

MINIAULA DE ALGORITMOS

VETORES

Prof. Ivanilton Polato

Departamento Acadêmico de Computação (DACOM-CM)

ipolato@utfpr.edu.br

Vetores: o que são?

- Variáveis compostas, homogêneas e unidimensionais
- Agregam um conjunto de variáveis:
 - *de mesmo tipo*
 - *com um mesmo nome*
 - *com índices (posições) diferentes*
 - *alocadas sequencialmente na memória*
- Cada índice do vetor é acessado como uma variável comum

Vetores: o que são?

- Exemplo: um vetor de números inteiros com 5 posições

```
int v[5];           // Declaração do vetor
v[0] = 5;           // índice 0 recebe o valor 5
v[1] = 7;           // índice 1 recebe o valor 7
v[2] = 12;          // índice 2 recebe o valor 12
v[3] = -1;          // índice 3 recebe o valor -1
v[4] = 0;           // índice 4 recebe o valor 0
```

v	5	7	12	-1	0
[]	0	1	2	3	4

Vetores: índices

- Permitem o acesso direto a cada posição do vetor
- São representados por valores inteiros e sequenciais



Sempre começam no índice ZERO!

- Representam a posição em memória da variável dentro do vetor
 - *Deslocamento dos endereços (tamanho em bytes) a partir da posição inicial*

Vetores: declaração

- Similar às variáveis simples, apenas adicionando os “[]”
- Podem ser de qualquer tipo:

```
int v[10]; // vetor de inteiro com 10 posições  
float v1[15]; // vetor de float com 15 posições  
char str[50]; // vetor de char com 50 posições
```
- Atenção: os vetores do tipo CHAR são também conhecidos como STRINGS. São um tipo especial e serão estudados em aula específica.

Vetores: manipulação

- Nas declarações, o vetor pode ser inicializado em sua totalidade:

```
int v[5] = {5, 7, 12, -1, 0};
```

índices: 0 1 2 3 4

- No caso acima, o vetor será declarado e inicializado com os números nas respectivas posições. A quantidade de elementos do conjunto deve ser do mesmo tamanho do vetor.

```
int vet[10] = {0};
```

- Nesse caso, todas as posições do vetor serão inicializadas com 0. Esse mecanismo só funciona para o número zero.

Vetores: manipulação

- As posições do vetor recebem valores como variáveis simples:

- *Atribuição direta:* `v[0] = 5;`
- *Comando de entrada:* `scanf ("%d", &v[0]);`



ATENÇÃO: as posições do vetor devem ser manipuladas **INDIVIDUALMENTE**, uma por vez, mesmo que em uma estrutura de repetição!

Vetores: manipulação + repetição

```
1. #include <stdio.h>
2. int main() {
3.     int i, v[5];
4.     for(i=0; i<5; i++) {
5.         printf("Número: ");
6.         scanf("%d", &v[i]);
7.     }
8.     return 0;
9. }
```

Exemplo de execução:

Número: 4 $i == 0 \rightarrow v[0] = 4;$

Número: 13 $i == 1 \rightarrow v[1] = 13;$

Número: -8 $i == 2 \rightarrow v[2] = -8;$

Número: 7 $i == 3 \rightarrow v[3] = 7;$

Número: 25 $i == 4 \rightarrow v[4] = 25;$

v	4	13	-8	7	25
[]	0	1	2	3	4

Vetores: manipulação + repetição

```
1. #include <stdio.h>
2. int main() {
3.     int i, v[5] = {5,4,7,2,1};
4.     for(i=0; i<5; i++) {
5.         printf("V[%d]: %d\n", i, v[i]);
6.     }
7.     return 0;
8. }
```

Exemplo de execução:

V[0]: 5

V[1]: 4

V[2]: 7

V[3]: 2

V[4]: 1

v	5	4	7	2	1
[]	0	1	2	3	4

Vetores: dicas

- Índices sempre começam em ZERO!
 - *Lembre-se do limite da variável contadora na repetição!*
- Manipulação individual das posições!
 - *Todas as posições manipuladas uma por vez!*
 - *Não se esqueça do & no scanf! (`scanf("%d", &v[i]);`)*
- Use como uma variável comum:
 - `if (v[i] > 0) { ... }`
 - `switch (v[i]) { ... }`
 - `soma = soma + v[i];`