

VETORES

Algoritmo

2

- **Algoritmo** é uma sequência finita de instruções bem definidas e não ambíguas, cada uma das quais devendo ser executadas mecânica ou eletronicamente em um intervalo de tempo finito e com uma quantidade de esforço finita. (Wikipédia)

Problema

3

- Realizar a média de cinquenta números.
- Estrutura de resolução:
 - ▣ Entrada: armazenar cada um dos cinquenta valores.
 - ▣ Processamento: soma de todos os valores e divisão.
 - ▣ Saída: impressão da média na tela.

Problema

4

- Realizar
- Estruturar
 - ▣ Entradas
 - ▣ Processamento
 - ▣ Saída:

```
#include <stdio.h>

int main(void)
{
    //DECLARANDO 50 VARIÁVEIS.
    float V0, V1, V2, V3, V4, V5, V6, V7, V8, V9;
    float V10, V11, V12, V13, V14, V15, V16, V17, V18, V19;
    float V20, V21, V22, V23, V24, V25, V26, V27, V28, V29;
    float V30, V31, V32, V33, V34, V35, V36, V37, V38, V39;
    float V40, V41, V42, V43, V44, V45, V46, V47, V48, V49;

    float soma = 0, media = 0;

    //LENDO 50 VALORES
    printf("Insira uma nota: ");
    scanf("%f", &V0);

    printf("Insira uma nota: ");
    scanf("%f", &V1);

    /*
       Continuação da leitura...
    */

    printf("Insira uma nota: ");
    scanf("%f", &V48);

    printf("Insira uma nota: ");
    scanf("%f", &V49);

    //CÁLCULO DA SOMA E DA MÉDIA DE 50 VARIÁVEIS.
    soma = V0 + V1 + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9;
    soma += V10 + V11 + V12 + V13 + V14 + V15 + V16 + V17 + V18 + V19;
    soma += V20 + V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 + V29;
    soma += V30 + V31 + V32 + V33 + V34 + V35 + V36 + V37 + V38 + V39;
    soma += V40 + V41 + V42 + V43 + V44 + V45 + V46 + V47 + V48 + V49;

    media = soma / 50.0;

    //IMPRIMINDO O VALOR DA MÉDIA
    printf("Media = %f", media);

    return 0;
}
```

a valores.
e divisão.

Problema

```
#include <stdio.h>
```

5

```
int main(void)
```

```
{
```

```
    //DECLARANDO 50 VARIÁVEIS.
```

```
    float V0, V1, V2, V3, V4, V5, V6, V7, V8, V9;
```

```
    float V10, V11, V12, V13, V14, V15, V16, V17, V18, V19;
```

```
    float V20, V21, V22, V23, V24, V25, V26, V27, V28, V29;
```

```
    float V30, V31, V32, V33, V34, V35, V36, V37, V38, V39;
```

```
    float V40, V41, V42, V43, V44, V45, V46, V47, V48, V49;
```

```
    float soma = 0, media = 0;
```

```
    //LENDO 50 VALORES
```

```
    printf("Insira uma nota: ");
```

```
    scanf("%f", &V0);
```

```
    printf("Insira uma nota: ");
```

```
    scanf("%f", &V1);
```

```
    /*
```

Problema

6

```
/*  
    Continuação da leitura...  
*/
```

```
printf("Insira uma nota: ");  
scanf("%f", &V48);
```

```
printf("Insira uma nota: ");  
scanf("%f", &V49);
```

```
//CÁLCULO DA SOMA E DA MÉDIA DE 50 VARIÁVEIS.
```

```
soma = V0 + V1 + V2 + V3 + V4 + V5 + V6 + V7 + V8 + V9;
```

```
soma += V10 + V11 + V12 + V13 + V14 + V15 + V16 + V17 + V18 + V19;
```

```
soma += V20 + V21 + V22 + V23 + V24 + V25 + V25 + V27 + V28 + V29;
```

```
soma += V30 + V31 + V32 + V33 + V34 + V35 + V36 + V37 + V38 + V39;
```

```
soma += V40 + V41 + V42 + V43 + V44 + V45 + V46 + V47 + V48 + V49;
```

```
media = soma / 50.0;
```

```
//IMPRIMINDO O VALOR DA MÉDIA
```

```
printf("Media = %f", media);
```

```
return 0;
```

```
}
```

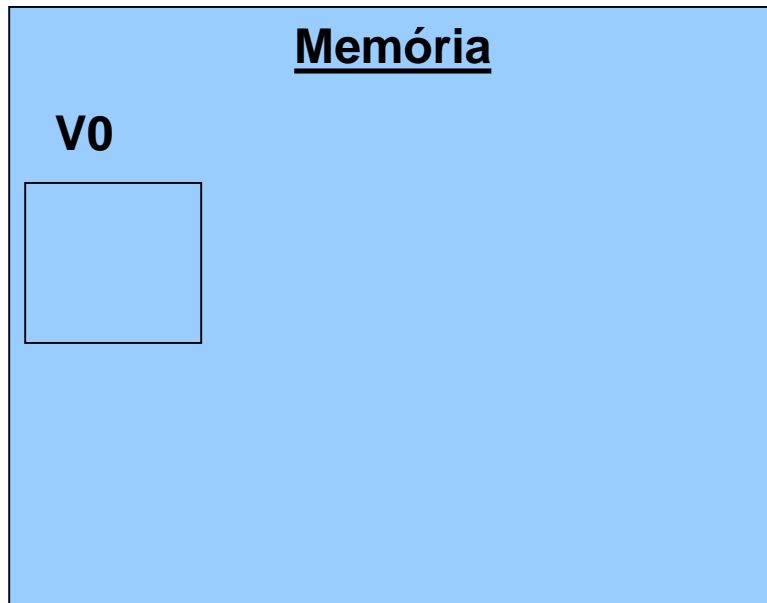
es.

).

Conceito de variável

7

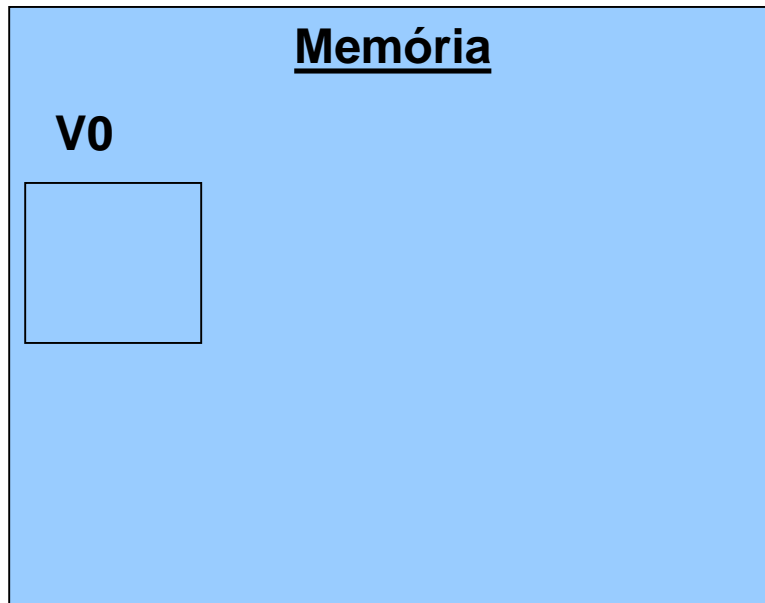
- ❑ Declarar uma variável equivale a reservar um espaço na memória temporariamente.
- ❑ Exemplo: float V0;



Conceito de variável

8

- ❑ Declarar uma variável equivale a reservar um espaço na memória temporariamente.
- ❑ Exemplo: **float** V0;

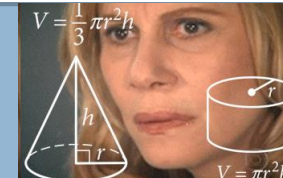


Só armazena **UM VALOR** de **ponto flutuante**;
Possui quantidade exata de bytes para o tipo de dado;

Conceito de variável

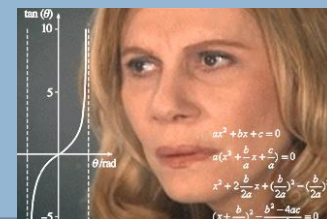
9

- Declarar uma variável equivale a reservar um espaço
- Exemp



O QUE FAZER PARA ARMAZENAR
MAIS DE UM VALOR COM O
MESMO NOME DE VARIÁVEL?

v0



de ponto

e bytes

Estruturas de dados

10

- Modo de armazenar e manipular dados na memória de um computador.

Estruturas de dados

11

- Modo de armazenar e manipular dados na memória de um computador.
- Estruturas de dados homogêneos:
- Estrutura de dados heterogêneos:

Estruturas de dados

12

- Modo de armazenar e manipular dados na memória de um computador.
- Estruturas de dados homogêneas:
 - ▣ Mesmo tipo de dado agrupado.
 - ▣ Variável indexada (utilizam índices para diferenciar).
 - ▣ Vetores, Strings e Matrizes.
- Estrutura de dados heterogêneas:

Estruturas de dados

13

- Modo de armazenar e manipular dados na memória de um computador.
- Estruturas de dados homogêneas:
 - ▣ Mesmo tipo de dado agrupado.
 - ▣ Variável indexada (utilizam índices para diferenciar).
 - ▣ Vetores, Strings e Matrizes.
- Estrutura de dados heterogêneas:
 - ▣ Diferentes dados agrupados.
 - ▣ Registros (structs em C).

Conceito de vetor

14

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.

- ❑ Agrupamento de variáveis do mesmo tipo, identificadas por índices.

Conceito de vetor

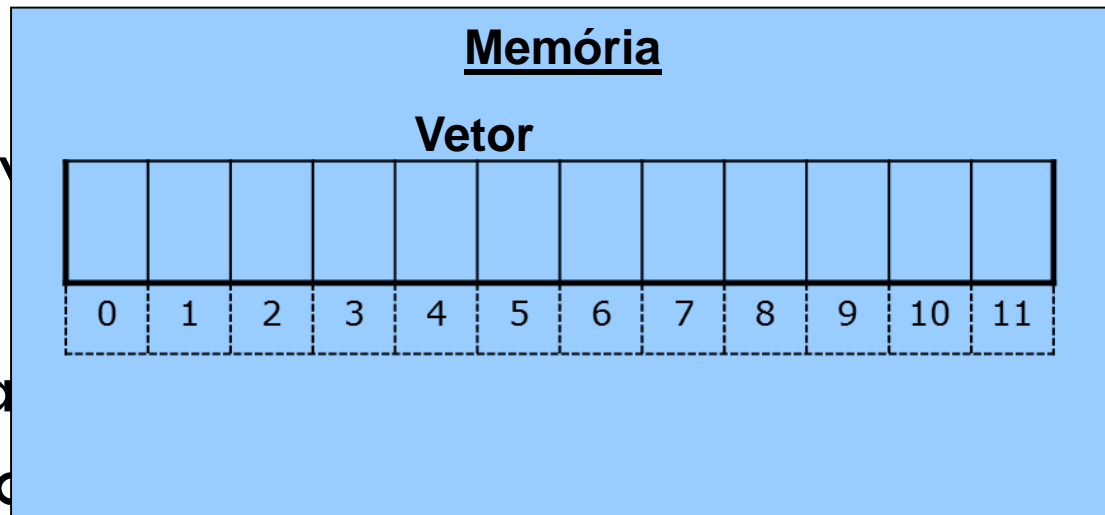
15

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.

- ❑ Espaço

- ❑ Vários y

- ❑ Agrupa
identific



ado.

Conceito de vetor

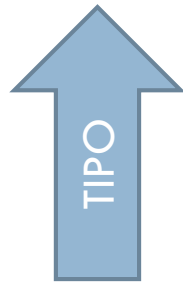
16

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`

Conceito de vetor

17

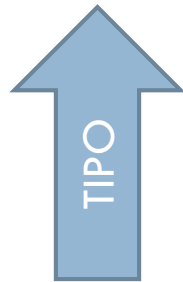
- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`



Conceito de vetor

18

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores de mesmo tipo.
- ❑ Exemplo: `int Meses[12];`

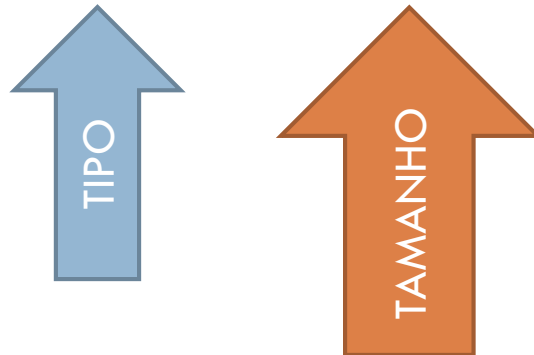


Conceito de vetor

19

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores de mesmo tipo.

❑ Exemplo: `int Meses[12];`

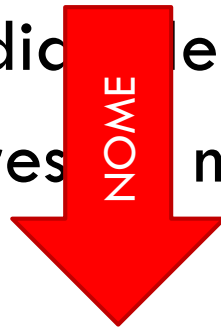
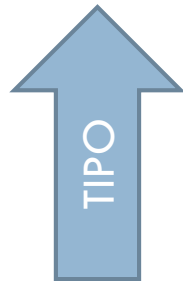


Conceito de vetor

20

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores de mesmo tipo.

❑ Exemplo: `int Meses[12];`

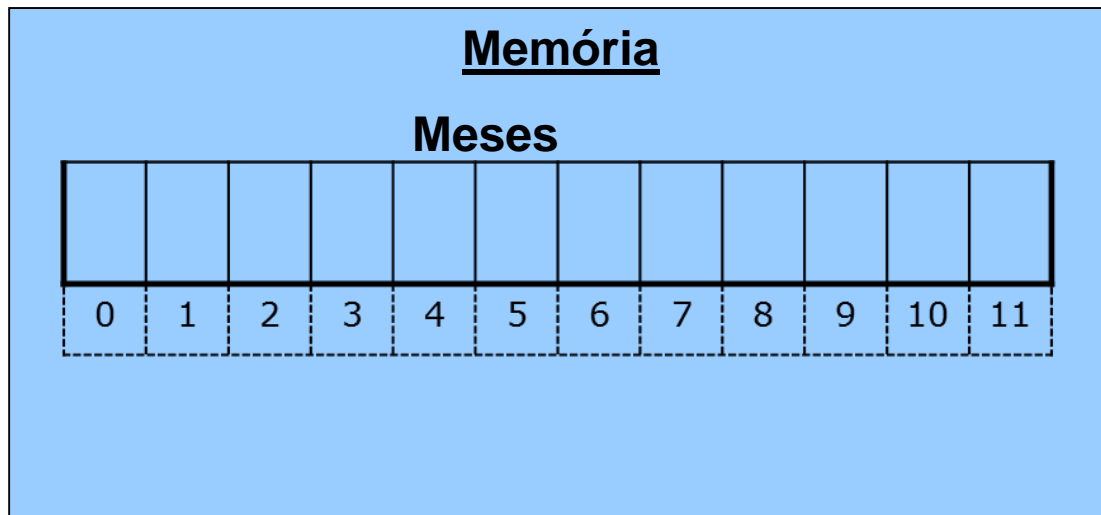


TAMANHO: quantidade
máxima de valores do
mesmo tipo.
Capacidade do vetor

Conceito de vetor

21

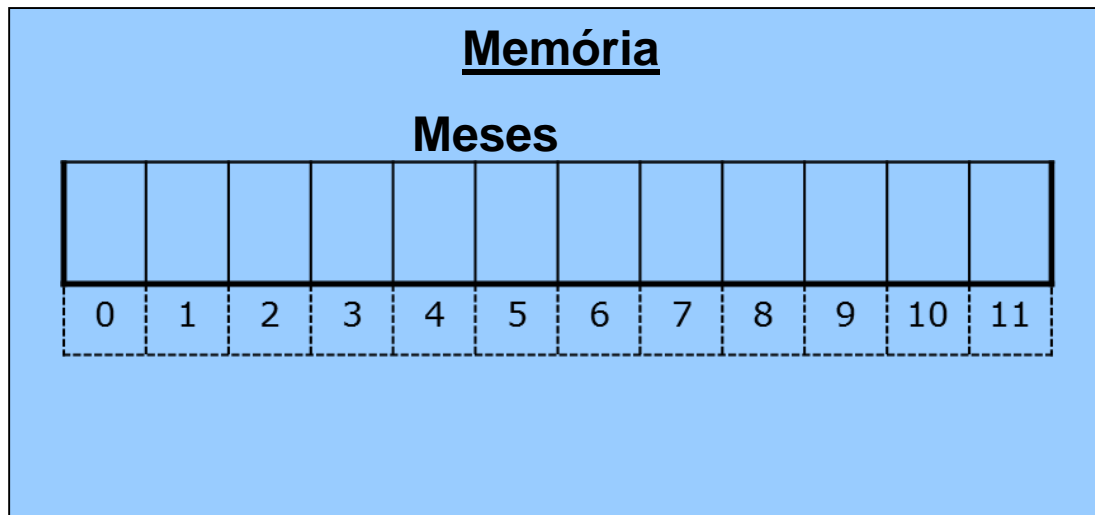
- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`



Conceito de vetor

22

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`

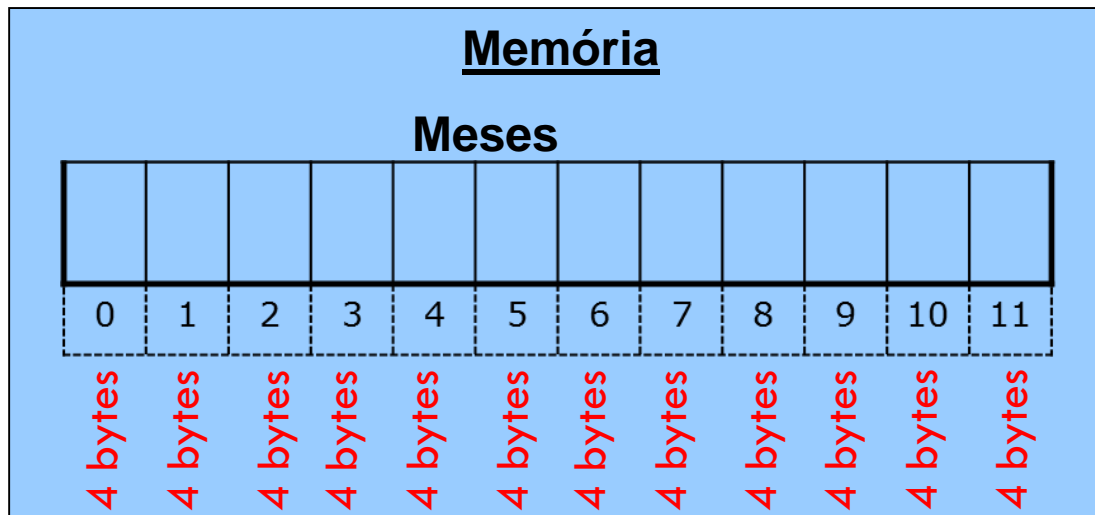


Só armazena **inteiro**;

Conceito de vetor

23

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`

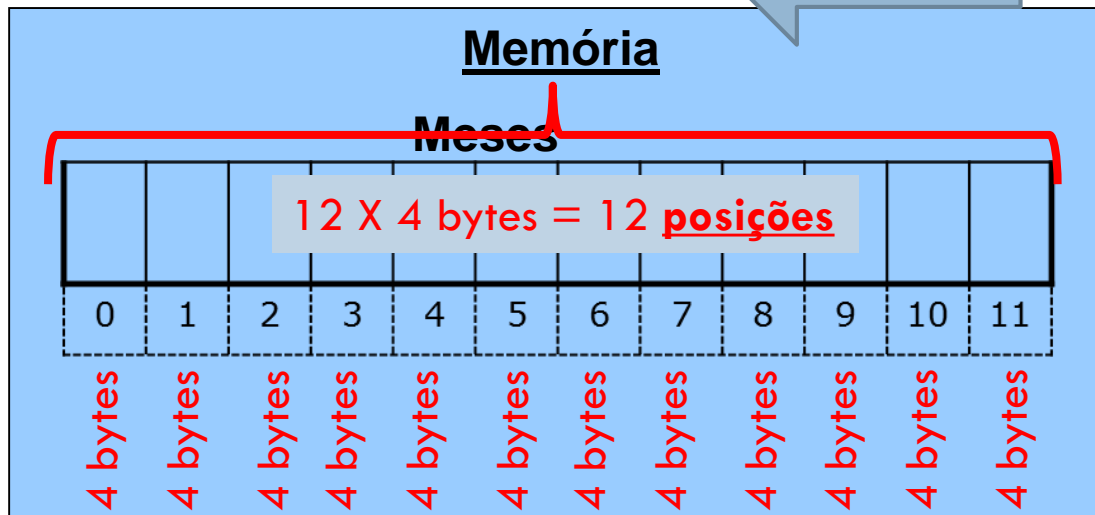


Só armazena inteiro;
Possui quantidade para armazenar N vezes os bytes para o tipo de dado;

Conceito de vetor

24

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12]`.

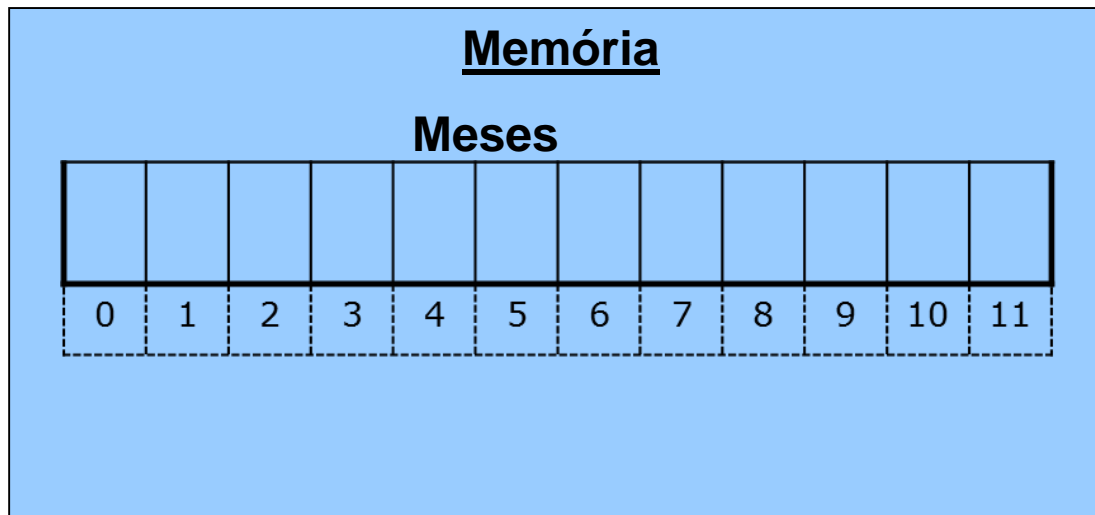


Só armazena inteiro;
Possui quantidade para
armazenar N vezes os bytes
para o tipo de dado;

Conceito de vetor

25

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`

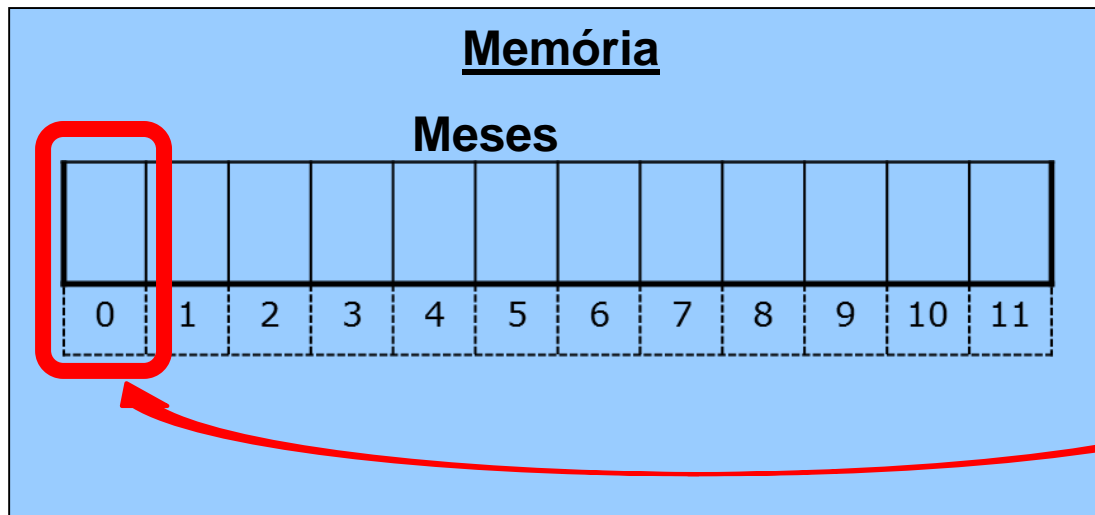


Só armazena inteiro;
Possui quantidade para
armazenar N vezes os bytes
para o tipo de dado;
Acessado por índices
(**Meses[0]**, Meses[1]...);

Conceito de vetor

26

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`

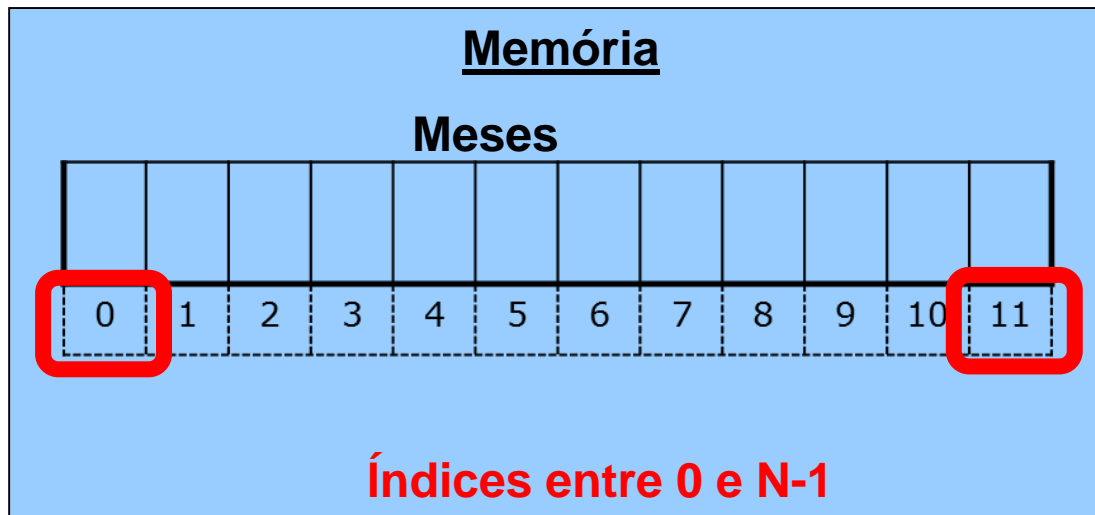


Só armazena inteiro;
Possui quantidade para
armazenar N vezes os bytes
para o tipo de dado;
Acessado por índices
(**Meses[0]**, Meses[1]...);

Conceito de vetor

27

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`



Só armazena inteiro;
Possui quantidade para armazenar N vezes os bytes para o tipo de dado;
Acessado por índices
(**Meses[0]**, Meses[1]...);

Conceito de vetor

28

- ❑ Declarar um vetor equivale a reservar um espaço na memória temporariamente.
- ❑ Espaço dividido de acordo com o tipo de dado.
- ❑ Vários valores do mesmo tipo.
- ❑ Exemplo: `int Meses[12];`



Só armazena inteiro;
Possui quantidade para armazenar N vezes os bytes para o tipo de dado;
Acessado por índices
(**Meses[0]**, Meses[1]...);

Conceito de vetor

29

- Armazenar e recuperar um valor **a cada posição**;
- `int vetor[10];` //declarar
- `vetor[0] = 20;` //atribuir valor
- `int indice = 8;`
- `vetor[indice] = 29;`

- `printf("%d", vetor[0]);` //imprimir na tela
- `scanf("%d", &vetor[3]);` //ler e armazenar

Conceito de vetor

30

- Armazenar e recuperar um valor **a cada posição**;
- `int vetor[10];`
- `vetor[0] = 20;`
- `int indice = 8;`
- `vetor[indice] = 29;`
- `printf("%d", vetor[0]);`
- `scanf("%d", &vetor[3]);`

SEMPRE INFORMAR O ÍNDICE
DO VETOR!

//imprimir na tela

//ler e armazenar

Conceito de vetor

31

```
int main(void)
{
    int n = 5;
    int a[n];

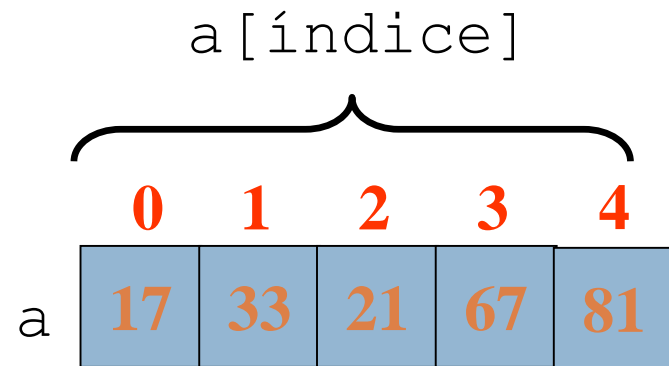
    a[0] = 17;
    a[1] = 33;
    a[2] = 21;
    a[3] = 67;
    a[4] = 81;
}
```

Conceito de vetor

32

```
int main(void)
{
    int n = 5;
    int a[n];

    a[0] = 17;
    a[1] = 33;
    a[2] = 21;
    a[3] = 67;
    a[4] = 81;
}
```



Conceito de vetor

33

- Inicialização na declaração:

- ▣ Para variáveis:

- ```
int VAR = 20;
```

# Conceito de vetor

34

- Inicialização na declaração:

- ▣ Para variáveis:

- ```
int VAR = 20;
```

- ▣ Para vetores:

- ```
int VETOR[5] = {0, 10, 2, 30, 4};
```

# Conceito de vetor

35

## □ Inicialização na declaração:

### ▣ Para variáveis:

```
int VAR = 20;
```

### ▣ Para vetores:

```
int VETOR[5] = {0, 10, 2, 30, 4};
```

VETOR[4] TERÁ VALOR 4  
VETOR[1] TERÁ VALOR 10

# Questionário

36

- 1 – Defina o que é vetor, na computação, com suas palavras.
- 2 – Qual a faixa de índices para um vetor de  $X$  posições?
- 3 – Escreva um exemplo utilizando `scanf` e `printf`, para ler e imprimir uma determinada posição  $i$  do vetor.
- 4 – Declare e inicialize um vetor de 3 posições.

# Conceito de vetor

37

- Elementos de vetor possuem todas as características de uma variável comum.
  - ▣ Podem ser usados em expressões e atribuições.

# Conceito de vetor

38

- Elementos de vetor possuem todas as características de uma variável comum.
  - ▣ Podem ser usados em expressões e atribuições.

```
int i = 2, j = 3;
```

```
int Vet[6] = {4, 8, 15, 16, 23, 42}
```

```
Vet[i] = 11;
```

```
Vet[5] = Vet[1] * vet[i + j];
```

# Conceito de vetor

39

- Elementos de vetor possuem todas as características de uma variável comum.
  - ▣ Podem ser usados em expressões e atribuições.

```
int i = 2, j = 3;
```

```
int Vet[6] = {4, 8, 15, 16, 23, 42}
```

```
Vet[i] = 11;
```

```
Vet[5] = Vet[1] * vet[i + j];
```

**5 - Como ficarão os valores de Vet após a operação?**

# Conceito de vetor

40

- Características do vetor:
  - ▣ **Alocação estática:** as dimensões (tamanho/capacidade) dos vetores devem ser conhecidas no momento da declaração.
  - ▣ **Estrutura homogênea:** dados do conjunto são do mesmo tipo.
  - ▣ **Alocação sequencial de memória:** bytes contíguos, posições na memória seguem os índices.



# Exemplos

41

```
#include <stdio.h>

int main(void)
{
 float valor[50];
 float soma = 0, media = 0;
 int i;

 for(i = 0; i < 50; i++)
 {
 printf("Insira uma nota: ");
 scanf("%f", &valor[i]);
 soma = soma + valor[i];
 }

 media = soma / 50.0;

 printf("Media = %.1f", media);

 return 0;
}
```

# Exemplos

42

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
 float valor[50];
```

```
 float soma = 0, media = 0;
```

```
 int i;
```

```
 for(i = 0; i < 50; i++)
```

```
 {
```

```
 printf("Insira uma nota: ");
```

```
 scanf("%f", &valor[i]);
```

```
 soma = soma + valor[i];
```


```
 }
```

```
 media = soma / 50.0;
```

```
 printf("Media = %.1f", media);
```

```
 return 0;
```

```
}
```



Declarando vetor de 50 posições;  
Espaço para 50 valores;  
Equivalente a 50 variáveis;

# Exemplos

43

```
#include <stdio.h>

int main(void)
{
 float valor[50];
 float soma = 0, media = 0;
 int i;

 for(i = 0; i < 50; i++)
 {
 printf("Insira uma nota: ");
 scanf("%f", &valor[i]);
 soma = soma + valor[i];
 }

 media = soma / 50.0;

 printf("Media = %.1f", media);

 return 0;
}
```

Lendo o valor para cada **ÍNDICE**;  
Cada valor em uma **posição**;  
Variar i para acessar cada posição;

# Exemplos

44

```
#include "stdio.h"

int main (void)
{
 int i;
 //declaração
 int A[10], C[10];
 //atribuição
 A[0]=15;
 A[1]=20;

 //atribuição por entrada de dados
 for (i=2; i<10; i++)
 scanf("%d", &A[i]);

 //atribuição por operação de valor de outro vetor
 for (i=0; i<10; i++)
 C[i] = A[i] * 2;

 //escrita
 for (i=0; i<10; i++)
 printf("%d\t", C[i]);
}
```

# Exercício

45

- Escreva um código em C que:
  - ▣ Leia um valor informado pelo usuário;
  - ▣ Declare um vetor float com o tamanho informado;
  - ▣ Preencha o vetor com valores informados;
  - ▣ Divida todos os valores por 2;
  - ▣ Imprima os valores do vetor.