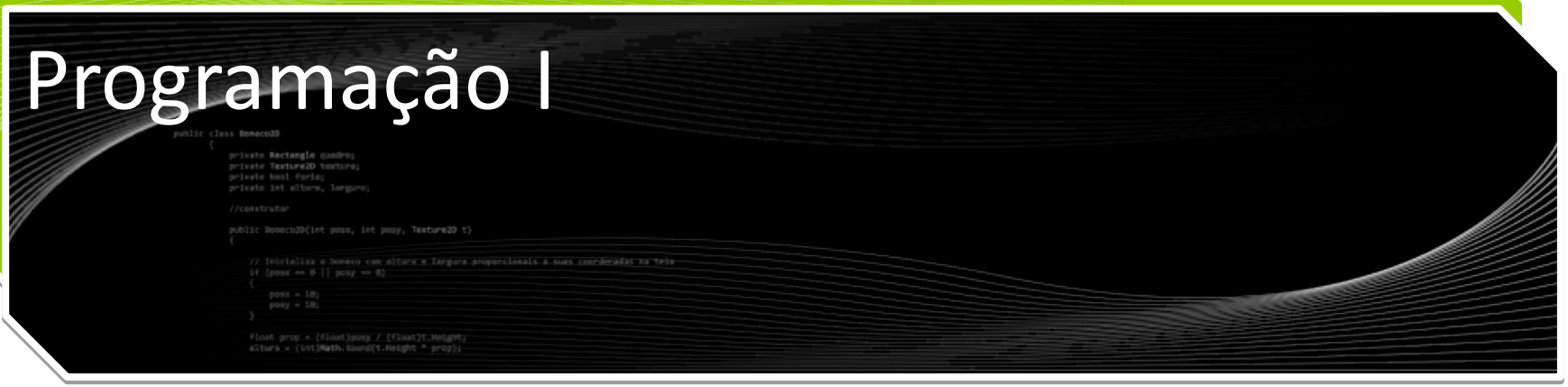

Departamento de Computação

CCENS – UFES

# Programação I



# Variáveis compostas

- Os vetores (arrays) também são conhecidos como **variáveis compostas**.

# Variáveis compostas

- Correspondem a um certo número de posições de memória (variáveis) que podem ser acessadas através de um único **identificador** (nome da variável) e um **índice**.
- Os **índices**, que geralmente devem ser apresentados junto com o nome da variável composta, servem para indicar a **posição** da variável que será feito o **acesso** (*leitura/modificação*).

# Variáveis compostas homogêneas unidimensionais

- Os **vetores** são chamados de ***variáveis compostas homogêneas unidimensionais***.

# Variáveis compostas homogêneas unidimensionais

- Uma variável *composta homogênea unidimensional* é considerada:
  - ***composta*** porque não consiste de uma variável, mas de um conjunto de variáveis.
  - ***unidimensional*** porque é necessário apenas ***um índice*** para o acesso aos dados de uma posição da variável.
  - ***homogênea*** porque o conteúdo de todas as posições de memória será de um ***mesmo tipo*** especificado na declaração da variável.

# Variáveis compostas homogêneas unidimensionais

- Declaração de ***vetores (arrays)*** unidimensionais:

```
<tipo> <identificador>[<TAM>];
<tipo> <identificador>[<TAM>] = {valor1, …, valorN};

Sendo:
    TAM: Tamanho
    TAM é tipo ordinal, começando sempre em 0 (zero)
```

- Exemplos:

```
float nota[10];
int numeros[20];
char nome[20];
```

# Variáveis compostas homogêneas unidimensionais

- Acesso ao valor da n-ésima posição do ***array***

```
Linguagem C
...
   <identificador_variavel>[indice]
  {sendo indice uma expressão ordinal}
...
```

- Exemplos:

Em: `float nota[10];`
Multiplicando por uma constante:
`nota[1] = nota[2] * 0.02;`

Lendo uma variável para o índice 2:

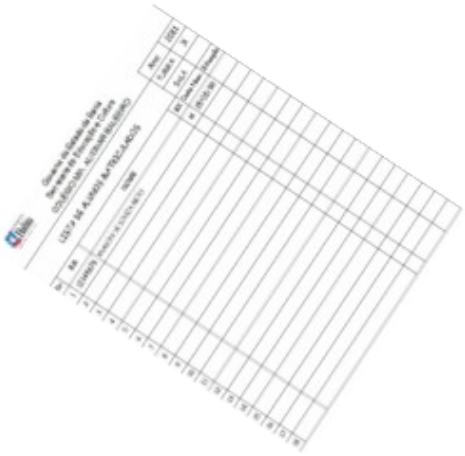
```
scanf("%f", &nota[2])
```

# Variáveis compostas homogêneas unidimensionais

- O acesso a **todos elementos** de um **array** é realizado **elemento por elemento**, isto é, **para cada acesso** deve-se especificar **um índice**.
- Isto pode ser feito de forma prática utilizando-se uma **estrutura de repetição** e uma **variável de controle** para indicar o **índice** de acesso em cada iteração.
- A **ordem do acesso** não é relevante
  - pode-se percorrer os elementos em qualquer ordem:
    - direta, inversa, aleatória, etc.

# Variáveis compostas homogêneas unidimensionais

- Considerando a variável nota (array de valores reais) declarada anteriormente:
  - O identificador **nota** fará referência à variável composta por **10 posições** de memória, nas quais se pode armazenar um valor real (em cada uma posição).
  - O **acesso** ao **valor** de cada uma das notas será realizado como **nota[x]**, onde **x** é um valor ou **expressão ordinal** (neste caso inteira).



# Variáveis compostas homogêneas unidimensionais

- Supondo que a variável NOTA tenha sido previamente inicializada com os valores abaixo:

NOTA

| Conteúdo | 60.0 | 70.0 | 90.0 | 62.0 | 55.0 | 91.0 | 20.0 | ... | 86.0 |
|---|---|---|---|---|---|---|---|---|---|
| Índice | 0 | 1 | 2 | 3 | 4 | 5 | 6 | ... | 9 |

nota[2] referencia terceiro elemento do conjunto cujo conteúdo é 90.0

Exemplo alterando seu valor:

nota[2] := nota[2] – 80.0; *{ = 90.0 + 80.0}*

| Conteúdo | 60.0 | 70.0 | 10.0 | 62.0 | 55.0 | 91.0 | 20.0 | ... | 86.0 |
|---|---|---|---|---|---|---|---|---|---|
| Índice | 0 | 1 | 2 | 3 | 4 | 5 | 6 | ... | 9 |

# Variáveis compostas homogêneas unidimensionais

NOTA

| Conteúdo | 60.0 | 70.0 | 90.0 | 62.0 | 55.0 | 91.0 | 20.0 | ... | 86.0 |
|---|---|---|---|---|---|---|---|---|---|
| Índice | 0 | 1 | 2 | 3 | 4 | 5 | 6 | ... | 9 |

Sendo i uma variável do tipo inteiro:
Se i = 6, a especificação **nota**[i] indica um acesso ao elemento de índice 6: no *array* acima seria então acessada a posição cujo valor atual é 20.0.

**Exemplos:**

```
printf("%f", nota[i-3]);
```
*{será impresso 62}*

```
printf("%f", nota[i/4]+i);
```
*{será impresso 76 – não altera o valor da posição 1}*

```
printf("nota", [i % 3 + 1]);
```
*{será impresso 70}*

```
nota[i - 1] -= 7;
```
*{a posição 5 passa a armazenar o valor 84 (91 - 7)}*

```
for (i=2; i<10;i++)
   printf("%f \n", nota[i]);
```
*{serão impressos os valores do índice 2 ao índice 9}*

# Variáveis compostas homogêneas unidimensionais

- Que tal escrever um programa em **C** que faça a leitura de 5 notas de alunos de uma disciplina e armazene-as no vetor **nota**:

```
int main(){
    float nota[5];
    int i;
    for (i=0;i<5;i++) {
       printf("Digite a nota do aluno %d: ", i);
       scanf("%f", &nota[i]);
    }
}
```

# Variáveis compostas homogêneas unidimensionais

- Exemplos:

```c
#include <stdio.h>

int main()
{
    float notas[3];
    int i;

    notas[0] = 4.2;
    notas[1] = 4.4;
    notas[2] = 5.0;
    for (i=0 ; i<3 ; i++)
        printf("Nota[%d] = %.2f\n", i, notas[i]);
}
```

# Variáveis compostas homogêneas unidimensionais

- Exemplos:

```c
#include <stdio.h>

int main()
{
    float notas[3] = {4.2, 4.4, 5.0};
    int i;
    for (i=0 ; i<3 ; i++)
        printf("Nota[%d] = %.2f\n", i, notas[i]);
}
```

```c
#include <stdio.h>

int main()
{
    float notas[] = {4.2, 4.4, 5.0};
    int i;
    for (i=0 ; i<3 ; i++)
        printf("Nota[%d] = %.2f\n", i, notas[i]);
}
```

# Variáveis compostas homogêneas unidimensionais

- Exemplos:

```c
#include <stdio.h>

int main()
{
    float notas[3];
    int i;

    for (i=0 ; i<3 ; i++)
    {
        printf("Digite a nota número %d: ", i);
        scanf("%f", &notas[i]);
    }

    printf("\nValores lidos:\n");
    for (i=0 ; i<3 ; i++)
        printf("Nota[%d] = %.2f\n", i, notas[i]);
}
```