

# Problema

---

- Ler 30 valores, calcular a média aritmética dos mesmos e imprimir a média e os valores iguais ou superiores à média.
  - Quantas variáveis são necessárias para ler os valores?  
30?  
ou 1?

## **Resposta:**

30 variáveis de mesmo tipo!!

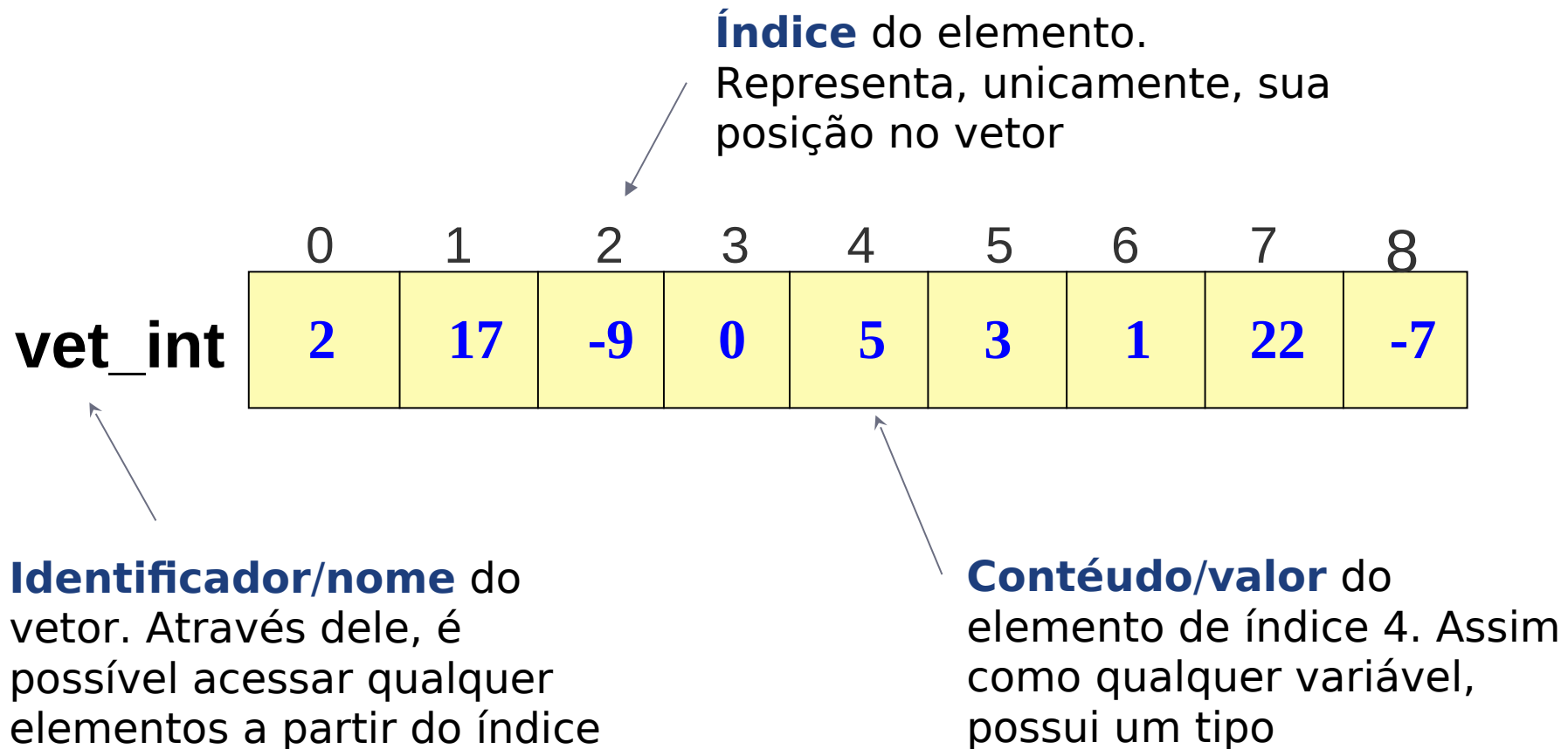
# Arranjos (Arrays)

---

- Estruturas de dados homogêneas compostas por vários elementos:
  - **homogêneas**: todos os elementos são do mesmo tipo
  - identificadas por **um só nome**, comum a todos os elementos
- Elementos:
  - ocupam **posições contíguas** de memória
  - referência: pela posição na estrutura - **índices**
- Podem ser: **unidimensionais**  
**bidimensionais**

# Arrays unidimensionais = Vetores

---



# Vetores: declaração

---

**<tipo> <identificador> [tamanho];**

- Onde:
  - **tipo**: tipo primitivo de dado (ex.: int, char, float)
  - **identificador**: nome que referencia o conjunto
  - **tamanho**: número de posições (inteiro) a serem criadas
    - primeira posição → 0
    - última posição → tamanho-1
- Exemplos:

```
int valores[10];  
float notas[5];
```

---

# Inicialização de vetores

---

**int** **numeros**[5] = { 5, 10, 15, 20, 25 };

5	10	15	20	25
---	----	----	----	----

Alternativa: **int** **numeros**[] = { 5, 10, 15, 20, 25 };

**float** **notas**[7] = { 8.1, 5.5, 9.8, 10 };

8.1	5.5	9.8	10	0	0	0
-----	-----	-----	----	---	---	---

Se nro itens inicializados  
for menor que o tamanho  
do vetor, os elementos  
não inicializados são  
**zerados**

# Inicialização e atualização de vetores

---

É possível inicializar (e atualizar) os elementos do vetor por **atribuição**:

<code>int numeros[5];</code>	<table><tr><td>?</td><td>?</td><td>?</td><td>?</td><td>?</td></tr></table>	?	?	?	?	?	O vetor ainda não foi inicializado!*
?	?	?	?	?			
<code>numeros[0] = 10;</code>	<table><tr><td>10</td><td>?</td><td>?</td><td>?</td><td>?</td></tr></table>	10	?	?	?	?	Apenas numeros[0] possui um valor válido
10	?	?	?	?			
<code>numeros[4] = 15;</code>	<table><tr><td>10</td><td>?</td><td>?</td><td>?</td><td>15</td></tr></table>	10	?	?	?	15	Apenas numeros[0] e numeros[4] possuem valores válidos
10	?	?	?	15			
<code>numeros[4] = 20;</code>	<table><tr><td>10</td><td>?</td><td>?</td><td>?</td><td>20</td></tr></table>	10	?	?	?	20	numeros[4] teve seu valor atualizado
10	?	?	?	20			

\* Para que os elementos sejam automaticamente zerados é preciso fazer algo como:

```
int numeros[5] = { };
```

# Inicialização por atribuição

```
1  #include <stdio.h>
2
3  #define TAMANHO 10
4
5  int main(){
6      int i, numeros[TAMANHO];
7
8      for(i = 0; i < TAMANHO; i++)
9          numeros[i] = i * 2;
10
11     for (i = 0; i < TAMANHO; i++)
12         printf("%d ", numeros[i]);
13
14     return 0;
15 }
```

```
0 2 4 6 8 10 12 14 16 18
```

O que há em `numeros[11]`?

O que aconteceria se:

```
for(i=0; i <= TAMANHO; i++)
    numeros[i] = 1;
```

E se:

```
for(i=0; i <= TAMANHO-1; i++)
    numeros[i] = 1;
```

# Leitura com scanf

---

```
1  #include <stdio.h>
2
3  #define TAM 50
4
5  ▼ int main(){
6      int i, vet[TAM];
7
8      printf("digite %d valores: \n", TAM);
9      for(i = 0; i < TAM; i++)
10         scanf("%d", &vet[i]);
11
12     for (i = 0; i < TAM; i++)
13         printf("Valor %d: %d\n", i+1, vet[i]);
14
15     return 0;
16 }
```



**Faça um programa que leia 30 valores, calcule a sua média aritmética e imprima a média e os valores iguais ou superiores à média**

```
1  #include <stdio.h>
2  #define MAX 30
3
4  ▼ int main(){
5      int i, soma=0, vetor[MAX];
6      double media;
7
8      printf("digite %d valores (inteiros):\n", MAX);
9  ▼  for (i=0; i<MAX; i++){
10         scanf("%d", &vetor[i]);
11         soma = soma + vetor[i];
12     }
13     media = (double)soma/MAX;
14     printf("\nMedia = %8.2lf\n", media);
15     printf("\n\nValores iguais ou superiores a media:\n");
16  ▼  for (i=0; i<MAX; i++){
17         if (vetor[i] >= media)
18             printf("%4d\n", vetor[i]);
19     }
20 }
21
```

Exercício:

Fazer teste de mesa  
(supondo MAX = 5)

# Vetores como parâmetros

- Sempre passados por referência (ponteiro)
- Passar um vetor p/ uma função significa passar um ponteiro
  - ptr armazena o endereço do 1º elemento do vetor
- No C, há 3 formas de passar um vetor como parâmetro:

```
void mostraVetor(int v[5], int n, ...);
```

```
void mostraVetor(int v[], int n ...);
```

```
void mostraVetor(int *v, int n...);
```

```
#include <stdio.h>

void le_vet(int *vet){
    int i;
    for(i=0;i<5;i++)
        scanf("%d", &vet[i]);
}

float media(int *vet){
    float med=0;
    int i;
    for(i=0;i<5;i++)
        med+=vet[i];
    med/=5;
    return med;
}

int main(){
    int numeros[5];
    float res;

    le_vet(numeros);
    res = media(numeros);
    printf("Media: %.2f\n", res);
}
```

```
1
2
3
4
5
Media: 3.00
```

## Exercícios de vetor: Faça um programa que ...

---

- 1) Leia valores para um vetor de 10 elementos. A seguir, leia um valor e imprima o número de ocorrências desse valor no vetor.
  - 2) Leia um vetor de reais (positivos ou nulos) e o escreva. A seguir, substitua os valores iguais a zero por -99 e escreva novamente o vetor.
  - 3) Leia um vetor de inteiros e o escreva. A seguir, apresente o menor valor, juntamente com sua posição no vetor.
  - 4) Leia um conjunto de 50 valores, coloque-os em 2 vetores classificando-os como pares ou ímpares. Imprimir os 2 vetores.
  - 5) Leia um vetor de 10 posições e mostre-o. A seguir, troque o primeiro elemento com o último, o segundo com o penúltimo, o terceiro com o antepenúltimo, e assim sucessivamente. Mostre o vetor após a troca.
  - 6) Leia 2 vetores A e B de tamanho 10 e os escreva. Crie, a seguir, um vetor C de 20 posições, que contenha os elementos dos vetores A e B em ordem decrescente.
-