[[anterior](http://www.dm.ufscar.br/~waldeck/curso/java/part22.html), [índice](http://www.dm.ufscar.br/~waldeck/curso/java/), [seguinte](http://www.dm.ufscar.br/~waldeck/curso/java/part24.html)]

Semelhante às liguagens C, C++ e Pascal, a linguagem Java também dá suporte a vetores e matrizes (*arrays*) de diversas formas. Os vetores constituem uma forma muito conveniente de organizar informações em fileira. Por exemplo, podemos formar um vetor com as notas de cinco alunos de uma sala de aula do seguinte modo:

float nota[] = { 7.8, 8.4, 4.2, 1.8, 6.4 };

Neste caso **nota[0]** é a nota do primeiro aluno, isto é, 7.8, **nota[1]** é a nota do segundo, ou seja, 8.4, e assim por diante.

A utilização de vetores e matrizes em Java envolve três etapas:

1. **Declarar o vetor ou matriz.** Para isto, basta acrescentar um par de colchetes antes ou depois do nome da variável. Por exemplo:

int ind[];

double A[][],T[][][];

int []nota;

1. **Reservar espaço de memória e definir o tamanho**. É preciso definir o tamanho do vetor, isto é, a quantidade total de elementos que terá de armazenar. Em seguida é necessário reservar espaço de memória para armazenar os elementos. Isto é feito de maneira simples pelo operador**new**:

ind = new int[10];

nota = new int[70];

A = new double[10][20];

1. **Armazenar elementos no vetor ou matriz**. Para armazenar uma informação em um dos elementos de um vetor ou matriz, é necessário fornecer um índice que indique a posição desse elemento. Por exemplo, para armazenar um valor na quarta posição do vetor **nota**, fazemos o seguinte:

nota[3] = 5.2;

Como podemos observar, os índices começam em zero e vão até o número de posições reservadas, menos um. No vetor **nota** criado acima, os índices válidos vão de 0 até 69. Caso haja a tentativa de atribuir um valor a um elemento cujo índice esteja fora desse intervalo, ocorrerá um erro que impedirá a execução do programa. Por isso, é necessário um certo cuidado ao manejar com esses índices, garantindo o perfeito funcionamento do programa.

Existe, porém, um atalho para esses três passos quando desejamos criar um vetor com valores atribuídos de modo estático. Foi o que fizemos no primeiro exemplo acima, declarando o vetor **nota** com as notas de cinco alunos. Nesse caso, espaço suficiente para as notas de cinco alunos foi reservado e as notas foram guardadas em respectivas posições do vetor.

Entretanto, nem sempre é tão fácil assim. Em geral, estaremos interessados em trabalhar com vetores muito muito maiores, e cujos elementos sejam provenientes de outras fontes, que variam com o tempo. Assim, seremos obrigados a seguir os passos acima.

Eis mais alguns exemplos de vetores e matrizes:

// 12 primeiros termos da seqüência de Fibonacci:

long Fibonacci[] = {1,1,2,3,5,8,13,34,55,89,144};

// Tabela de sen(n\*pi/6), n=0,1,2,...5

float seno[] = {0.0000,0.5000,0.8660,1.0000,0.8660,0.5000};

// Tabela de log(1+n), n=0,2...99:

double tlog[] = new double[100];

for(int n=0; n<100; n++) tlog[i] = Math.log(1+n);

// Matriz dos coeficientes

double A[][] = { {1,2,3}, {0,1,3}, {0,0,-1} };

[[anterior](http://www.dm.ufscar.br/~waldeck/curso/java/part22.html), [índice](http://www.dm.ufscar.br/~waldeck/curso/java/), [seguinte](http://www.dm.ufscar.br/~waldeck/curso/java/part24.html)]

*Copyright ©*