Algoritmos e Lógica de Programação

80 horas // 4 h/semana

Vetores

Aula 12

Prof. Piva

Para começar...

- Vamos imaginar um programa para armazenar as médias finais dos 20 alunos da disciplina de Algoritmos e, em seguida mostrar todas essas médias.
- Uma variável simples, ocupando determinada posição de memória, só consegue armazenar um valor, de um mesmo tipo de dado, por vez.
- Portanto, usando variáveis simples, cada nota digitada substituirá a anterior, dentro dessa variável.
- Para solucionar esse, e outros problemas relativos ao uso de variáveis temos o VETOR, também denominado variável composta homogênea unidimensional.

Vetores

- O VETOR é uma variável composta homogênea Unidimensional.

  Composta porque é constituído de n elementos ou variáveis; Homogênea porque armazena dados de um único tipo; e Unidimensional porque é linear ou seja possui somente uma dimensão.

- Sendo o VETOR uma variável composta de n elementos, então devemos, no momento de sua definição, estabelecer o número máximo de elementos que ele irá conter.

Vetores

- Por exemplo, para armazenar as médias finais dos 20 alunos, usando variáveis simples, teríamos a seguinte definição em VisuAlg.

```
Var
    Media_AL1, Media_AL2,...,Media_AL20 : Real
```

- Usando VETOR, teríamos a seguinte definição em VisuAlg.

```
Var
    Vet_Medias_AL : VETOR [1..20] de Real
```

Vetores

- Representação simbólica da alocação do vetor Vet_Medias_AL, na memória do computador:

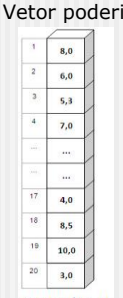
8,0	6,0	5,3	7,0	...	...	4,0	8,5	10,0	3,0
1	2	3	4	...	...	17	18	19	20

Vet_Medias_AL

- Essa representação simbólica demonstra o conceito da linearidade do Vetor.

Vetores

- A representação simbólica do Vetor poderia ser feita na forma vertical.



Vetores

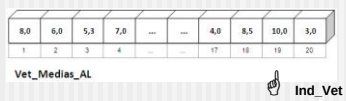
- Para o computador acessar um Vetor é preciso que ele conheça o Nome do Vetor, e um valor contido em uma Variável Índice que irá apontar para o elemento do vetor cujo conteúdo será acessado.
- O Índice de um vetor deve conter um valor numérico inteiro sem sinal, podendo ser:
  a) uma variável simples;
  b) uma constante numérica, ou mesmo;
  c) uma expressão aritmética simples, desde que esta retorne um valor numérico inteiro sem sinal.

Vetores

- Definição do Índice do vetor Vet_Medias_AL:

```
Var
    IndVet: Inteiro
```

- Supondo que Ind_Vet contenha um valor igual a 19. Com Ind_Vet apontando para o vetor Vet_Medias_AL, acessamos a média 10,0.



Vetores

- **Combinando Vetores**

Supondo que além de armazenar as médias finais dos alunos seja necessário armazenar , também, seus Nomes. Como dados Médias e Nomes são de tipos diferentes (real e cadeia de caracteres), então a solução é definir um vetor para conter os nomes – Vet_Nomes_AL, e em seguida combina-lo com o Vet_Medias_AL.

Definição dos vetores em VisuAlg:

```
Var
    Vet_Medias_AL : VETOR [1..20] de Real
    Vet_Nomes_AL : VETOR [1..20] de Caractere
```

Vetores

- O mesmo Índice que aponta para um elemento do vetor Vet_Medias_AL, apontará para o elemento correspondente no vetor Vet_Nomes_AL:

Exemplo

- Considerando que os vetores: **Vet_Medias_AL** e **Vet_Nomes_AL** - já estão alocados na memória do computador, e já possuem valores, então, para acessar e mostrar o conteúdo deles...

Exemplo - VisuAlg

```
Algoritmo "Media_Final"
Var
   Vet_Medias_AL : Vetor [1..20] de Real
   Vet_Nomes_AL : Vetor [1..20] de Caractere
    Ind_Vet : Inteiro
Inicio
   Limpatela
   Para Ind_Vet := 1 ate 20 faca
        Escreval ("Nome do Aluno(a):
   ",Vet_Nomes_AL[Ind_Vet])
        Escreval ("Média Final: ", Vet_Medias_AL[Ind_Vet])
   FimPara
   .....................................
FimAlgoritmo
```

Vetores
Em C

Variáveis Compostas Homogêneas - Unidimensionais

ALGORITMO Define.
<Nome>: VETOR [INICIOV : FIMV] DE <tipo>;

INICIO
<Comandos>
FIM

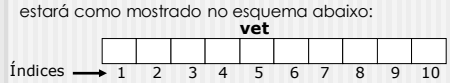
Em C:
```
Main() {
  <tipo> <nome>[qtd];

...
}
```

Vetores
Variáveis Compostas Homogêneas - Unidimensionais

Exemplo Definir uma variável indexada como sendo do tipo REAL, sendo que a mesma deverá corresponder a 10 posições de memória.

```
#include <stdio.h>
main() {
  float vet[10];
     ...
  <Comandos>
}
```

No **Exemplo** acima, após a definição da variável, a memória estará como mostrado no esquema abaixo:



Vetores
Variáveis Compostas Homogêneas - Unidimensionais

Atribuição

<Nome>[<Índice>] ← Valor;

```
exemplo: vet[2] = 3.5;
scanf("%f", &vet[2]);
```

Exemplo:

```
#include <stdio.h>
main() {
    char letras[20];
    int i=0;
    letras[0] ← 'J';
    for(i=1; i<=19; i++) {
       letras[i]=getch();
    }
}
```

EXERCÍCIOS

Algoritmos ...
vetores

EXERCÍCIO 1

Faça um algoritmo que carregue um vetor de 10 elementos numéricos inteiros. Após a finalização da entrada, o algoritmo deve escrever o mesmo vetor, na ordem inversa de entrada.

EXERCÍCIO 2

Faça um algorimto que carregue um vetor de 10 elementos numéricos inteiros. Após a finalização da entrada, o algoritmo deve escrever o maior valor e sua posição.

EXERCÍCIO 3

Faça algoritmo que carregue dois vetores de dez elementos numéricos cada um e mostre um vetor resultante na intercalação desses dois vetores

EXERCÍCIO 4

Faça um algoritmo que leia 20 palavras de no máximo 10 caracteres, e após a leitura, realize um processo qualquer que inverta os caracteres de cada uma das palavras.

Valor Aleatório (randômico)

- Nas principais linguagens de programação existem comandos específicos para gerar números aleatórios.
- Em VisuALG existe o comando RAND que retorna um valor aleatório (randômico entre 0 e 1)

RAND (exemplo)

```
algoritmo "NUMERO ALEATÓRIO"
var
 x:inteiro
 y:real
inicio
   y<-rand
   x<-int(rand*10)
   escreval("x=", x)
   escreval("y=", y)
fimalgoritmo
```

```
Início da execução
x= 7
y= 0.203424050239846

Fim da execução.
```

Note que em cada atribuição, **rand** gera um valor diferente!

EXERCÍCIO 5

Faça um Algoritmo que simule 6000 jogadas de um dado de 6 faces. Para simular o resultado utilize a função rand

Ao final, mostre a frequência de sorteio de cada uma das faces

Uma possível resposta exercício 5

```
algoritmo "NUMERO"
var
 dados: vetor[1..6] de inteiro
 x,face,i:inteiro
 y:real
inicio
   para i:=1 ate 6 faca
      dados[i]<-0
   fimpara
   i<-0
   repita
    x<-int(rand*10)
    se ((x>=1) e (x<=6)) entao
       dados[x]<-dados[x]+1
       i<-i+1
    fimse
   ate (i>=6000)

   para i:=1 ate 6 faca
      escreval("lado ",i," = ",dados[i])
   fimpara
fimalgoritmo
```

```
Início da execução
lado  1 =  1033
lado  2 =  979
lado  3 =  1045
lado  4 =  1012
lado  5 =  988
lado  6 =  943

Fim da execução.
```

EXERCÍCIO 6

Faça um algoritmo que simule a jogada de dois dados de 6 faces. O programa deve usar rand para rolar o primeiro dado e deve usar rand novamente para rolar o segundo dado. A soma das duas faces deve ser calculada. Assim: a soma variará de 2 a 12
O programa deve rolar 30.000 vezes e mostrar a frequência com que a soma (de 2 a 12) aparecem. Verifique se o valor 7 corresponde a 1/6 das jogadas!

EXERCÍCIO 7

Faça um algoritmo que armazenará os 10 primeiros números primos acima de 100. Ao final, o algoritmo deve mostrar os valores desse vetor.

EXERCÍCIO 8

Faça um algoritmo que lê 10 números inteiros e os armazena em um vetor A.
Depois de armazenado, o algoritmo fará a ordenação desses números (em ordem crescente de valores) e os colocará no vetor B
Ao final o algoritmo deve mostrar os dois vetores: A e B.