

Algoritmos e Estruturas de Dados I

Vetor

Prof. Ivre Marjorie

Introdução

- ▶ Suponhamos que é necessário armazenar um conjunto de elementos semelhantes numa estrutura de dados.
- ▶ Por exemplo, é necessário armazenar a altura de um certo número de pessoas a fim de calcular a sua média.
- ▶ É possível armazenar essas alturas em diversas variáveis, ou em um vetor.
- ▶ Ou então é possível armazenar essas alturas em um vetor.
- ▶ Um **array** é um vetor ou uma matriz que permite guardar um certo conjunto de variáveis, todas com o mesmo tipo.



Introdução

- ▶ Um vetor é uma variável, **composta**, **homogênea**, **unidimensional**

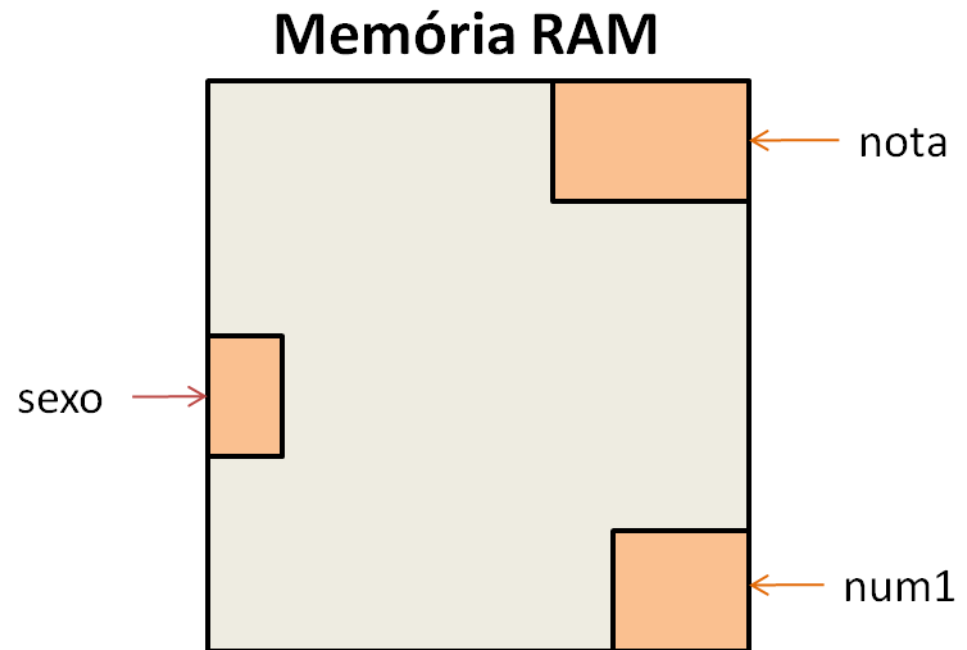
Sua representação é feita em apenas uma dimensão: uma linha ou uma coluna

Armazena mais de um valor em uma mesma variável

Os valores devem ser de um mesmo tipo, por exemplo, int, float, double ou char

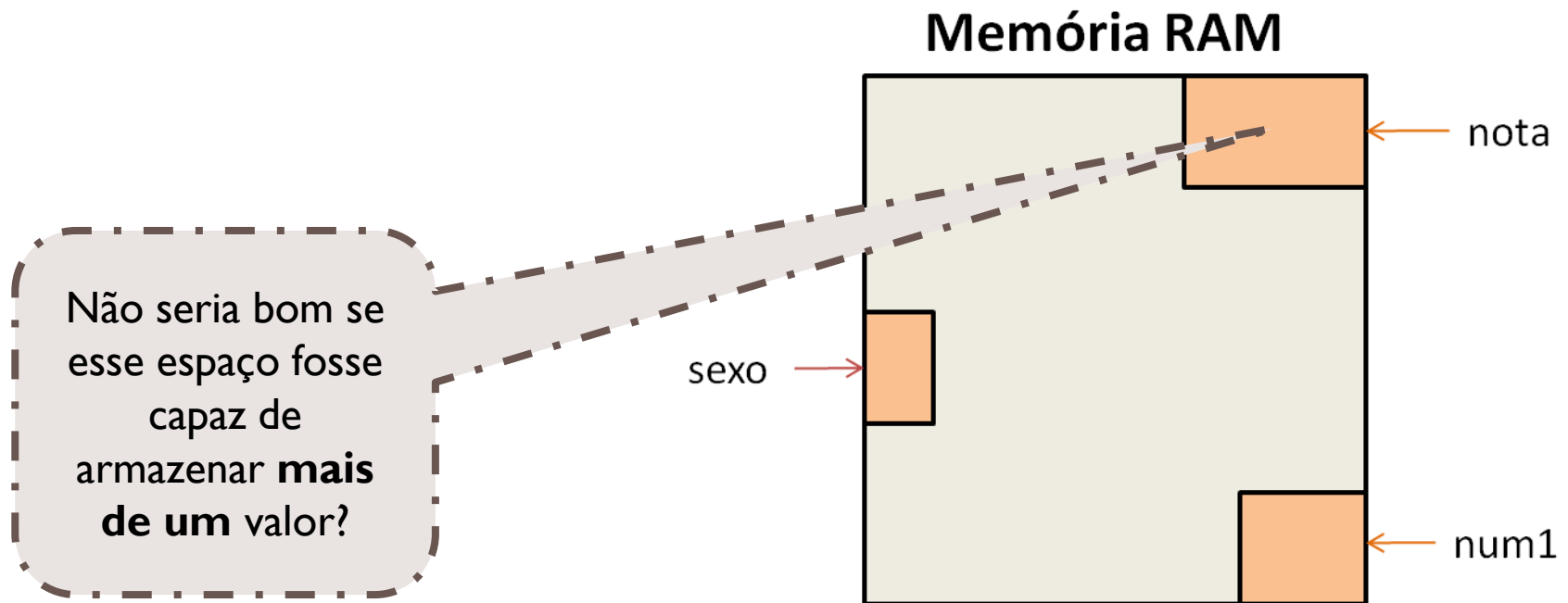
Relembrando...

- ▶ **Variável** espaço na memória capaz de armazenar um valor



Relembrando...

- ▶ Variável espaço na memória capaz de armazenar um valor



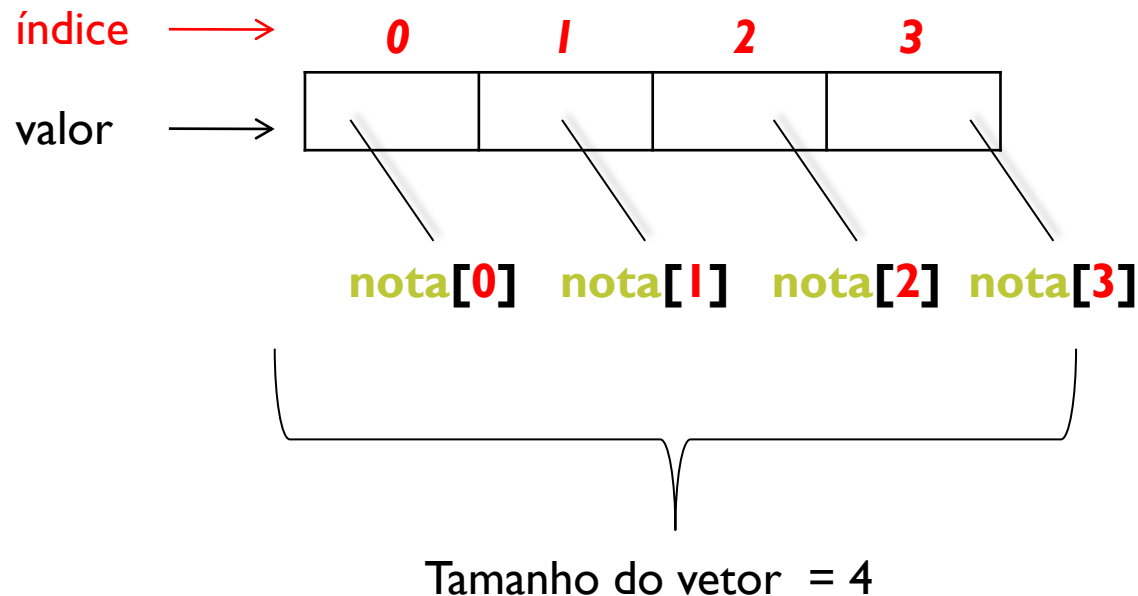
Vetor

- ▶ Esse é o objetivo do vetor
 - ▶ conseguir armazenar mais de um valor em um espaço na memória
- ▶ Podemos fazer uma analogia com as divisórias que colocamos em uma gaveta para organizá-la

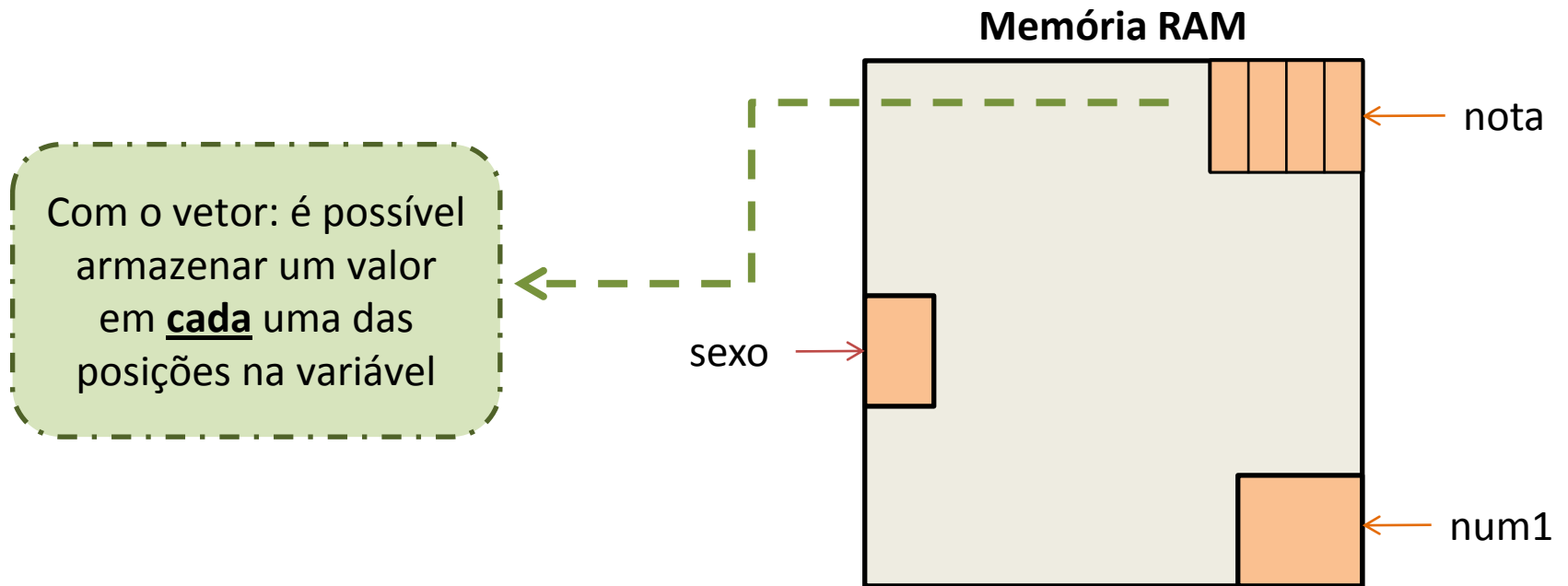


Vetor - Representação

- Considere que o vetor abaixo possui quatro posições e seu nome é nota:



Vetor - Representação



Vetor - Exemplo

int nota[4];



Vetor de inteiros

nota[0], nota[1], nota[2], nota[3]

índice	→	0	1	2	3
valor	→	20	12	3	13

Obs.: tamanho $m \Rightarrow$ índice 0 a $(m-1)$



Vetor - Declaração em C

- ▶ Declarando uma variável do tipo vetor

Tipo de Dados **nome**[**tamanho_do_vetor**];

- ▶ **Tipo de Dados**: indica do tipo de cada elemento do vetor
- ▶ **nome**: indica o nome da coleção de variáveis (seguir mesmas regras para um nome de variável qualquer)
- ▶ **tamanho_do_vetor**: indica o tamanho do vetor de elementos, onde o menor valor é 1.



Vetor - Declaração em C

Tipo de Dados **nome**[**tamanho_do_vetor**];

► Exemplos:

```
int idade[20];
```

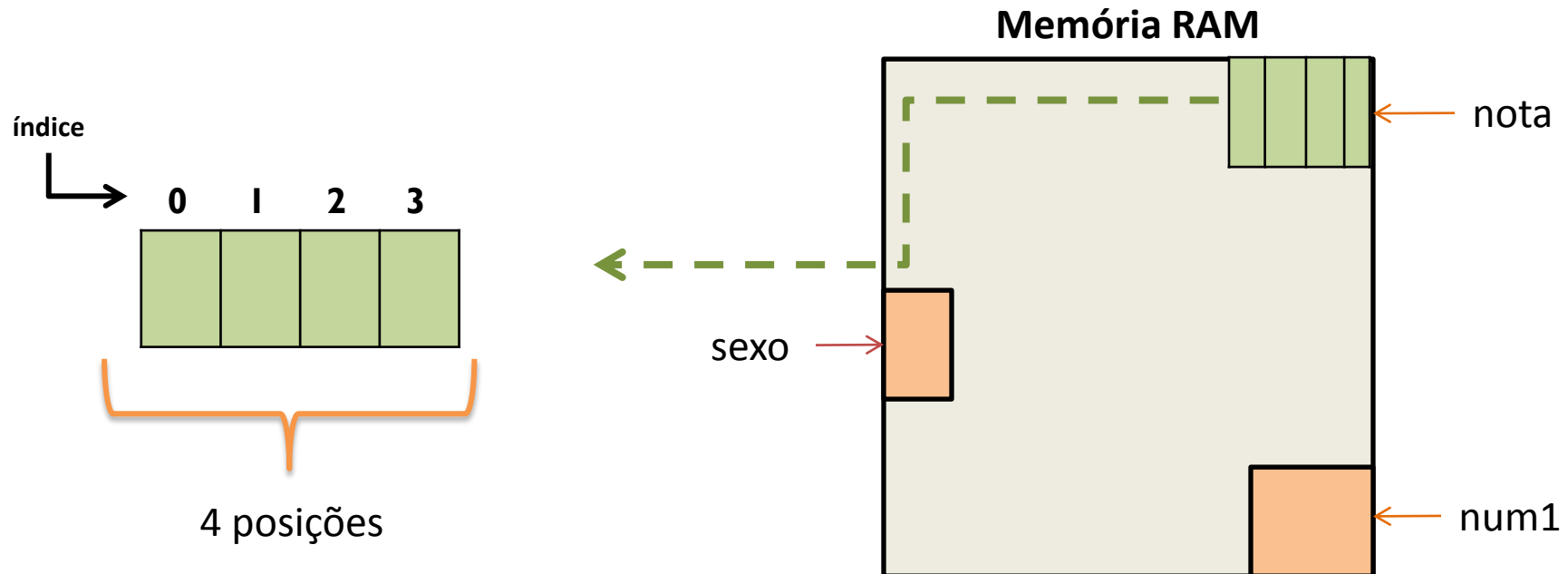
```
double peso[10];
```

```
char nome[80];
```



Vetor - Índices

- É necessário conseguir acessar cada uma das posições do vetor, para isso, são usados os índices
 - Começam sempre em ZERO e terminam em tamanho do vetor menos UM



Vetor - Acesso a valores

- ▶ Para acessar os elementos do vetor, deve utilizar o valor do índice desejado, juntamente com o nome da variável, por exemplo, `peso[2]` está associado ao terceiro elemento do vetor pois o primeiro elemento está relacionado ao índice **0**



Vetor - Atribuição de valores

- ▶ Atribui o valor 3 ao primeiro elemento do vetor

=> `vet[0] = 3;`

- ▶ Inicializando um vetor na declaração

`int valores[10] = { 20, 12, 2, 3, 6, 1, 0, 4, 7, 15 };`



Vetor - Preenchendo


- ▶ Preencher ou carregar um vetor, significa armazenar um valor em **todas** as posições do vetor
- ▶ Para isso, é necessário usar uma estrutura de repetição
 - ▶ Para percorrer todo o vetor
 - ▶ e a mais simples e indicada, é a estrutura **for**

```
for (i = 0; i < tamanho do vetor; i++)  
{  
    scanf("% tipo de dados", &nome_vetor[ i ]);  
}
```



Vetor - Preenchendo

A variável **i** começa com **ZERO**, pois o índice do vetor começa com zero



```
for (i = 0; i < tamanho do vetor; i++)  
{  
    scanf("% tipo de dados", &nome_vetor[ i ]);  
}
```

*A variável **i** é usada para identificar o índice do vetor*



Vetor - Mostrando

- ▶ Mostrar um vetor, significa mostrar na tela **todas** as posições do vetor
- ▶ Para isso, é necessário usar uma estrutura de repetição
 - ▶ Para percorrer todo o vetor
 - ▶ e a mais simples e indicada, é a estrutura **for**

```
for (i = 0; i < tamanho do vetor; i++)  
{  
    printf ("%s tipo de dados", nome_vetor[ i ]+ " | ");  
}
```



Vetor - Mostrando

A variável **i** começa com **ZERO**, pois o índice do vetor começa com zero



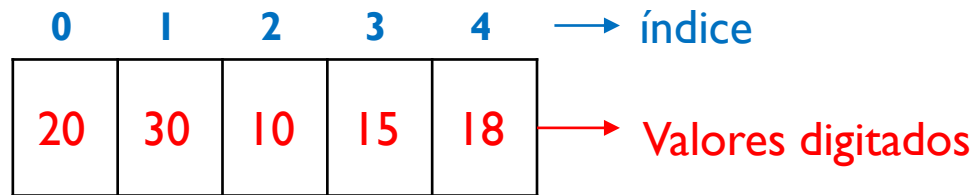
```
for (i = 0; i < tamanho do vetor; i++)  
{  
    printf ("% tipo de dados", nome_vetor[ i ]+ " | ");  
}
```

A variável **i** é usada para identificar o índice do vetor



Exemplo

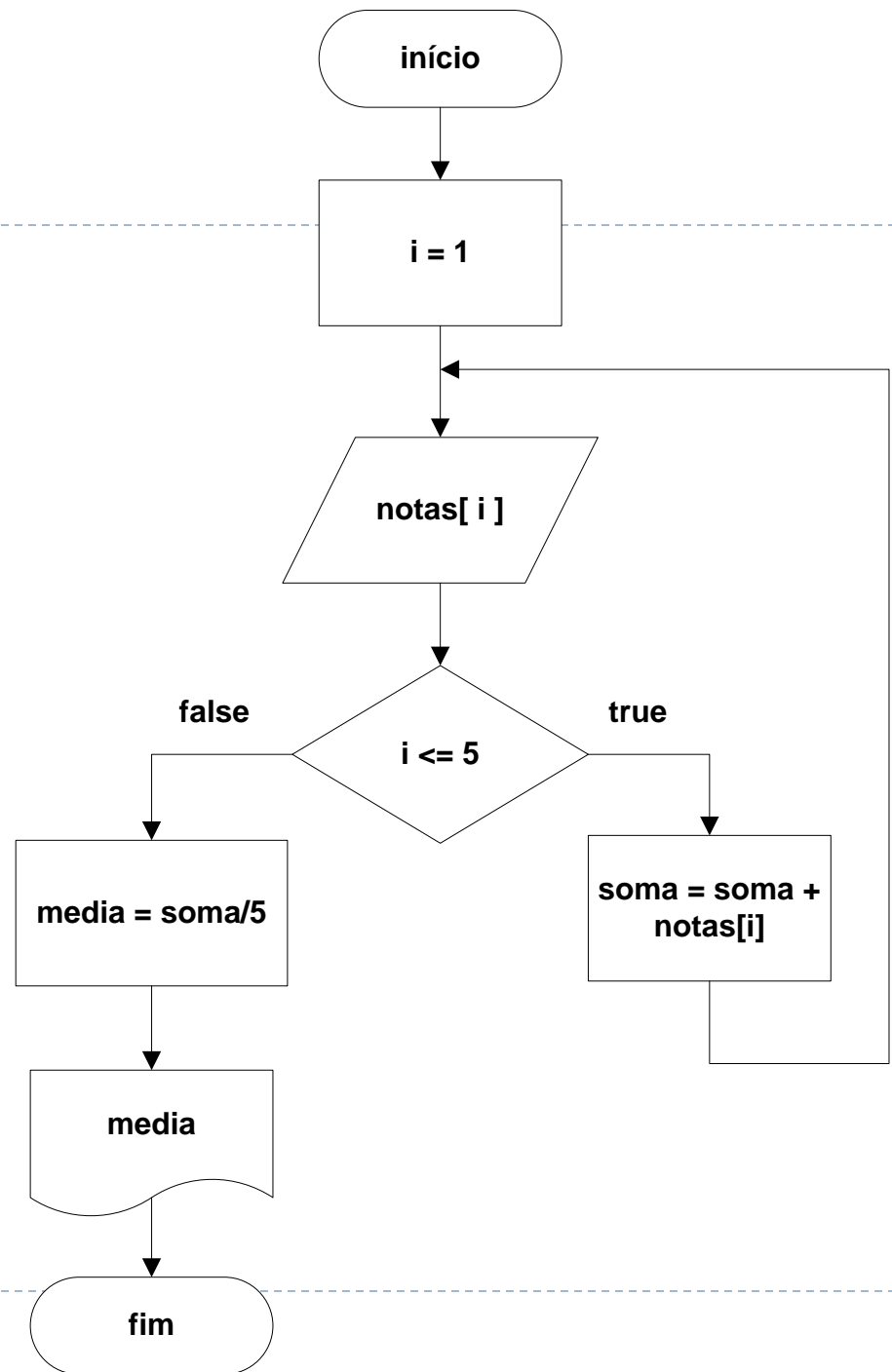
Faça um programa em C para calcular a média aritmética das notas das provas de 5 alunos. Use vetor.



Obs.: o índice do vetor vai de ZERO até 4 (tamanho -1)
O índice representa a posição no qual será armazenado o valor



Fluxograma



Código – C

```
int main()
{
    double notas[5], soma = 0, media;
    int i;
    for(i=0; i<5; i++)
    {
        printf("Digite a nota do aluno %d ", (i+1));
        scanf("%lf", &notas[i]);
        soma = soma + notas[i];
    }
    system("cls");
    printf("\n As notas dos alunos sao:");
    for(i=0; i<5; i++)
    {
        printf("\n Aluno %d: %.2lf", (i+1), notas[i]);
    }
    media=soma/5;
    printf("\n\nA media das notas e: %.2lf", media);
    return 0;
}
```



Exemplo (Resultado – Tela)

A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Users\345328\Desktop\aulaVetor\bin\Debug\aulaVetor.exe. The window has standard Windows window controls (minimize, maximize, close). The command prompt displays the following text:

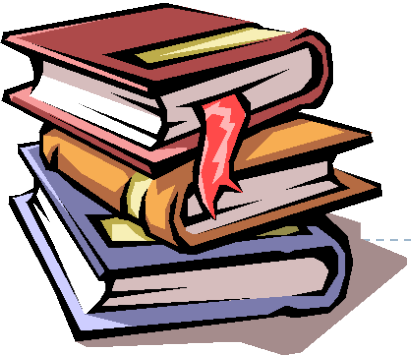
```
As notas dos alunos sao:  
Aluno 1: 2.00  
Aluno 2: 5.00  
Aluno 3: 6.00  
Aluno 4: 3.00  
Aluno 5: 4.00  
  
A media das notas e: 4.00  
Process returned 0 (0x0)   execution time : 4.136 s  
Press any key to continue.  
-
```



Exercício 1

- ▶ **Exercício 1** - Faça um programa que leia um vetor de 100 posições de números inteiros e, em seguida, mostre somente os números positivos.
 - ▶ **Exercício 2** - Em uma cidade, sabe-se que, de janeiro a fevereiro de 2012, não ocorreu temperatura inferior a **15°C**, nem superior a **40°C**. Faça um programa que armazene as temperaturas de cada dia em um vetor (de **30 posições**), calcule e imprima:
 - ▶ A menor e a maior temperatura ocorrida
 - ▶ A temperatura média
 - ▶ O número de dias nos quais a temperatura foi inferior a temperatura média
-





Referência Bibliográfica

- ▶ ASCENCIO, Ana Fernanda Gomes e CAMPOS, Edilene A.Veneruchi. **Fundamentos da Programação de Computadores: algoritmos, Pascal, C/C++ e Java** - 3ª edição. São Paulo: Pearson Prentice Hall, 2012. Capítulo 6.

