**Vetores**

Vetores são estruturas que armazenam uma quantidade fixa de dados de um certo tipo. Por esta razão, esse tipo de estrutura também é conhecida como estrutura homogênea de dados.

Em um vetor, os espaços capazes de armazenar dados são indexados por índices que posteriormente servem como endereço para acesso. Essa figura representa como um vetor é armazenado na memória RAM.

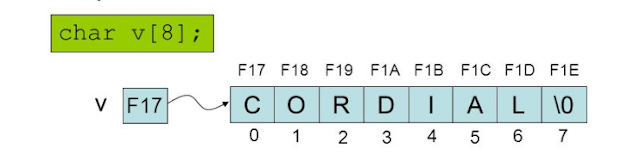
[](https://1.bp.blogspot.com/-5TqP5XsD2VA/V7JFLsdnPZI/AAAAAAAABmk/FMitr-Un0ZYGO5uT_oJLiCF7j4KbUsTZgCLcB/s1600/vetorArmazenado.PNG)

Figura 1 – Vetor armazenado na memória

Os vetores podem ser manipulados de forma direta, ou seja, é possível ler os dados contidos nele ou gravar diretamente sem nenhuma regra ou restrição. Outra característica dos vetores é que o número de posições disponíveis é igual ao tamanho definido pelo programador. Por exemplo: Um vetor declarado com tamanho 10, possuirá 10 posições e pode armazenar até 10 elementos distintos. Os índices dos vetores são formados por números inteiros, positivos, linear e sequencialmente enumerados. [ 0, 1, 2, 3, 4, … n ]. É interessante ressaltar que as linguagens C, C++, Java, entre outras, iniciam seus vetores em 0.

**2.1.       Declarando vetores**

Na declaração de vetores deverão ser fornecidas três informações: 1) o nome do vetor, 2) o número de posições do vetor (seu tamanho), e 3) o tipo de dado que será armazenado no vetor. A declaração de um vetor para “inteiros”, de nome “vetor”, em Java:

int vetor[ ];

Apesar de declarado, o vetor declarado ainda não está pronto para uso. Isto ocorre por que o vetor necessita ser alocado na memória. Esta operação é feita através do construtor chamado pela palavra reservada “new”. Observe o exemplo:

vetor = new int[10];

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| v[0] | v[1] | v[2] | v[3] | v[4] | v[5] | v[6] | v[7] | v[8] | v[9] |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

Observe a tabela:

**2.2.**       **Inicialização de vetores**

Para atribuir valores aos vetores é necessário acessar suas posições e inserir os valores. Podemos realizar esta operação da seguinte forma:

Vetor[1] = 1;

Vetor [2] = 2;

Também é possível realizar a inicialização de vetores diretamente em sua instanciação, observe o exemplo:

Int[ ] vet = {1,2,3,4,5,6,7,8};

Desta forma, o vetor instanciado automaticamente passa a ter o tamanho igual a quantidade de valores atribuídos.

**2.3.       Como percorrer vetores**

Para percorrer um vetor é necessário utilizar um laço de repetição. Os índices são variáveis inteiras e são incrementadas pelo laço de repetição. Observe o exemplo:

for( int i = 0 ; i < 0 ; i++ ) {

m[i];

}

**3.            Matrizes**

O Java não suporta nativamente vetores multidimensionais, então a solução adotada é utilizar vetores de vetores. Para declarar um vetor multidimensional em Java, deve-se seguir a seguinte sintaxe:

Int matriz [ ] [ ] = new int [ 2 ] [ 3 ];

Podemos representar esta estrutura da seguinte forma:

|  |  |  |  |
| --- | --- | --- | --- |
| Representação interna | | | |
| m[0][0] | m[0][1] | **m[0][2]** | m[0][3] |
| m[1][0] | m[1][1] | m[1][2] | m[1][3] |

**3.1.       Inicialização de matrizes**

A inicialização de matrizes e a atribuição de valores é muito semelhante a de vetores. Acompanhe o exemplo de acesso as posições de uma matriz:

Para inicialização direta de uma matriz devemos utilizar a seguinte sintaxe:

int m[][] = {{0,1,2},{2,4,5},{9,7,5}};

**3.2.       Como percorrer matrizes**

Para percorrer uma matriz podemos utilizar dois laços de repetição encadeados. Onde o mais externo irá percorrer as linhas e o mais interno irá percorrer as colunas. Observe o exemplo:

for( int i = 0 ; i < matriz.length; i++){

for ( int j = 0 ; j < matriz.length; j++){

m[i][j];

}

}

**Referências**

“Vetores em Java” Francisco, O. Disponível em: <http://www.devmedia.com.br/vetores-em-java/21449> acesso em: 15/08/2016

“Vetores e Matrizes”, Francisco, O. Disponível em: < http://www.devmedia.com.br/matrizes-vetores-em-java-parte-3/21573> acesso em: 15/08/2016