nina edelweiss
maria aparecida castro livi

bookman

as autoras

Nina Edelweiss é engenheira eletricista e doutora em Ciência da Computação pela Universidade Federal do Rio Grande do Sul. Durante muitos anos, lecionou em cursos de Engenharia e de Ciência da Computação na UFRGS, na UFSC e na PUCRS. Foi, ainda, orientadora do Programa de Pós-Graduação em Ciência da Computação da UFRGS. É coautora de três livros, tendo publicado diversos artigos em periódicos e em anais de congressos nacionais e internacionais. Participou de diversos projetos de pesquisa financiados por agências de fomento como CNPq e FAPERGS, desenvolvendo pesquisas nas áreas de bancos de dados e desenvolvimento de software.

Maria Aparecida Castro Livi é licenciada e bacharel em Letras, e mestre em Ciência da Computação pela Universidade Federal do Rio Grande do Sul. Desenvolveu sua carreira profissional na UFRGS, onde foi programadora e analista de sistema, antes de ingressar na carreira docente. Ministrou por vários anos a disciplina de Algoritmos e Programação para alunos dos cursos de Engenharia da Computação e Ciência da Computação. Sua área de interesse prioritário é o ensino de Linguagens de Programação, tanto de forma presencial quanto a distância.

E22a Edelweiss, Nina.
Algoritmos e programação com exemplos em Pascal e C [recurso eletrônico] / Nina Edelweiss, Maria Aparecida Castro Livi. – Dados eletrônicos. – Porto Alegre : Bookman, 2014.

Editado também como livro impresso em 2014.
ISBN 978-85-8260-190-7

1. Informática. 2. Algoritmos – Programação. I. Livi, Maria Aparecida Castro. II. Título.

CDU 004.421

Catalogação na publicação: Ana Paula M. Magnus – CRB 10/2052

7.5 → em Pascal

7.5.1 declaração de uma matriz

Um tipo de dado matriz (arranjo de mais de uma dimensão) é declarado em Pascal de forma semelhante ao tipo vetor, incluindo os limites para cada uma das dimensões:

```
array [<limite inferior>..<limite superior>,
       <limite inferior>..<limite superior>, ...] of <tipo>
```

Os limites superior e inferior, índices do primeiro e do último elemento de cada dimensão, devem ser do tipo `inteiro`, `caractere` ou `enumeração` (tipo que será visto no Capítulo 8). Não existem restrições quanto aos valores que podem ser utilizados como índices. O tipo declarado no final é o tipo de todos os elementos da matriz.

Exemplo de declarações de matrizes:

```
var
   nota: array [1..7, 1..6] of real;
   contagem_letras: array ['a'..'j', 1..20] of integer;
```

Em algumas situações, duas variáveis que tenham sido declaradas de forma independente, com tipos não definidos explicitamente e aparentemente iguais, podem não ser reconhecidas como variáveis de mesmo tipo pelo sistema. Por essa razão, recomenda-se que a declaração de uma matriz em Pascal seja feita sempre como está a seguir, definindo primeiro a(s) constante(s) usada(s) na declaração da matriz, em seguida o tipo definido para a matriz e, finalmente, a variável do tipo matriz:

```
const
  MAX1 = 6;
  MAX2 = 8;
type
  matriz = array [1..MAX1, 1..MAX2] of integer;
var
  mat: matriz;
```

7.5.2 acesso aos elementos de uma matriz

O acesso aos elementos de uma matriz em Pascal é feito definindo-se um índice para cada uma das dimensões. Esse índice pode ser dado por um valor constante, pelo conteúdo de uma variável ou pelo resultado de uma expressão, devendo ser do mesmo tipo definido para cada um dos índices, na ordem em que foram definidos. Os comandos a seguir acessam elementos das matrizes definidas na seção anterior:

```
{PREENCHER POR LEITURA O ELEMENTO DA LINHA 3, COLUNA 5 DA MATRIZ nota:}
readln(nota [3, 5]);

{SUPONDO A VARIAVEL INTEIRA i = 2, INFORMAR VALOR CONTIDO NA LINHA 1,
COLUNA 4 DA MATRIZ contagem_letras}
writeln(contagem_letras['a', i+2]);

{COPIAR VALOR DA LINHA 2, COLUNA 1 PARA LINHA 1, COLUNA 2 DE mat}
mat[1,2] := mat[2,1];
```

O trecho a seguir imprime a matriz `mat`, linha por linha, separando seus valores adequadamente. São utilizadas as variáveis inteiras `lin` e `col` para percorrer respectivamente as linhas e as colunas da matriz:

```
for lin := 1 to MAX1 do                {PARA CADA LINHA DA MATRIZ}
   begin
   writeln;          {INICIA NOVA LINHA PARA CADA LINHA DA MATRIZ}
   for col := 1 to MAX2 do          {PARA CADA COLUNA DESTA LINHA}
      write(mat[lin, col]:5, ' ')            {ESCREVE UM ELEMENTO}
   end;
```

7.5.3 inicialização de matriz na declaração

É possível inicializar matrizes na declaração, declarando-as como constantes tipadas. Constantes tipadas, apesar do nome, funcionam na realidade como variáveis, uma vez que seus valores iniciais podem ser alterados durante o processamento.

Declaração de uma matriz como constante tipada:

```
const
   <nome da constante> : <tipo da constante> =
((valor1 da 1ª linha, valor2 da 1ª linha , ... , valorn da 1ª linha),
 (valor1 da 2ª linha, valor2 da 2ª linha , ... , valorn da 2ª linha),
 (valor1 da 3ª linha, ...), (...));
```

O número de valores informados não pode ser nem menor nem maior que o número de valores previstos para cada linha da matriz, caso contrário o código não será compilado sem erros.

Exemplo de inicialização de uma matriz por meio de sua declaração como constante tipada:

```
const
   MAXMERC = 5;
   MAXMES = 12;
type
   mt = array [1..MAXMERC , 1..MAXMES] of integer;
const
    matriz : mt = ((0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1),
                   (50, 51, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5),
                   (5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5),
                   (100, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10),
                   (0, 0, 0, 0, 10, 5, 10, 5, 5, 5, 5, 5));
```

7.5.4 atribuição em bloco

Em Pascal é possível fazer uma atribuição em bloco de um arranjo a outro, desde que tenham exatamente a mesma estrutura.

Supondo que `matriz_original` e `matriz_copia` tenham o mesmo número de dimensões e, em cada uma das dimensões correspondentes (na ordem em que foram definidas), o mesmo número de elementos, a atribuição a seguir copia todos os valores de uma para a outra:

```
matriz_copia := matriz_original;
```

7.6 → em C

7.6.1 declaração de uma matriz

A declaração de uma matriz multidimensional em C é feita definindo-se, após o nome da matriz, o número de elementos em cada uma das suas dimensões, cada um entre colchetes:

```
<tipo_da_variável> <nome_da_variável>
    [<número de elementos da 1ª dimensão>]
    [número de elementos da 2ª dimensão] ...;
```

onde `<tipo_da_variável>` é o tipo de todos os elementos da matriz e `<nome_da_variável>` é o nome da matriz.

Como os índices em C iniciam obrigatoriamente em 0, os índices válidos de uma dimensão de um arranjo variam de 0 ao número de elementos dessa dimensão menos 1. A declaração em C da matriz `notas_turma` da Figura 7.3 é:

```
real notas_turma [2] [7] [6];
```

Os índices de cada uma das dimensões da matriz `notas_turma`, iniciando sempre em zero, são respectivamente: 0 a 1, 0 a 6 e 0 a 5.

7.6.2 acesso aos elementos de uma matriz

O acesso aos elementos de uma matriz em C é feito por índices, um para cada dimensão, que podem ser valores constantes, variáveis ou expressões, preferencialmente de tipo `integer` ou `char`.

No trecho a seguir, a matriz `mat_val` é apresentada linha por linha, com as variáveis `linha` e `coluna` sendo usadas respectivamente como índices de linha e coluna e as constantes `MAX1` e `MAX2` como valores máximos de índices:

```
for (linha = 0; linha < MAX1; linha++)
    {
      for (coluna = 0; coluna < MAX2; coluna++)
           printf ("%6.2f    ", mat_val[linha][coluna]);
      printf("\n");
    }
```

7.7 → dicas

usar índices fora de intervalo válido. O erro mais frequente no uso de matrizes é a tentativa de empregar índices com valores que gerem acessos fora dos limites das matrizes. Esse tipo de erro, por implicar no acesso a áreas indevidas de memória, produz por vezes resultados muito estranhos.

verificar correção de dados de entrada usados como índices. Todo dado de entrada que deva estar dentro de um intervalo válido de valores só deve ser aceito se estiver correto. Esse cuidado é particularmente importante quando o dado de entrada vier a ser usado como índice de matrizes.

escrever índices de matrizes multidimensionais em C. Na escrita dos índices de uma matriz em C, devem ser colocados pares de colchetes cercando os índices de cada dimensão. Escrever todos os índices dentro de um único par de colchetes, apenas separados por vírgulas, mesmo que não resulte em erro de sintaxe, resulta em erro lógico.

escrever índices obedecendo à ordem das dimensões das matrizes. Cuidar para referenciar corretamente cada uma das dimensões das matrizes, na ordem em que foram definidas. Por exemplo, se no exemplo da Figura 7.3 as dimensões foram definidas na ordem turmas/alunos/notas, uma referência ao elemento 1,3,2 está se referindo à 2ª nota do 3º aluno da 1ª turma. Errar a ordem dos índices resulta no acesso e utilização de elementos diferentes dos que se quer.

usar constantes. É altamente recomendável utilizar constantes nos limites das dimensões das matrizes e no código que as acessa, sempre que pertinente. Isso facilita os trabalhos de teste e torna o código mais genérico.

evitar processamento desnecessário. Um exemplo típico de processamento desnecessário é o acesso aos elementos da diagonal principal de uma matriz bidimensional. Os valores de índice desses elementos são iguais: 1 1, 2 2, 3 3, etc. Embora soluções com a variação de dois índices sejam por vezes adotadas, um só índice é suficiente para acessar seus elementos.

não expor desnecessariamente o usuário a particularidades da linguagem de programação utilizada. Por exemplo, na linguagem C os índices partem de zero. Mas, no mundo real, o usuário conhece valores que iniciam em 1 e vão até um valor limite. Nesses e em outros casos semelhantes, ao interagir com o usuário, é necessário fazer os ajustes necessários de modo a garantir que ele opere com a informação da forma como está habituado.

7.8 → testes

testar com versões reduzidas da matriz. Sempre que possível, testar os programas com a matriz reduzida a um pequeno conjunto de valores. Se funcionar para um número reduzido de elementos por dimensão, também funcionará quando esse número for aumentado.

testar com o volume mais próximo possível do real. Fazer pelo menos um teste com o volume aproximado de dados que deve ser o realmente utilizado. Há questões que só surgem com um volume maior de dados.

testar valores limite. Nos testes, verificar, sobretudo, os itens que vão até o limite dos intervalos de valores dos índices.

encontrar e solucionar erros. Verificar inicialmente a declaração da matriz. A maior parte dos problemas surge de declarações incorretas ou inconsistentes com o uso da matriz. Verificar os valores que estão sendo utilizados nos índices. Índices fora de intervalo podem provocar erros em áreas sem relação com as matrizes. Verificar se os valores de índices usados nas dimensões coincidem com o previsto na declaração da matriz.

7.9 → exercícios sugeridos

exercício 7.1 Dada uma matriz M(10, 20), escreva um programa que realize a seguinte sequência de operações:

a preencha a matriz por leitura;
b imprima seus valores na forma de uma matriz;
c procure e imprima o maior elemento de cada linha da matriz;
d calcule e imprima a média dos elementos de cada coluna;
e calcule e imprima o produto de todos os elementos diferentes de zero;
f conte e imprima quantos elementos são negativos;
g informe qual a posição ocupada (linha-coluna) por um determinado elemento cujo valor será lido pelo programa.

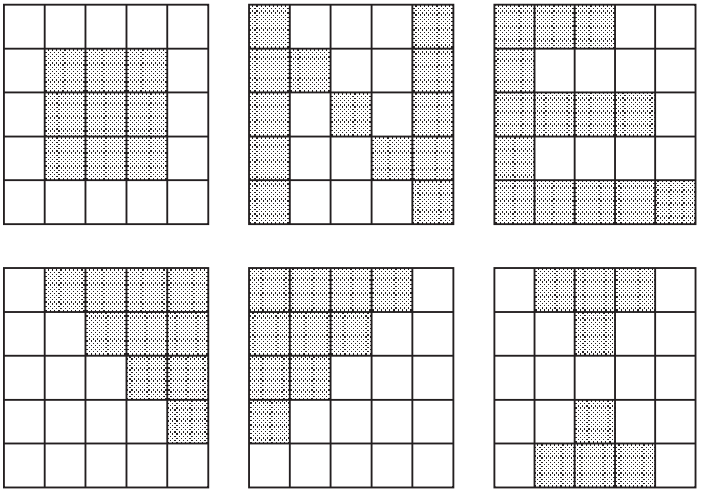
exercício 7.2 Declare uma matriz M(4, 4), de tipo `inteiro`. Sobre essa matriz efetue as seguintes operações:

a preencha por leitura;
b imprima o conteúdo na forma de uma matriz;
c troque a primeira linha da matriz com a primeira coluna e imprima novamente;
d imprima os elementos da diagonal principal em uma linha e depois os da diagonal secundária em outra linha;
e zere a segunda coluna da matriz e imprima novamente;
f preencha um vetor com o produto dos elementos de cada coluna e imprima esse vetor;
g multiplique a matriz por um valor inteiro lido e imprima novamente.

exercício 7.3 Preencha por leitura uma matriz M(10,10). Em seguida, forme um vetor com os elementos das linhas pares da matriz. Depois forme outro vetor com os elementos da diagonal principal, somados com os elementos da mesma linha, contidos na diagonal secundária. Imprima a matriz e os dois vetores.

exercício 7.4 Na Teoria de Sistemas, define-se como elemento minimax de uma matriz o menor elemento da linha em que se encontra o maior elemento da matriz. Escreva um programa que preencha uma matriz M(15,15) por leitura e determine o seu elemento minimax.

exercício 7.5 Preencha por leitura uma matriz M(5,5). Em seguida, calcule e imprima a média dos elementos das áreas assinaladas:



exercício 7.6 A transposta de uma matriz é definida trocando-se, ordenadamente, suas linhas por suas colunas, conforme mostrado no exemplo abaixo:

Matriz A

1	4
2	5
3	6

Matriz T

1	2	3
4	5	6

Faça um programa que:

- preencha por leitura uma matriz A(10, 20);
- obtenha a matriz transposta de A, chamando a transposta de matriz T;
- imprima o conteúdo de T na forma de uma matriz.

exercício 7.7 Implemente o Exercício de Fixação 7.3 (Seção 7.4), utilizando a segunda estratégia apresentada.

exercício 7.8 Faça um programa para o controle do estoque de uma rede de 5 lojas. Cada loja vende 15 produtos diferentes, com códigos de 1 a 15. O programa deve iniciar lendo o total de itens de cada um dos produtos disponível para venda em cada loja, armazenando esses estoques em uma matriz (1ª dimensão – loja; 2ª dimensão – código do produto). O programa deve processar um conjunto de atualizações de estoque decorrentes de vendas realizadas. No final do processo, o programa deve fazer uma análise do estoque que restou em cada loja, informando, por loja:

a código dos produtos que estão com estoque inferior a 10 unidades;
b número de produtos que apresentam estoque de 10 a 20 unidades (inclusive).

No final, informar o número total de itens em estoque na rede para cada produto.

exercício 7.9 Estenda o programa do exercício anterior para fazer atualização de estoques decorrentes de aquisição de produtos por parte da loja. Após preencher a matriz de estoques por leitura, o programa deve processar diversas atualizações, sendo fornecidos a cada vez a loja, o produto e o número de unidades a serem adicionadas ao estoque. Ao final das atualizações, imprima um relatório com as seguintes informações:

- total de unidades de cada produto em estoque em cada loja;
- códigos dos produtos que estão com estoque abaixo de 5 unidades, com as lojas correspondentes.

exercício 7.10 Considere a mesma rede de lojas do Exercício 7.8. Supondo que o preço de um produto possa ser diferente em cada loja, armazene os preços em uma matriz (1ª dimensão – loja; 2ª dimensão – produto). Faça um programa que:

a informe os códigos dos produtos que têm preços diferentes em pelo menos duas lojas;
b identifique e informe o código, o preço e a loja dos três produtos (diferentes entre si) mais caros;
c calcule e informe a média geral de preço de cada produto;
d para cada produto, informe as lojas em que o preço é superior à média de preço do produto.

exercício 7.11 Desenvolva um programa que, com base nas matrizes definidas nos três exercícios anteriores e já preenchidas, processe uma série de vendas. Em cada pedido feito por um cliente, ler a loja em que está sendo efetuada a compra, os códigos e as quantidades dos produtos a serem comprados. Verificar se existe estoque para atender todo o pedido na loja em questão e, caso não haja estoque de algum produto, cancelar a venda. Caso a venda possa ser realizada, fazer a atualização do estoque e informar o preço final a ser pago pelo cliente.

exercício 7.12 Uma locadora de DVDs armazena os dados referentes à quantidade de locações em um arranjo, no qual a primeira dimensão corresponde ao tipo de filme contido no DVD (comédia, drama, desenho animado, suspense, aventura, musical) e a segunda dimensão corresponde ao código do DVD (para cada tipo de filme, os códigos são sequenciais, iniciando em 1). Faça um programa que:

a processe um conjunto de locações, sendo que para cada locação são fornecidos o tipo e o código do filme, e que armazene o número de locações em uma matriz, onde a primeira dimensão é o tipo do filme e a segunda, seu código;
b processe um conjunto de pagamentos de locações, armazenando os valores arrecadados em um arranjo tridimensional no qual a primeira dimensão corresponde ao tipo do filme, a segunda ao seu código e a terceira ao código do cliente. O valor armazenado nesse arranjo será o total (em reais) arrecadado em locações, acumulado;
c forneça um relatório com os seguintes dados:

- número de locações de cada um dos tipos de filme;
- códigos dos DVDs da categoria de drama que apresentaram mais de três locações;
- códigos dos DVDs que não apresentaram locação;
- total arrecadado para cada DVD, indicando seu tipo;
- total pago por cada cliente;
- valor médio pago por cliente.

exercício 7.13 Suponha que a locadora disponha, além de DVDs, também de BDs (*Blu-Ray disk*). Adapte o programa anterior para esse caso, criando mais uma dimensão de armazenamento na qual é identificado o tipo do disco, se DVD ou BD.

exercício 7.14 Escreva um programa para gerenciar algumas ações em um restaurante. O programa deve processar as opções que estão a seguir, escolhidas de um *menu*. Após atender a uma solicitação, o *menu* deve ser reapresentado, a menos que a opção seja de terminar o programa. As opções são as seguintes:

- controle de ocupação de mesas – o restaurante tem 20 mesas. As mesas podem estar reservadas, ocupadas ou livres. Essa opção deve i) reservar uma mesa quando solicitado (qualquer mesa – o cliente não pode escolher mesa), ii) colocar o *status* de uma mesa reservada ou livre em ocupada, e iii) liberar uma mesa quando for desocupada. Nesse último caso, as despesas dessa mesa devem ser zeradas a fim de que comece o registro para a nova ocupação.
- registrar despesas de uma mesa – cada produto consumido no restaurante tem um código (inteiro, de 1 a 20). O garçom fornece esse código e a quantidade correspondente a esse código. O programa calcula a despesa (o valor unitário de cada produto deve ser lido no início do programa, uma vez só) e vai acumulando nas despesas da mesa.
- calcular valor total a ser pago por uma mesa e emitir nota fiscal – quando essa opção for solicitada, o programa deve "fechar" as despesas da mesa para a qual foi pedida a conta e imprimir o valor a ser pago. Esse valor deve ser acrescentado no total arrecadado pelo restaurante no dia em processamento.

No final do dia, o programa deve informar o total arrecadado.

exercício 7.15 Uma empresa possui duas lojas, cada uma com quatro departamentos. Cada departamento tem três funcionários. Escreva um programa que forneça à gerência algumas informações relativas aos salários dos funcionários dessa empresa. Os salários devem ser armazenados em uma matriz tridimensional (loja – departamento – identificador do funcionário). O programa deve:

a preencher a matriz por leitura;
b informar os salários de todos os funcionários, identificando qual a loja e qual o departamento em que trabalha;
c informar o total pago em salários por loja;

d informar quantos funcionários recebem salário superior a R$ 1.000,00 na primeira loja;
e informar a média salarial de cada departamento da segunda loja.

exercício 7.16 As notas de uma turma de 50 alunos são armazenadas em uma matriz `NOTA` (50, 12). Para cada aluno são armazenadas três notas para cada disciplina: colunas 1, 2 e 3 para a disciplina 1; colunas 4, 5 e 6 para a disciplina 2; colunas 7, 8 e 9 para a disciplina 3; e colunas 10, 11 e 12 para a disciplina 4. Em um vetor são armazenados os números de matrícula dos alunos, sendo que o índice de linha da matriz corresponde ao índice do vetor em que está a matrícula do aluno. Faça um programa que realize as seguintes tarefas:

a preencha por leitura a matriz e o vetor. Os dados são fornecidos da seguinte maneira: para cada aluno, seu número de matrícula, seguido de suas notas;
b calcule e informe as médias de cada aluno em cada disciplina; e
c calcule e informe as médias da turma em cada disciplina.

exercício 7.17 O controle de reservas de poltronas de um ônibus é feito através de uma matriz com 20 filas com 4 poltronas em cada fila. As poltronas ocupadas serão assinaladas na matriz por meio do valor 1 e as desocupadas, por meio de 0. Faça um programa que:

a assinale uma poltrona como ocupada, sendo fornecida sua fila e sua posição. O programa deve processar diversas reservas de lugares até que seja fornecido um sinal de final de reservas. Caso seja solicitada a reserva de um lugar que já está ocupado, informar ao usuário;
b apresente, ao final das reservas, os totais de poltronas livres e de poltronas ocupadas.

exercício 7.18 Uma loja comercializa 50 produtos diferentes, identificados por códigos de 1 a 50. Na loja atendem 5 funcionários, identificados por códigos de 1 a 5, os quais recebem 5% de comissão sobre o total vendido. Faça um programa que, tendo como base os códigos e preços dos produtos, processe um número indeterminado de vendas efetuadas ao longo de um dia. Os preços dos produtos devem ser fornecidos como dados no início do programa. Cada venda poderá ser composta de vários itens e de produtos diversos. A cada venda realizada, o número de unidades vendidas de cada produto deve ser acumulado em uma matriz, onde a 1ª dimensão corresponde aos funcionários e a 2ª aos produtos. Essa matriz deve ser inicializada com zeros no início do processamento. Além dessa matriz, outras estruturas poderão ser definidas para apoiar o processamento. O programa deverá informar o valor total a pagar em cada venda e, no final do dia, fornecer:

- total de vendas (em reais);
- total de vendas efetuadas (em reais) por cada vendedor;
- total de comissões (em reais) recebidas por cada vendedor;
- código do vendedor que efetuou a maior venda;
- código do produto que vendeu o menor número de unidades;
- total de unidades vendidas de cada produto.

exercício 7.19 Uma biblioteca possui duas salas. Cada sala tem três estantes capazes de conter, cada uma, 50 livros. Para controle dos empréstimos foram armazenados os estados de cada livro (1 – emprestado ou 0 – disponível) em uma matriz tridimensional. Faça um programa que:

a preencha a matriz por leitura (livros / estantes / salas);
b escreva os códigos referentes ao estado dos livros de cada sala em forma de matriz (uma matriz por sala);
c conte e apresente quantos livros da primeira sala estão disponíveis;
d conte e apresente quantos livros da segunda estante da segunda sala estão emprestados.