

MC-102 — Aula 10

Vetores

Instituto de Computação – Unicamp

15 de Setembro de 2016

Roteiro

1 Introdução

2 Vetores

- Definição de Vetores
- Vetores – Como usar
- Vetores e a Memória
- Vetores – Exemplos

3 Informações Extras: Inicialização de um vetor

4 Exercícios

Vetores

- Vetores são construções de linguagens de programação que servem para armazenar vários dados de um mesmo tipo de forma simplificada.

Vetores

- Suponha que desejamos guardar notas de alunos.
- Com o que sabemos, como armazenaríamos 3 notas?

```
float nota1, nota2, nota3;  
  
printf("Nota do aluno 1: ");  
scanf("%f", &nota1);  
printf("Nota do aluno 2: ");  
scanf("%f", &nota2);  
printf("Nota do aluno 3: ");  
scanf("%f", &nota3);
```

- Com o que sabemos, como armazenaríamos 100 notas?

```
float nota1, nota2, nota3, ..., nota100;
```

```
printf("Nota do aluno 1: ");
```

```
scanf("%f", &nota1);
```

```
printf("Nota do aluno 2: ");
```

```
scanf("%f", &nota2);
```

```
...
```

```
printf("Nota do aluno 100: ");
```

```
scanf("%f", &nota100);
```

- Criar 100 variáveis distintas não é uma solução elegante para este problema.

Definição de Vetores

- Um vetor em C é uma coleção de variáveis de um mesmo tipo que são referenciadas por um **identificador único**.
- Características de um vetor:
 - ▶ As variáveis ocupam posições contíguas na memória.
 - ▶ O acesso se dá por meio de um índice inteiro.
 - ▶ O vetor possui um tamanho pré-definido.
 - ▶ O acesso do vetor com um índice fora dos limites, pode causar comportamento anômalo do programa.

Declaração de um vetor

Para se declarar um vetor usamos a seguinte sintaxe:

- **tipo_variável** identificador[**tamanho do vetor**];

Exemplos

```
float notas[100]; //vetor "notas" corresponde
                  //a 100 variáveis do tipo float
```

```
int primos[20]; //vetor "primos" corresponde
                // a 20 variáveis do tipo int
```

Usando um vetor

- Após declarada uma variável do tipo vetor, pode-se acessar uma determinada posição do vetor utilizando-se um índice de valor inteiro.
- Sendo n o tamanho do vetor, os índices válidos para o vetor vão de 0 até $n - 1$.
 - ▶ A primeira posição de um vetor tem índice 0.
 - ▶ A última posição de um vetor tem índice $n - 1$.
- A sintaxe para acesso de uma determinada posição é:
 - ▶ `identificador[posição];`

O vetor em uma posição específica tem o mesmo comportamento que uma variável simples.

Exemplo

```
int nota[10];
int a;
nota[5] = 95; // "nota[5]" corresponde a uma var. inteira
a = nota[5];
```


Usando um vetor

- Você deve usar valores inteiros como índice para acessar uma posição do vetor.
- O valor pode ser inclusive uma variável inteira.

Exemplo

```
int g, vet[10];  
for(g=0; g<10; g++)  
    vet[g]=5*g;
```

- Quais valores estarão armazenados em cada posição do vetor após a execução deste código?

Vetores e a Memória

- Suponha o código:

```
int d;  
int vetor[5];  
int f;
```

- Na memória temos:

Nome	d	vetor					f
Índice	-	0	1	2	3	4	-

Vetores e a Memória

- Ao executar o comando
`vetor[3]=10;`

temos:

Nome	d	vetor					f
Índice	-	0	1	2	3	4	-
					10		

Vetores e a Memória

- O que ocorre se forem executados os comandos abaixo?

```
vetor[3]=10;
```

```
vetor[5]=5;
```

```
vetor[-1]=1;
```

Vetores e a Memória

- Ao executar

```
vetor[3]=10;
```

```
vetor[5]=5;
```

```
vetor[-1]=1;
```

teremos:

Nome	d	vetor					f
Índice	-	0	1	2	3	4	-
	1				10		5

- O seu programa estará errado pois você está alterando inadvertidamente valores de outras variáveis.
- Em alguns casos o seu programa será encerrado (**Segmentation Fault**).
- Em outros casos seu programa poderá continuar executando, mas ocorrerão erros difíceis de serem rastreados.

Questões importantes sobre vetores

- O tamanho do vetor é pré-definido (durante a execução do programa não pode ser alterado).
- O uso de índices fora dos limites podem causar comportamento anômalo do programa.

Como armazenar até 100 notas?

```
float nota[100];
int n, i;

printf("Número de alunos: ");
scanf("%d", &n);

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

- O programa acima está correto?

Como armazenar até 100 notas?

- Você deve testar se $n > 100$ para evitar erros!!

```
float nota[100];
int n, i;

printf("Número de alunos: ");
scanf("%d", &n);
if(n>100){
    n=100;
    printf("\nNumero máximo de alunos alterado para 100");
}
for (i = 0; i < n; i++) {
    printf("Digite a nota do aluno %d: ", i);
    scanf("%f", &nota[i]);
}
```


Exemplo: Produto Interno de dois vetores

- Ler dois vetores de dimensão 5 e computar o produto interno destes.
- Quais tipos de variáveis usar?

Exemplo: Produto Interno de dois vetores

- Abaixo temos o código para ler dois vetores de dimensão 5.

```
int main(){
    double vetor1[5], vetor2[5], resultado;
    int i;

    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 1:",i);
        scanf("%lf",&vetor1[i]);
    }
    printf("\n\n");
    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 2:",i);
        scanf("%lf",&vetor2[i]);
    }

    //calculando o produto interno
    .....
}
```

Exemplo: Produto Interno de dois vetores

- Abaixo temos a parte do código para computar o produto interno dos vetores.

```
int main(){
    double vetor1[5], vetor2[5], resultado;
    int i;

    ...

    //calculando o produto interno
    resultado = 0.0;
    for(i=0; i < 5; i++){
        resultado = resultado + ( vetor1[i]*vetor2[i] );
    }
    printf("\n\n0 produto interno é: %lf\n",resultado);
}
```

Exemplo: Produto Interno de dois vetores

- Agora o código completo.

```
int main(){
    double vetor1[5], vetor2[5], resultado;
    int i;

    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 1:",i);
        scanf("%lf",&vetor1[i]);
    }
    printf("\n\n");
    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 2:",i);
        scanf("%lf",&vetor2[i]);
    }

    //calculando o produto interno
    resultado = 0.0;
    for(i=0; i < 5; i++){
        resultado = resultado + ( vetor1[i]*vetor2[i] );
    }
    printf("\n\nO produto interno é: %lf\n",resultado);
}
```

Exemplo: Elementos Iguais

- Ler dois vetores com 5 inteiros cada.
- Checar quais elementos do segundo vetor são iguais a algum elemento do primeiro vetor.
- Se não houver elementos em comum, o programa deve informar isso.

Exemplo: Elementos Iguais

- Abaixo está o código que faz a leitura de dois vetores.

```
int main(){
    int vetor1[5], vetor2[5];
    int i, j, umEmComum;

    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 1:",i);
        scanf("%d",&vetor1[i]);
    }
    printf("\n\n");
    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 2:",i);
        scanf("%d",&vetor2[i]);
    }

    ...
}
```

Exemplo: Elementos Iguais

- Para cada elemento do **vetor1** testamos todos os outros elementos do **vetor2** para saber se são iguais.
- Usamos uma variável indicadora para decidir ao final dos laços encaixados, se os vetores possuem ou não um elemento em comum.

```
int main(){
    int vetor1[5], vetor2[5];
    int i, j, umEmComum;

    ...

    umEmComum = 0; //Assumimos que não hajam elementos comuns
    for(i = 0; i < 5 ; i++)
        for(j = 0; j < 5; j++)
            if(vetor1[i] == vetor2[j]){
                umEmComum = 1; //Descobrimos que há elemento comum
                printf("Posicao %d do vetor1 igual a posição %d do vetor2.\n",i,j);
            }
    if(!umEmComum)
        printf("Nenhum elemento em comum!\n");
}
```

Exemplo: Elementos Iguais

- Código completo abaixo.

```
int main(){
    int vetor1[5], vetor2[5];
    int i, j, umEmComum;

    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 1:",i);
        scanf("%d",&vetor1[i]);
    }
    printf("\n\n");
    for(i=0; i<5; i++){
        printf("Entre com valor da posição %d para vetor 2:",i);
        scanf("%d",&vetor2[i]);
    }

    umEmComum = 0;
    for(i = 0; i < 5 ; i++)
        for(j = 0; j < 5; j++)
            if(vetor1[i] == vetor2[j]){
                umEmComum = 1;
                printf("Posicao %d do vetor1 igual a posição %d do vetor2.\n",i,j);
            }
    if(!umEmComum)
        printf("Nenhum elemento em comum!\n");
}
```


Informações Extras: Inicialização de um vetor

- Em algumas situações é necessário declarar e já atribuir um conjunto de valores constantes para um vetor.
- Em C, isto é feito atribuindo-se uma lista de elementos para o vetor na sua criação da seguinte forma:

```
tipo identificador[] = {elementos separados por vírgula} ;
```

- Exemplos:

```
double vet1[] = {2.3, 3.4, 4.5, 5.6};
```

```
int vet2[] = {5, 4, 3, 10, -1, 0};
```

- Note que automaticamente é criado um vetor com tamanho igual ao número de dados da inicialização.

Informações Extras: Inicialização de um vetor

```
#include <stdio.h>

int main(){
    double vet1[] = {2.3, 3.4, 4.5, 5.6};
    int vet2[] = {5, 4, 3, 10, -1, 0};
    int i;

    for(i=0; i<4; i++)
        printf("%lf\n", vet1[i]);

    for(i=0; i<6; i++)
        printf("%d\n", vet2[i]);

}
```

Exercício

- Escreva um programa que lê 10 números inteiros e os salva em um vetor. Em seguida o programa deve encontrar a posição do maior elemento do vetor e imprimir esta posição.

Exercício

- Escreva um programa que lê 10 números ponto flutuante e os salva em um vetor. Em seguida o programa deve calcular a média dos valores armazenados no vetor e imprimir este valor.

Exercício

- Escreva um programa que lê 10 números inteiros e os salva em um vetor. Em seguida o programa deve ler um outro número inteiro C . O programa deve então encontrar dois números de posições distintas do vetor cuja multiplicação seja C e imprimi-los. Caso não existam tais números, o programa deve informar isto.
- Exemplo: Se $vetor = (2, 4, 5, -10, 7)$ e $C = 35$ então o programa deve imprimir “5 e 7”. Se $C = -1$ então o programa deve imprimir “Não existem tais números”.