VETORES

Os vetores são estruturas de dados que permitem o armazenamento de um conjunto de dados de mesmo tipo. Por este motivo, são chamadas de estruturas homogêneas. Os vetores são unidimensionais, pois cada elemento do vetor é identificado por um índice.

Similarmente, podemos definir vetores como posições de memória, identificadas por um mesmo nome, individualizadas por índices e cujo conteúdo é de mesmo tipo.

Para acessarmos um elemento de um vetor, referimo-nos ao nome do vetor acompanhado pelo seu índice que virá entre colchetes ([e]).

Veja a sintaxe da declaração de um vetor:

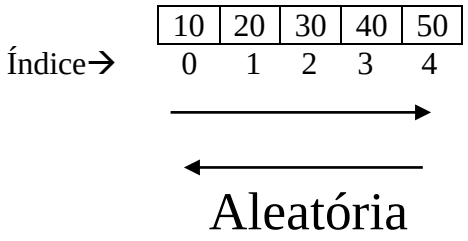
Nome do vetor : vetor [nº de elementos] de <tipo básico do vetor >

Para fazermos referência a um elemento do vetor, colocamos:

Nome do vetor [índice]

Cada elemento de um vetor é tratado como se fosse uma variável simples.

idade : vetor[5] de inteiros



 Exemplo:

Supondo que pedíssemos para criar um algoritmo para ler a idade de 5 pessoas, e mostrasse a idade na ordem inversa de leitura. A princípio, vocês pensariam em cinco variáveis: idade1, idade2, idade3, idade4 e idade5.

Veja como ficaria a solução, nesse caso:

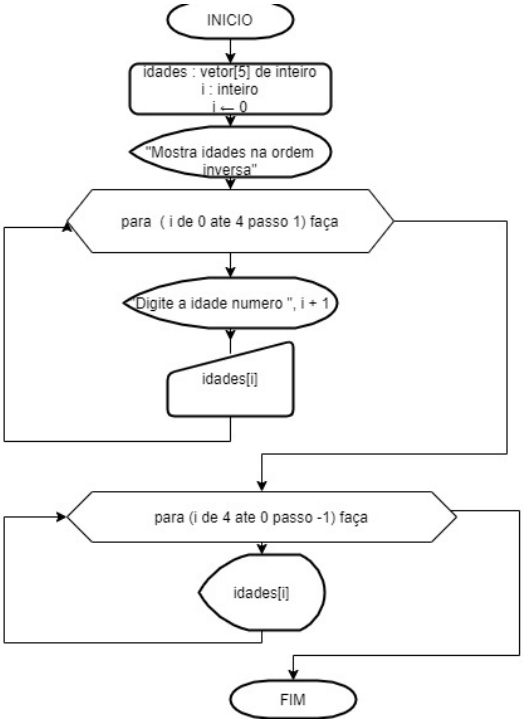
```
var
    idade1, idade2, idade3, idade4 e idade5  : inteiro;
início
    escreva  "Informe a idade de 5 pessoas: "
    leia idade1
    leia idade2
    leia idade3
    leia idade4
    leia idade5
        escreva "Ordem Inversa de Leitura "
  escreva idade5
  escreva idade4
  escreva idade3
  escreva idade2
  escreva idade1
fim
```

Assim, na memória teríamos ...

idade1	**idade**2	**idade**3	**idade**4	**idade**5
10	20	30	40	50

Todavia, se alterássemos esse algoritmo para ler a idade de 100 pessoas, a solução anterior se tornaria inviável. Para casos como este, podemos fazer uso de vetores. Se tivéssemos criado 100 variáveis, teríamos que declarar e usar: idade1, idade2, idade3, ... , idade99, idade100. Com o vetor passamos a ter: idade[1], idade[2], idade[3], idade[99], idade[100], onde a declaração do vetor se limita à linha: **nome : vetor[1..100] de inteiros**.

Veja que para todos os elementos nos referimos ao mesmo nome de vetor.



Assim, veja a solução do algoritmo anterior com o uso de vetores:

```
var
    idades : vetor[5] de inteiros
    i : inteiro
início
    para (i ← 0 até 4 passo 1 ) faça
       escreva "Informe a idade ", i+1
       leia idades[i]
    fim-para

  escreva  "Ordem Inversa de Leitura "

    para (i ← 4 até 0 passo -1) faça
       escreva idades[i]
    fim-para
fim
```

Veja a representação da memória:

Nome[1]	Nome[2]	Nome[3]	Nome[4]	Nome[5]
50	40	30	20	10

CODIFICAÇÃO EM C++

```
int main()
{
            int idades[5], i=0;
            for (i=0 ; i<=4 ; i++)
            {
                  cout <<"informe a idade"<<i+1<<"\n";
                  cin >>idades[i];
            }
            cout <<"ordem inversa de leitura";
            for (i=4 ; i>=0 ; i--)
            {
                  cout <<idades[i]<<"\n";
            }
}
```