim-para
  para I de 1 até 3 faça
    para J de 1 até 4 faça
      imprima(MB[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P11;
var
  I, J : integer;
  MA, MB : array[1..3,1..4] of real;
begin
  for I := 1 to 3 do
    for J := 1 to 4 do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(MA[I,J]);
    end;
  for I := 1 to 3 do
    for J := 1 to 4 do
      MB[I,J] := 3 * MA[I,J];
  for I := 1 to 3 do
  begin
    for J := 1 to 4 do
      write(MB[I,J],' ');
    writeLn(' ');
  end;
end.
```

```
for I = 1 : 3
  for J = 1 : 4
    MA(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
for I = 1 : 3
  for J = 1 : 4
    MB(I,J) = 3 * MA(I,J);
  end
end
for I = 1 : 3
  for J = 1 : 4
    fprintf(1,'%f\t',MB(I,J));
  end
  fprintf(1,'\n');
end
```

12) Entrar com valores inteiros para um matriz $A_{4x4}$ e para uma matriz $B_{4x4}$. Gerar e imprimir a SOMA (A+B).

```
algoritmo L5P12;
constante
  N = 4;
var
  inteiro   : I, J, MA[1..N,1..N], MB[1..N,1..N], MS[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(MA[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      leia(MB[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      MS[I,J] <- MA[I,J] + MB[I,J];
    fim-para
  fim-para
  imprima ("Matriz SOMA: ");
  para I de 1 até N faça
    para J de 1 até N faça
      imprima(MS[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P12;
const
  N = 4;
var
  MA, MB, MS : array [1..N,1..N] of integer;
  I, J : integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o termo (',I,'x',J,') da matriz A: ');
      readLn(MA[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o termo (',I,'x',J,') da matriz B: ');
      readLn(MB[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
      MS[I,J] := MA[I,J] + MB[I,J];
  writeLn('Matriz SOMA: ');
  for I := 1 to N do
  begin
    for J := 1 to N do
      write(MS[I,J],chr(9));
    writeLn('');
  end;
end.
```

```
N = 4;
for I = 1 : N
  for J = 1 : N
    MA(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz A: ', I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    MB(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz B: ', I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    MS(I,J) = MA(I,J) + MB(I,J);
  end
end
disp('Matriz Soma: ');
for I = 1 : N
  for J = 1 : N
    fprintf(1,'%d\t',MS(I,J));
  end
  fprintf(1,'\n');
end
```

13) Entrar com valores para duas matrizes inteiras de ordem cinco. Gerar e imprimir a matriz diferença.

```
algoritmo L5P13;
constante
  N = 5;
var
  inteiro   : I, J, MA[1..5,1..5], MB[1..5,1..5], MD[1..5,1..5];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(MA[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      leia(MB[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      MD[I,J] <- MA[I,J] - MB[I,J];
    fim-para
  fim-para
  imprima("matriz diferença: ");
  para I de 1 até N faça
    para J de 1 até N faça
      imprima(MD[I,J])
    fim-para
  fim-para
fim
```

```
program L5P13;
const
  N = 5;
var
  I, J : integer;
  MA, MB, MD : array [1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz A: ');
      readLn(MA[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz B: ');
      readLn(MB[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
      MD[I,J] := MA[I,J] - MB[I,J];
  writeLn('A matriz diferenca:');
  for I := 1 to N do
  begin
    for J := 1 to N do
      write(MD[I,J],chr(9));
    writeLn('');
  end;
end.
```

```
N = 5;
for I = 1 : N
  for J = 1 : N
    MA = input(sprintf('Digite o elemento (%dx%d) da matriz A: ',I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    MB = input(sprintf('Digite o elemento (%dx%d) da matriz B: ',I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    MD(I,J) = MA(I,J) - MB(I,J);
  end
end
disp('A matriz diferença e:');
for I = 1 : N
  for J = 1 : N
    fprintf(1,'%d\t',MD(I,J));
  end
  fprintf(1,'\n');
end
```

14) Ler uma matriz 4x5 de inteiros, calcular e imprimir a soma de todos os seus elementos.

```
algoritmo L5P14;
var
  inteiro : I, J, S, M[1..4,1..5];
início
  para I de 1 até 4 faça
    para J de 1 até 5 faça
      leia(M[I,J]);
    fim-para
  fim-para
  S <- 0;
  para I de 1 até 4 faça
    para J de 1 até 5 faça
      S <- S + M [I,J];
    fim-para
  fim-para
  imprima(S);
fim
```

```
program L5P14;
var
  I, J, S : integer;
  M : array[1..4,1..5] of integer;
begin
  for I := 1 to 4 do
    for J := 1 to 5 do
    begin
      writeLn('Digite o elemento (',I,',',J,'): ');
      readLn(M[I,J]);
    end;
  S := 0;
  for I := 1 to 4 do
    for J := 1 to 5 do
      S := S + M[I,J];
  writeLn('A soma eh:',S);
end.
```

```
for I = 1 : 4
  for J = 1 : 5
    M(I,J) = input(sprintf('(%dx%d): ',I,J));
  end
end
for I = 1 : 4
  for J = 1 : 5
    S = S + M(I,J);
  end
end
disp(S);
```

15) Ler valores inteiros para a matriz $A_{3x5}$. Gerar e imprimir a matriz (vetor) SL (soma das 3 linhas), onde cada elemento é a soma dos elementos de uma linha da matriz A. Faça o trecho que gera a matriz SL separado (laços de repetição) da entrada e da saída de dados.

```
algoritmo L5P15;
var
  inteiro : I, J, MA[1..3,1..5], SL[1..3];
início
  { entrada de dados }
  para I de 1 até 3 faça
    para J de 1 até 5 faça
      leia(MA[I,J]);
    fim-para
  fim-para
  { cálculo de SL }
  para I de 1 até 3 faça
    SL[I] <- 0;
    para J de 1 até 5 faça
      SL[I] <- SL[I] + MA[I,J];
    fim-para
  fim-para
  { saída de dados }
  para I de 1 até 3 faça
    imprima(SL[I]);
  fim-para
fim
```

```
program L5P15;
var
  I, J : integer;
  MA : array [1..3,1..5] of integer;
  SL : array [1..3] of integer;
begin
  { entrada de dados }
  for I := 1 to 3 do
    for J := 1 to 5 do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(MA[I,J]);
    end;
  { calculo de SL }
  for I := 1 to 3 do
  begin
    SL[I] := 0;
    for J := 1 to 5 do
      SL[I] := SL[I] + MA[I,J];
  end;
  { saida de dados }
  for I := 1 to 3 do
    writeLn('Soma da ',I,'a. linha: ',SL[I]);
end.
```

```
% entrada de dados
for I = 1 : 3
  for J = 1 : 5
    MA(I,J) = input(sprintf('Digite Elemento (%dx%d) da matriz: ',I,J));
  end;
end
% calculo de SL
for I = 1 : 3
  SL(I) = 0;
  for J = 1 : 5
    SL(I) = SL(I) + MA(I,J);
  end
end
% saida de dados
for I = 1 : 3
  fprintf(1,'Soma da %da. linha: %d\n',I,SL(I));
end
```

16) Uma floricultura conhecedora de sua clientela gostaria de fazer um algoritmo que pudesse controlar sempre um estoque mínimo de determinadas plantas, pois todo dias, pela manhã, o dono faz novas aquisições. Criar um algoritmo que deixe cadastrar 50 tipos de plantas e nunca deixar o estoque ficar abaixo do ideal. Para cada planta, o dono gostaria de cadastrar o nome, o estoque ideal e a quantidade em estoque. Dessa forma o algoritmo pode calcular a quantidade que o dono da loja precisa comprar no próximo dia. Essa quantidade a ser comprada deve ser impressa (quando maior que zero) como uma lista para o dono da floricultura.

```
algoritmo L5P16;
var
  literal : TP[1..50,80];
  inteiro : I, J, M[1..50,1..3];
início
  para I de 1 até 50 faça
    imprima("Planta: ");
    leia(TP[I]);
    leia(M[I,1]);
    leia(M[I,2]);
    M[I,3] <- M[I,2] - M[I,1];
  fim-para
  para I de 1 até 50 faça
    se ( M[I,3] < 0 ) então
      imprima(TP[I],-(M[I,3]));
    fim-se
  fim-para
fim
```

```
program L5P16;
var TP : array[1..50] of string[80];
    M : array[1..50,1..3] of integer;
    I, J : integer;
begin
  for I:=1 to 50 do
  begin
    write('Planta: ');
    readLn(TP[I]);
    write('Quantidade ideal para estoque: ');
    readLn(M[I,1]);
    write('Quantidade atual no estoque: ');
    readLn(M[I,2]);
    M[I,3] := M[I,2]-M[I,1];
  end;
  for I:=1 to 50 do
    if ( M[I,3] < 0 ) then
      writeLn(TP[I],' comprar: ',-(M[I,3]));
end.
```

```
for I = 1 : 50
  TP{I} = input('Planta: ','s');
  M(I,1) = input('Quantidade ideal para estoque: ');
  M(I,2) = input('Quantidade atual no estoque: ');
  M(I,3) = M(I,2) - M(I,1);
end
for I = 1 : 50
  if ( M(I,3) < 0 )
    fprintf(1,'%s comprar %d\n',TP{I},-M(I,3));
  end
end
```

17) A gerente do cabeleireiro Sempre Bela tem uma tabela em que registra os "pés" as "mãos" e o serviço de podologia das cinco manicures. Sabendo-se que cada uma ganha 50% do que faturou ao mês, criar um algoritmo que possa calcular e imprimir quanto cada um vai receber, uma vez que não têm carteiras assinadas; os valores, respectivamente, são R$ 10,00; R$ 15,00 e R$ 30,00.

```
algoritmo L5P17;
var
  inteiro : S, M, MAT[1..5,1..3];
  real    :    SAL[1..5];
início
  para M de 1 até 5 faça
    imprima("Manicure ",M,": ");
    para S de 1 até 3 faça
      leia(MAT[M,S]);
    fim-para
  fim-para
  para M de 1 até 5 faça
    para S de 1 até 3 faça
      se ( S = 1 ) então
        MAT[M,S] <- MAT[M,S]*10;
      fim-se
      se ( S = 2 ) então
        MAT[M,S] <- MAT[M,S]*15;
      fim-se
      se ( S = 3 ) então
        MAT[M,S] <- MAT[M,S]*30;
      fim-se
    fim-para
  fim-para
  para M de 1 até 5 faça
    SAL[M] <- 0;
  fim-para
  para M de 1 até 5 faça
    para S de 1 até 3 faça
      SAL[M] <- MAT[M,S] / 2 + SAL[M];
    fim-para
  fim-para
  para M de 1 até 15 faça
    imprima(M,SAL[M]);
  fim-para
fim
```

```
program L4P17;
var
   MAT : array [1..5,1..3] of integer;
   SAL : array [1..5] of real;
   S, M : integer;
begin
  for M := 1 to 5 do
  begin
    writeLn ('Manicure ',M,': ');
    for S := 1 to 3 do
    begin
      if ( S = 1 ) then
        write('Digite o numero de "pes" feitos: ');
      if ( S = 2 ) then
        write('Digite o numero de "maos" feitas: ');
      if ( S = 3 ) then
        write('Digite o numero de servicos de podologia feitos: ');
      readLn(MAT[M,S]);
    end;
  end;
  for M := 1 to 5 do
    for S := 1 to 3 do
    begin
      if ( S = 1 ) then
        MAT[M,S] := MAT[M,S]*10;
      if ( S = 2 ) then
        MAT[M,S] := MAT[M,S]*15;
      if ( S = 3 ) then
        MAT[M,S] := MAT[M,S]*30;
    end;
  for M := 1 to 5 do
    SAL[M] := 0;
  for M := 1 to 5 do
    for S := 1 to 3 do
      SAL[M] := MAT[M,S] / 2 + SAL[M];
  writeLn ('Salario das manicures:');
  for M := 1 to 5 do
    writeLn('Manicure ',M,': ',SAL[M]);
end.
```

```
for M = 1 : 5
  fprintf(1,'Manicure: %d\n',M);
  for S = 1 : 3
    if ( S == 1 )
      MAT(M,S) = input('Digite o numero de pes feitos: ');
    end
    if ( S == 2 )
      MAT(M,S) = input('Digite o numero de maos feitas: ');
    end
    if ( S == 3 )
      MAT(M,S) = input('Digite o numero de serviços de podologia feitos: ');
    end
  end
end
for M = 1 : 5
  for S = 1 : 3
    if ( S == 1 )
      MAT(M,S) = MAT(M,S)*10;
    end
    if ( S == 2 )
      MAT(M,S) = MAT(M,S)*15;
    end
    if ( S == 3 )
      MAT(M,S) = MAT(M,S)*30;
    end
  end
end
for M = 1 : 5
  SAL(M) = 0;
end
for M = 1 : 5
  for S = 1 : 3
    SAL(M) = MAT(M,S) / 2 + SAL(M);
  end
end
disp('Salarios das Manicures:');
for M = 1 : 5
    fprintf(1,'Manicure %d: %d\n',M,SAL(M));
end
```

18) A matriz dados contém na 1ª coluna a matrícula do aluno no curso; na 2ª, o sexo (0 para feminino e 1 para masculino); na 3ª, o código do curso, e na 4ª, o CR (Coeficiente de Rendimento). Suponha 10 alunos e que o CR é um número inteiro.
Faça um algoritmo que armazene esses dados sabendo-se que:

- O código do curso é uma parte de um número de matrícula: aasccccnnn (aa ano, s semestre, ccc código do curso e nnn matrícula no curso), que deve ser lido; Além, disso, o sexo e o CR devem ser lidos também.
Um grupo empresarial resolveu premiar a aluna com CR mais alto de um curso cujo código deverá ser digitado.

```
algoritmo L5P18;
constante
  N = 10;
var
  inteiro   : I, J, K, C;
  inteiro   : COD, AUX, MAT[1..N,1..4];
início
  para C de 1 até N faça
    leia(COD)
    MAT[C,1] <- COD mod 1000; { matricula no curso }
    MAT[C,3] <- ( COD div 1000 ) mod 10000; { codigo do curso }
    leia(MAT[C,2]); { sexo }
    leia(MAT[C,4]); { CR }
  fim-para;
  { ordenando }
  para I de 1 até N-1 faça
    K <- I;
    para J de I+1 até N faça
      se ( MAT[K,4] < MAT[J,4] )
        K <- J;
      fim-se
    fim-para
    AUX <- MAT[I,1]; MAT[I,1] <- MAT[K,1]; MAT[K,1] <- AUX;
    AUX <- MAT[I,2]; MAT[I,2] <- MAT[K,2]; MAT[K,2] <- AUX;
    AUX <- MAT[I,3]; MAT[I,3] <- MAT[K,3]; MAT[K,3] <- AUX;
    AUX <- MAT[I,4]; MAT[I,4] <- MAT[K,4]; MAT[K,4] <- AUX;
  fim-para
  leia(COD);
  para C de 1 até N faça
    se ( COD = MAT[C,3] ) e ( MAT[C,2] = 0 )
      imprima(MAT[C,1],MAT[C,4]);
      C <- N;
    fim-se
  fim-para
fim
```

```
program L5P18;
const
  N = 10;
var
  I, J, K, C : integer;
  COD : real;
  AUX : integer;
  MAT : array [1..10,1..4] of integer;
begin
  for C := 1 to N do
  begin
    write('Digite o codigo do aluno: ');
    readLn(COD);
    MAT[C,1] := trunc((COD/1000-trunc(COD/1000))*1000); { matricula no curso}
    MAT[C,3] := trunc(COD/1000);
    MAT[C,3] := trunc((MAT[C,3]/10000-trunc(MAT[c,3]/10000))*10000);
    {codigo do curso}
    write('Digite o sexo do aluno (0=F,1=M): ');
    readLn(MAT[C,2]); { sexo }
    write('Digite o Coeficiente de Rendimento do Aluno: ');
    readLn(MAT[C,4]); { CR }
  end;
  { ordenando }
  for I := 1 to N-1 do
  begin
    K := I;
    for J := I+1 to N do
    begin
      if MAT[K,4] < MAT[J,4] then
        K := J;
    end;
    AUX := MAT[I,1]; MAT[I,1] := MAT[K,1]; MAT[K,1] := AUX;
    AUX := MAT[I,2]; MAT[I,2] := MAT[K,2]; MAT[K,2] := AUX;
    AUX := MAT[I,3]; MAT[I,3] := MAT[K,3]; MAT[K,3] := AUX;
    AUX := MAT[I,4]; MAT[I,4] := MAT[K,4]; MAT[K,4] := AUX;
  end;
  write('Qual o codigo do curso: ');
  readLn(COD);
  for C := 1 to N do
    if ( COD = MAT[C,3] ) and ( MAT[C,2] = 0 ) then
    begin
      write(MAT[C,1]:4,' ',MAT[C,4]:4);
      C := N; { finaliza loop }
    end;
end.
```

```
N = 10;
for C = 1 : N
  COD = input('Digite o codigo do aluno: ');
  MAT(C,1) = mod(COD,1000); % matricula no curso
  MAT(C,3) = mod(floor(COD/1000),10000); % codigo do curso
  MAT(C,2) = input('Digite o sexo do aluno (0=F,1=M): '); % sexo
  MAT(C,4) = input('Digite o Coeficiente de Rendimento do Aluno: '); % CR
end
% ordenando
for I = 1 : N-1
  K = I;
  for J = I+1 : N
    if ( MAT(K,4) < MAT(J,4) )
      K = J;
    end
  end
  AUX = MAT(I,1); MAT(I,1) = MAT(K,1); MAT(K,1) = AUX;
  AUX = MAT(I,2); MAT(I,2) = MAT(K,2); MAT(K,2) = AUX;
  AUX = MAT(I,3); MAT(I,3) = MAT(K,3); MAT(K,3) = AUX;
  AUX = MAT(I,4); MAT(I,4) = MAT(K,4); MAT(K,4) = AUX;
end
COD = input('Qual o codigo do curso: ');
for C = 1 : N
  if ( COD == MAT(C,3) ) & ( MAT(C,2) == 0 )
    fprintf(1,'%d %d\n',MAT(C,1),MAT(C,4));
    break;
  end
end
```

19) Criar um algoritmo que possa armazenar as alturas de dez atletas de cinco delegações que participarão dos jogos de verão. Imprimir a maior altura de cada delegação.

```
algoritmo L5P19;
var
  inteiro : I, J;
  real    : M[1..5,1..10], MAIOR[1..5];
início
  para I de 1 até 5 faça
    para J de 1 até 10 faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até 5 faça
    MAIOR[I] <- M[I,1];
    para J de 2 até 10 faça
      se ( M[I,J] > MAIOR[I] ) então
        MAIOR[I] <- M[I,J];
      fim-se
    fim-para
  fim-para
  para I de 1 até 5 faça
    imprima(I,MAIOR[I]);
  fim-para
fim
```

```
program L5P19;
var
  I, J : integer;
  M : array [1..5,1..10] of real;
  MAIOR : array [1..5] of real;
begin
  for I := 1 to 5 do
    for J := 1 to 10 do
    begin
      write('Digite a altura do ',J,'o. atleta da ',I,'a. delegacao: ');
      readLn(M[I,J]);
    end;
  for I := 1 to 5 do
  begin
    MAIOR[I] := M[I,1];
    for J := 2 to 10 do
      if ( M[I,J] > MAIOR[I] ) then
        MAIOR[I] := M[I,J];
  end;
  for I := 1 to 5 do
    writeln('A maior altura da delegacao ',I,'a. eh:',MAIOR[I]);
end.
```

```
for I = 1 : 5
  for J = 1 : 10
    M(I,J) = input(sprintf('Digite a altura do %do. atleta da %da. delegacao: ',J,I));
  end
end
for I = 1 : 5
  MAIOR(I) = M(I,1);
  for J = 2 : 10
    if ( M(I,J) > MAIOR )
      MAIOR(I) = M(I,J);
    end
  end
end
for I = 1 : 5
  fprintf('A maior altura da %da. delegacao eh: %f\n',I,M(I,J));
end
```

20) Criar um algoritmo que carregue uma matriz 12 x 4 com os valores das vendas de uma loja, em que cada linha represente um mês do ano, e cada coluna, uma semana do mês. Para fins de simplificação considere que cada mês possui somente 4 semanas. Calcule e imprima:

- Total vendido em cada mês do ano;
- Total vendido em cada semana durante todo o ano;
- Total vendido no ano.

```
algoritmo L5P20;
var
  inteiro : I, J;
  real    : TOTANO, TOTMES, TOTSEM, MAT[1..12,1..4];
início
  para I de 1 até 12 faça
    para J de 1 até 4 faça
      leia(MAT[I,J]);
    fim-para
  fim-para
  TOTANO <- 0;
  para I de 1 até 12 faça
    TOTMES <- 0;
    para J de 1 até 4 faça
      TOTMES <- TOTMES + MAT[I,J];
    fim-para
    imprima(I,TOTMES);
    TOTANO <- TOTANO + TOTMES;
  fim-para
  para J de 1 até 4 faça
    TOTSEM <- 0;
    para I de 1 até 12 faça
      TOTSEM <- TOTSEM + MAT[I,J];
    fim-para
    imprima(J,TOTSEM);
  fim-para
  imprima(TOTANO);
fim
```

```
program L5P20;
var
  I, J : integer;
  TOTANO, TOTMES, TOTSEM : real;
  MAT : array [1..12,1..4] of real;
begin
  for I := 1 to 12 do
    for J := 1 to 4 do
    begin
      write('Mes - ',I,', Semana - ',J,': ');
      readLn(MAT[I,J]);
    end;
  TOTANO := 0;
  for I := 1 to 12 do
  begin
    TOTMES := 0;
    for J := 1 to 4 do
      TOTMES := TOTMES + MAT[I,J];
    writeLn('Total do mes ',I,': ',TOTMES:3:2);
    TOTANO := TOTANO + TOTMES;
  end;
  for J := 1 to 4 do
  begin
    TOTSEM := 0;
    for I := 1 to 12 do
      TOTSEM := TOTSEM + MAT[I,J];
    writeLn('Total da semana ',J,': ',TOTSEM:3:2);
  end;
  writeLn('Total do ano: ',TOTANO:3:2);
end.
```

```
for I = 1 : 12
  for J = 1 : 4
    MAT(I,J) = input(sprintf('Mes - %d, Semana - %d: ',I,J));
  end
end
TOTANO = 0;
for I = 1 : 12
  TOTMES = 0;
  for J = 1 : 4
    TOTMES = TOTMES + MAT(I,J);
  end
  fprintf(1,'Total do mes %d: %3.2f\n',I,TOTMES);
  TOTANO = TOTANO + TOTMES;
end
for J = 1 : 4
  TOTSEM = 0;
  for I = 1 : 12
    TOTSEM = TOTSEM + MAT(I,J);
  end
    fprintf(1,'Total da semana %d: %3.2f\n',I,TOTSEM);

end
fprintf(1,'Total do ano: %3.2f\n',TOTANO);
```

21) Criar um algoritmo que entre com valores inteiros para uma matriz m 3 x 3 e imprima a matriz final, conforme mostrado a seguir:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \text{ a matriz gira } 90^{o} \begin{bmatrix} 7 & 4 & 1 \\ 8 & 5 & 2 \\ 9 & 6 & 3 \end{bmatrix}$$

```
algoritmo L5P21;
constante
  N = 3;
var
  inteiro   : I, J, M[1..N,1..N], F[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(N[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      F[J,N-I+1] <- M[I,J];
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      imprima(F[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P21;
const
  N = 3;
var
  I, J : integer;
  M, F : array[1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(M[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
      F[J,N-I+1] := M[I,J];
  for I := 1 to N do
  begin
    for J := 1 to N do
      write(F[I,J],chr(9));
    writeLn(' ');
  end;
end.
```

```
N = 3;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o o elemento (%dx%d) da matriz: ',I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    F(J,N-I+1) = M(I,J);
  end
end
for I = 1 : N
  for J = 1 : N
    fprintf(1,'%d\t',F(I,J));
  end
  fprintf(1,'\n');
end
```

22) Criar um algoritmo que entre com valores inteiros para uma matriz m 3 x 3 e imprima a matriz final, conforme mostrado a seguir:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \text{ a matriz gira } 180^{o} \begin{bmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

```
algoritmo L5P22;
constante
  N = 3;
var
  inteiro   : I, J, M[1..N,1..N], F[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      F[N-I+1,N-J+1] <- M[I,J];
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      imprima(F[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P22;
const
  N = 3;
var
  I, J : integer;
  M, F : array [1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(M[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
      F[N-I+1,N-J+1] := M[I,J];
  for I := 1 to N do
  begin
    for J := 1 to N do
      write(F[I,J],chr(9));
    writeLn('');
  end;
end.
```

```
N = 3;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    F(N-I+1,N-J+1) = M(I,J);
  end
end
for I = 1 : N
  for J = 1 : N
    fprintf(1,'%d\t',F(I,J));
  end
  fprintf(1,'\n');
end
```

23) Criar um algoritmo que entre com valores inteiros para uma matriz m 3 x 3 e imprima a matriz final, conforme mostrado a seguir:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \text{ a matriz gira } 270^{o} \begin{bmatrix} 3 & 6 & 9 \\ 2 & 5 & 8 \\ 1 & 4 & 7 \end{bmatrix}$$

```
algoritmo L5P23;
constante
  N = 3;
var
  inteiro   : I, J, M[1..N,1..N], F[1..N,1..N];
início
  para I de 1 até N faça
    para J de 1 até N faça
      leia(M[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      F[N-J+1,I] <- M[I,J];
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      imprima(F[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P23;
const
  N = 3;
var
  I, J : integer;
  M, F : array [1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J :=1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(M[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
      F[N-J+1,I] := M[I,J];
  for I := 1 to N do
  begin
    for J := 1 to N do
      write(F[I,J],chr(9));
    writeLn('');
  end;
end.
```

```
N = 3;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%d,%d) da matriz: ',I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    F(N-J+1,I) = M(I,J);
  end
end
for I = 1 : N
  for J = 1 : N
    fprintf(1,'%d\t',F(I,J));
  end
  fprintf(1,'\n');
end
```

24) Criar um algoritmo que leia e armazene os elementos de uma matriz inteira $M_{10x10}$ e imprimi-la. Troque, na ordem a seguir:
- a segunda linha pela oitava linha;
- a quarta coluna pela décima coluna;
- a diagonal principal pela diagonal secundária.

```
algoritmo L5P24;
constante
  N = 10;
var
  inteiro   : I, J, AUX, M[1..N,1..N];
inicio
  para I de 1 até N faça
    para J de 1 até N faça
      leia (M[I,J]);
    fim-para
  fim-para
  para J de 1 até N faça
    AUX    <- M[8,J];
    M[8,J] <- M[2,J];
    M[2,J] <- AUX;
  fim-para
  para I de 1 até N faça
    AUX     <- M[I,10];
    M[I,10] <- M[I, 4];
    M[I, 4] <- AUX;
  fim-para
  para I de 1 até N faça
    AUX        <- M[I,    I];
    M[I,    I] <- M[I,N-I+1];
    M[I,N-I+1] <- AUX;
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      imprima(M[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P24;
const
  N = 10;
var
  I, J, AUX : integer;
  M : array[1..N,1..N] of integer;
begin
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('entre com os elementos(',I,',',J,'):');
      readLn(M[I,J]);
    end;
  for J := 1 to N do
  begin
    AUX    := M[8,J];
    M[8,J] := M[2,J];
    M[2,J] := AUX;
  end;
  for I := 1 to N do
  begin
    AUX    := M[I,10];
    M[I,10]:= M[I, 4];
    M[I, 4]:= AUX;
  end;
  for I := 1 to N do
  begin
     AUX        := M[I,    I];
     M[I,I    ] := M[I,N-I+1];
     M[I,N-I+1] := AUX;
  end;
  for I := 1 to N do
  begin
    for J := 1 to N do
      write(M[I,J],chr(9));
    writeln('');
  end;
end.
```

```
N = 10;
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('(%dx%d): ',I,J));
  end
end
for J = 1 : N
  AUX    = M(8,J);
  M(8,J) = M(2,J);
  M(2,J) = AUX;
end
for I = 1 : N
  AUX     = M(I,10);
  M(I,10) = M(I, 4);
  M(I, 4) = AUX;
end
for I = 1 : N
  AUX        = M(I,    I);
  M(I,    I) = M(I,N-I+1);
  M(I,N-I+1) = AUX;
end
for I = 1 : N
  for J = 1 : N
    fprintf(1,'%d\t',M(I,J));
  end
  fprintf(1,'\n');
end
```

25) Criar um algoritmo que leia valores para uma matriz $M_{2x2}$. Calcular e imprimir o determinante. Para cálculo do determinante de uma matriz de ordem 2, é simplesmente computar a diferença entre os produtos das diagonais principal e secundária, respectivamente.

```
algoritmo L5P25;
var
  inteiro : I, J;
  real    : DET, M[1..2,1..2];
início
  para I de 1 até 2 faça
    para J de 1 até 2 faça
      leia(M[I,J]);
    fim-para
  fim-para
  DET <- M[1,1] * M[2,2] - M[1,2] * M[2,1];
  imprima(DET);
fim
```

```
program L5P25;
var
  I, J : integer;
  DET : real;
  M : array [1..2,1..2] of real;
begin
  for I := 1 to 2 do
    for J := 1 to 2 do
    begin
      write('Elemento (',I,',',J,'): ');
      readLn(M[I,J]);
    end;
  DET := M[1,1] * M[2,2] - M[1,2] * M[2,1];
  writeLn('Determinante: ',DET:5:4);
end.
```

```
for I = 1 : 2
  for J = 1 : 2
    M(I,J) = input(sprintf('Elemento (%d,%d): ',I,J));
  end
end
DET = M(1,1) * M(2,2) - M(1,2) * M(2,1);
fprintf(1,'Determinante: %5.4f\n',DET);
```

26) Criar um algoritmo que leia uma matriz $A_{NxN}$ ($N \leq 10$) e calcule a respectiva matriz transposta $A^t$.

```
algoritmo L5P26;
constante
  N = 10;
var
  inteiro   : I, J, N;
  real      : A[1..N,1..N], AT[1..N,1..N];
início
  leia(N);
  para I de 1 até N faça
    para J de 1 até N faça
      leia(A[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      AT[J,I] <- A[I,J];
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      imprima(AT[I,J]);
    fim-para
  fim-para
fim
```

```
program L5P26;
var
  N, I, J : integer;
  A, AT : array[1..10,1..10] of real;
begin
  write('Entre com a ordem da matriz: ');
  readLn(N);
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(A[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
      AT[I,J] := A[J,I];
  for I := 1 to N do
  begin
    for J := 1 to N do
      write(AT[I,J]:3:2,chr(9));
    writeLn(' ');
  end;
end.
```

```
N=input('Entre com a ordem da matriz: ');
for I = 1 : N
  for J = 1 : N
    A(I,J) = input(sprintf('Entre com o elemento %dx%d: \n',I,J));
  end
end
for I = 1 : N
  for I = 1 : N
    AT(J,I) = A(I,J);
    fprintf(1,'%.2f\t',AT(I,J));
  end
  fprintf(1,'\n');
end
```

27) Criar um algoritmo que leia uma matriz $A_{NxN}$ ($N \leq 10$) e verifique (informe) se tal matriz é ou não simétrica ($A^t = A$).

```
algoritmo L5P27;
var
  lógico  : SIT;
  inteiro : I, J, N;
  real    : A[1..K,1..K];
início
  leia(N);
  para I de 1 até N faça
    para J de 1 até N faça
      leia(A[I,J]);
    fim-para
  fim-para
  SIT <- verdadeiro;
  para I de 1 até N faça
    para J de I até N faça
      se ( A[I,J] <> A[J,I] ) então
        SIT <- falso;
      fim-se
    fim-para
  fim-para
  se ( SIT = verdadeiro ) então
    imprima("A matriz é simétrica!");
  senão
    imprima("A matriz não é simétrica!");
  fim-se
fim
```

```
program L5P27;
var
  SIT : boolean;
  I, J, N : integer;
  A : array [1..10,1..10] of real;
begin
  write('Digite a ordem da matriz (NxN): ');
  readLn(N);
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,'): ');
      readLn(A[I,J]);
    end;
  SIT := true;
  for I := 1 to N do
    for J := I to N do
      if ( A[I,J] <> A[J,I] ) then
        SIT := false;
  if ( SIT = true ) then
    writeLn('A matriz eh simetrica.')
  else
    writeLn('A matriz nao eh simetrica.');
end.
```

```
N = input('Digite o numero de linhas da matriz: ');
for I = 1 : N
  for J = 1 : N
    M(I,J) = input(sprintf('Digite o elemento (%dx%d) da matriz: ',I,J));
  end
end
SIT = 1;
for I = 1 : N
  for J = I : N
    if ( M(I,J) ~= M(J,I) )
      SIT = 0;
    end
  end
end
if ( SIT = 1 )
  disp('A matriz eh simetrica!');
else
  disp('A matriz nao eh simetrica!');
end
```

28) Criar um algoritmo que leia uma matriz $A_{NxN}$ ($N \leq 10$) e verifique (informe) se tal matriz é ou não anti-simétrica ($A^t = -A$).

```
algoritmo L5P28;
var
  lógico  : SIT;
  inteiro : I, J,;
  real    : A[1..10,1..10], AT[1..10,1..10], AS[1..10,1..10];
início
  leia(N);
  para I de 1 até N faça
    para J de 1 até N faça
      leia(A[I,J]);
    fim-para
  fim-para
  para I de 1 até N faça
    para J de 1 até N faça
      AT[I,J] <-  A[J,I];
      AS[I,J] <- -A[I,J];
    fim-para
  fim-para
  SIT <- verdadeiro;
  para I de 1 até N faça
    para J de 1 até N faça
      se ( AT[I,J] <> AS[I,J] ) então
        SIT <- falso;
      fim-se
    fim-para
  fim-para
  se (SIT = verdadeiro) então
    imprima("A matriz A é anti-simétrica!");
  senão
    imprima("A matriz A não é anti-simétrica!");
  fim-se
fim
```

```
program L5P28;
var
  SIT : boolean;
  I, J, N : integer;
  A, AS, AT : array [1..10,1..10] of real;
begin
  write('Qual a ordem da matriz A (NxN): ');
  readLn(N);
  for I := 1 to N do
    for J := 1 to N do
    begin
      write('Digite o elemento (',I,'x',J,') da matriz: ');
      readLn(A[I,J]);
    end;
  for I := 1 to N do
    for J := 1 to N do
    begin
      AT[I,J] := A[J,I];
      AS[I,J] := -A[I,J];
    end;
  SIT := true;
  for I := 1 to N do
    for J := 1 to N do
      if (AT[I,J] <> AS[I,J]) then
        SIT := false;
  if (SIT = true) then
    writeLn('A matriz A eh anti-simetrica!')
  else
    writeLn('A matriz A nao eh anti-simetrica!');
end.
```

```
N = input('Qual o número de colunas da Matriz A (NxN): ');
for I = 1 : N
  for J = 1 : N
    A(I,J) = input(sprintf('Digite o elemento (%d,%d) da matriz: ',I,J));
  end
end
for I = 1 : N
  for J = 1 : N
    AT(I,J) = A(J,I);
    AS(I,J) = -A(I,J);
  end
end
SIT = 1;
for I = 1 : N
  for J = 1 : N
    if ( AT(I,J) ~= AS(I,J) )
      SIT = 0;
    end
  end
end
if ( SIT == 1 )
  disp('A matriz A é anti-simétrica!');
else
  disp('A matriz A não é anti-simétrica!');
end
```

29) Criar um algoritmo que leia uma matriz $A_{2x2}$ e calcule a respectiva inversa $A^{-1}$.

```
algoritmo L5P29;
var
  inteiro : I, J;
  real    : DET, C, A[1..2,1..2], B[1..2,1..2];
início
  para I de 1 até 2 faça
    para J de 1 até 2 faça
      leia(A[I,J]);
    fim-para
  fim-para
  DET <- A [1,1] * A[2,2] - A[1,2] *A[2,1];
  se ( DET <> 0 ) então
    B [1,1] <- A[2,2];
    B [2,2] <- A[1,1];
    B [1,2] <- (-1) * A[1,2];
    B [2,1] <- (-1) * A[2,1];
    { calculo }
    para I de 1 até 2 faça
      para J de 1 até 2 faça
        B[I,J] <- ( 1 / DET ) * B[I,J];
      fim-para
    fim-para
    para I de 1 até 2 faça
      para J de 1 até 2 faça
        imprima(B[I,J]);
      fim-para
    fim-para
  senão
    imprima("A matriz não eh invertivel!");
  fim-se
fim
```

```
program L5P29;
var
  I, J : integer;
  DET : real;
  A : array[1..2,1..2] of real;
  B : array[1..2,1..2] of real;
begin
  for I := 1 to 2 do
    for J := 1 to 2 do
    begin
      write('Entre com o elemento (',I,',',J,') da matriz: ');
      readLn(A[I,J]);
    end;
  DET := A[1,1] * A[2,2] - A[1,2] * A[2,1];
  writeLn('o determinante eh:',DET:5:4);
  if ( DET <> 0) then
  begin
    B[1,1] := A[2,2];
    B[2,2] := A[1,1];
    B[1,2] := (-1) * A[1,2];
    B[2,1] := (-1) * A[2,1];
    writeLn('a inversa eh:');
    for I := 1 to 2 do
      for J := 1 to 2 do
        B[I,J] := ( 1 / DET ) * B[I,J];
    for I := 1 to 2 do
    begin
      for J := 1 to 2 do
        write(B[I,J],' ');
      writeln('');
    end;
  end
  else
    write('a matriz nao eh inversivel');
end.
```

```
for I = 1 : 2
  for J = 1 : 2
    A(I,J) = input(sprintf('(%dx%d): ',I,J));
  end
end
DET = A(1,1) * A(2,2) - A(1,2) * A(2,1);
fprintf(1,'O determinante eh: %f\n',DET);
if ( DET ~= 0 )
   B(1,1) = A(2,2);
   B(2,2) = A(1,1);
   B(1,2) = (-1) * A(1,2);
   B(2,1) = (-1) * A(2,1);
   % calculo
   for I = 1 : 2
     for J = 1 : 2
       B(I,J) = (1/DET) * B(I,J);
     end
   end
   disp('A inversa eh');
   for I = 1 : 2
     for J = 1 : 2
       fprintf(1,'%.2f\t',B(I,J));
     end
     fprintf(1,'\n');
   end
else
   disp('A matriz nao eh inversivel');
end
```

30) Criar um algoritmo que receba duas matrizes $A_{CxD}$ e $B_{ExF}$ (C, D, E e F ≤ 6). Esse algoritmo deve verificar se o produto matricial de A por B é possível (D = E). Caso seja possível, calcular o tal produto, imprimindo a matriz $G_{CxF}$ resultado.

```
algoritmo L5P30;
var
  inteiro : I, J, K;
  inteiro : C, D, E, F;
  real    : A[1..6,1..6], B[1..6,1..6], G[1..6,1..6];
início
  leia(C,D);
  leia(E,F);
  se ( C > 6 ) ou ( D > 6 ) ou ( E > 6 ) ou ( F > 6 ) então
    imprima("Erro!");
  senão
    { entrada de dados - matriz A }
    para I de 1 até C faça
      para J de 1 até D faça
        leia(A[I,J]);
      fim-para
    fim-para
    { entrada de dados - matriz B }
    para I de 1 até E faça
      para J de 1 até F faça
        leia(B[I,J]);
      fim-para
    fim-para
    se ( D = E ) então
      { calculo do produto matricial }
      para I de 1 até C faça
        para J de 1 até F faça
          G[I,J] <- 0;
          para K de 1 até D faça
            G[I,J] <- G[I,J] + A[I,K] * B[K,J];
          fim-para
        fim-para
      fim-para
      { saida de dados - matriz G }
      para I de 1 até C faça
        para J de 1 até F faça
          imprima(G[I,J]);
        fim-para
      fim-para
    senão
      imprima("Impossível calcular o produto matricial entre A e B!");
      imprima("O número de colunas de A (d) deve ser igual ao número de linhas de B (e)!");
    fim-se
  fim-se
fim
```

```
program L5P30;
var
  I, J, K : integer;
  C, D, E, F : integer;
  A, B, G : array [1..6,1..6] of real;
begin
  write('Qual a ordem da Matriz A (c x d): ');
  readLn(C,D);
  write('Qual a ordem da Matriz B (e x f): ');
  readLn(E,F);
  { entrada de dados - matriz A }
  for I := 1 to C do
    for J := 1 to D do
    begin
      write('Digite o elemento (',I,',',J,') da matriz A: ');
      readLn(A[I,J]);
    end;
  { entrada de dados - matriz B }
  for I := 1 to E do
    for J := 1 to F do
    begin
      write('Digite o elemento (',I,',',J,') da matriz B: ');
      readLn(B[I,J]);
    end;
  if ( D = E ) then
  begin
    { calculo do produto matricial }
    for I := 1 to C do
      for J := 1 to F do
      begin
        G[I,J] := 0;
        for K := 1 to D do
          G[I,J] := G[I,J] + A[I,K] * B[K,J];
      end;
    { saida de dados - matriz G }
    for I := 1 to C do
    begin
      for J := 1 to F do
        write(G[I,J]:7:4,chr(9));
      writeLn('');
    end;
  end
  else
  begin
    writeLn('Impossivel calcular o produto matricial entre A e B!');
    writeLn('O numero de colunas de A (d) deve ser igual ao numero de linhas de B (e)!');
  end;
end.
```

```
C = input('Qual o número de linhas da Matriz A (c): ');
D = input('Qual o número de colunas da Matriz A (d): ');
E = input('Qual o número de linhas da Matriz B (e): ');
F = input('Qual o número de colunas da Matriz B (f): ');
% entrada de dados - matriz A
for I = 1 : C
  for J = 1 : D
    A(I,J) = input(sprintf('Elemento (%d,%d): ',I,J));
  end
end
% entrada de dados - matriz B
for I = 1 : E
  for J = 1 : F
    B(I,J) = input(sprintf('Elemento (%d,%d): ',I,J));
  end
end
if ( D == E )
  % calculo do produto matricial
  for I = 1 : C
    for J = 1 : F
      ELEM = 0;
      for K = 1 : D
        ELEM = ELEM + A(I,K) * B(K,J);
      end
      G(I,J) = ELEM;
    end
  end
  % saida de dados - matriz G
  for I = 1 : C
    for J = 1 : F
      fprintf(1,'%7.4f\t',G(I,J));
    end
    fprintf(1,'\n');
  end
else
  disp('Impossivel calcular o produto matricial entre A e B!');
  disp('O numero de colunas de A (d) deve ser igual ao numero de linhas de B (e)!');
end
```