

# Vetores

SSC0304 - Introdução à Programação para  
Engenharias

# Pense na seguinte situação ...

- Imagine que desejamos armazenar as notas de 10 alunos.
- Com o nosso conhecimento até agora, faríamos:

```
int main() {  
    float a1,a2,a3,a4,a5,a6,a7,a8,a9,a10;  
    printf("\n Favor inserir 10 números reais:");  
    scanf("%f %f %f %f ...", &a1, &a2, ..., &a10);  
    ...  
}
```

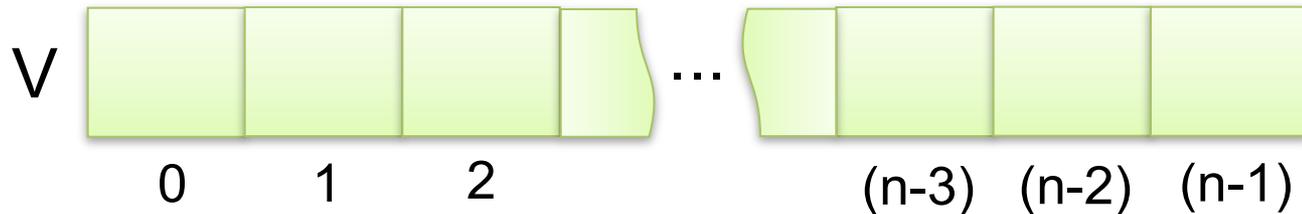
- Como proceder se tivéssemos 100 ou 200 alunos?
- Percebe o problema da escalabilidade do programa!?

# Tipos compostos

- O que precisamos é de um tipo de dados composto, capaz de armazenar não somente um valor, mas sim um conjunto de valores.
- Como resposta a isso temos:
  - Estrutura com dados homogêneos:  
Vetores e Matrizes
  - Estrutura com dados heterogêneos:  
Registros (Structs)

# Vetores (Arrays)

- Vetores são agrupamentos contíguos de dados em memória, capazes de armazenar um valor em cada uma de suas posições.
- Cada elemento do vetor é identificado de maneira exclusiva por intermédio de um índice seqüencial, iniciando em 0 (0,1,2,...,n-1).



# Vetores (Arrays)

- Declaração:

`tipo_dado nome_vetor[<tamanho>];`

define um arranjo de <tamanho> elementos adjacentes na memória do tipo *tipo\_dado*.

- Ex: `float m[10];`                    */\* m contém 10 posições reais \*/*  
`int v[3], k[5], i;`                */\* declara dois vetores, v e k\*/*  
`char nome[30];`                    */\* uma string \*/*

- Ex: */\* declaração e inicialização\*/*  
`int x[3] = { 10, 20, 30 };`  
`int y[] = { 10, 20, 30, 40, 50 };`

# Referenciando Elementos

- Referenciamos os elementos do array através do seu nome e, entre colchetes, o índice da posição desejada:
- Ex: `m[5] = 5.5;`  
`if (m[3] == 5.5)`  
 `printf("Êxito");`  
`else`  
 `printf("Falha");`

# Exemplo

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

```
#define SIZE 7          /* define uma constante ... aqui,  
                        representa o tamanho do array */
```

```
int main () {
```

```
    int i;
```

```
    int arint[7] = {10, 20, 30, 40, 50, 60, 70};
```

```
    for (i=0; i < 7; i++)
```

```
        printf("%d ", arint[i] );
```

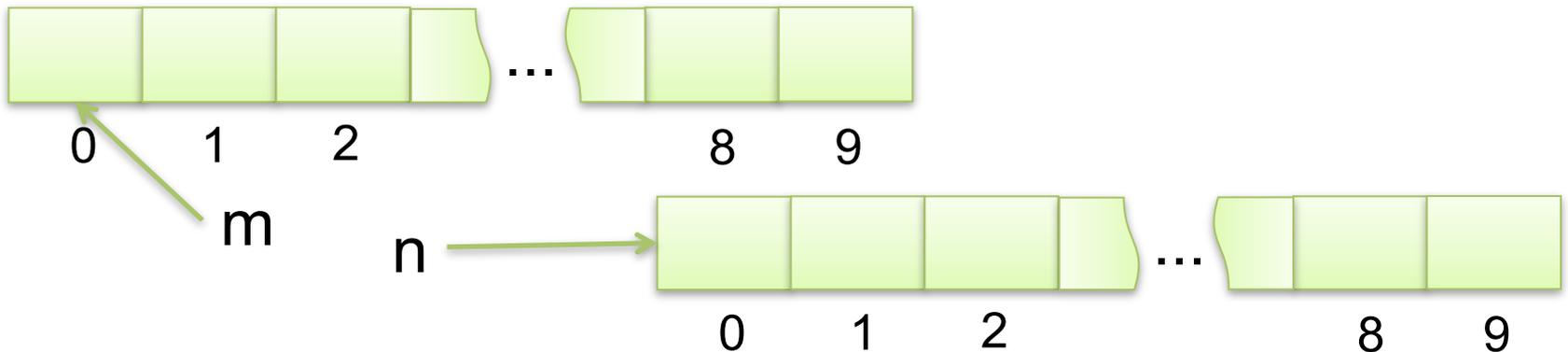
```
    return 0;
```

```
} /* saída: 10 20 30 40 50 60 70 */
```

# Cuidados ...

- Em **float** `m[10]` o identificador *m* é uma constante que endereça o primeiro elemento do array.
  - Portanto, não é possível mudar o valor de *m*.
- Ex: **float** `m[10], n[10];`

`m = n;`    */\* erro: m é constante ! \*/*



# Exercícios

- 1) Faça um programa em C que leia um conjunto de 10 valores inteiros e escreva-os em ordem contrária.
- 2) Desenvolva um programa que leia os valores de um vetor de 10 posições e calcule a soma dos valores armazenados em posições ímpares.
- 3) Faça um programa que leia 2 vetores de 5 elementos inteiros e que calcule o vetor soma.

# Exercícios

- 4) Ler um vetor A de 10 elementos inteiros.  
Determinar e imprimir a maior diferença entre dois elementos consecutivos desse vetor.
- 5) Faça um programa que leia 2 palavras de até 20 letras (vetor de 21 caracteres) e verifique se as mesmas são idênticas.

Obs: `gets(var);`

```
#include<string.h>  
strlen(char[]) // retorna  
o número de caracteres  
de uma string
```

# Exercício

Criar uma função que receba como argumento um número real e um número inteiro e retorne a raiz quadrada do número real através de Newton:

$$R_{n+1} = (R_n + (E/R_n))/2$$

$$\text{e } R_1 = E/2$$

para  $E$  = entrada,  $R$  = raiz quadrada  
e  $n$  = número de iterações