Algoritmos e Programação

Aula 20

Estruturas Complexas (Vetores)

Estruturas de Dados

- Completamos, em C, o repertório de **construções** vistas em **fluxogramas** e **pseudocódigo**.
- Até o momento todas variáveis utilizadas nos programas eram simples.
- A partir de agora iremos conhecer algumas ***estruturas de dados***, isto é, **formas** de **coleções de dados em memória**
- Hoje veremos o **vetor**.

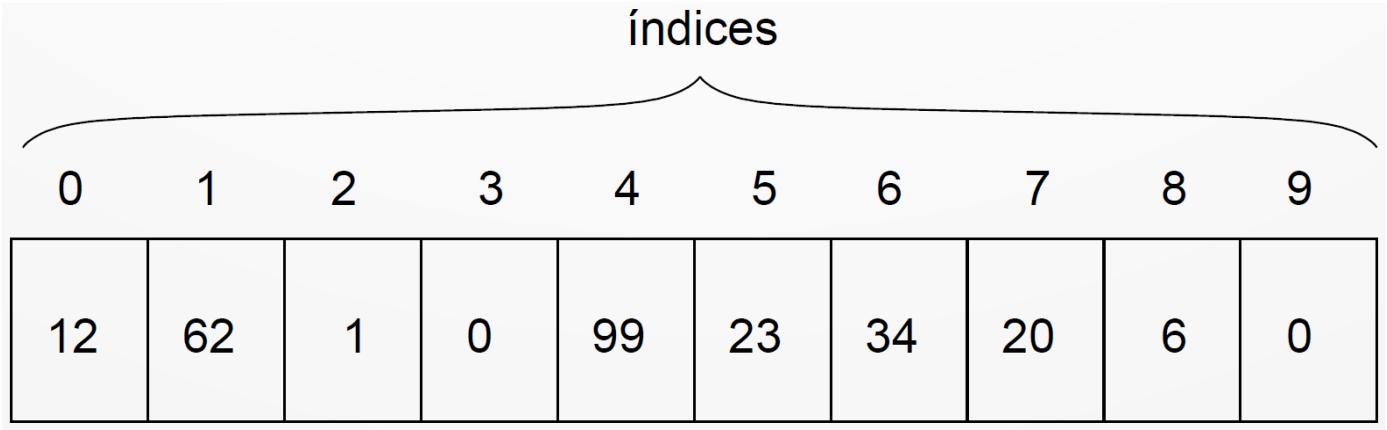
Vetores

- Um **vetor**, arranjo ou **matriz unidimensional** (em inglês, array) é uma **coleção de dados:**
  - De tamanho **fixo** (N).
  - Composta de elementos do **mesmo tipo**
  - Que estão armazenados de **forma contínua** em **memória**, ou seja, **um após o outro**, sequencialmente.
  - Podem ser **acessados a partir** de um **índice numérico** de `0` a `N-1`.

Vetores

- Um **vetor**, arranjo ou **matriz unidimensional** (em inglês, array) é uma **coleção de dados:**
  - De tamanho **fixo** (N).
  - Composta de elementos do **mesmo tipo**
  - Que estão armazenados de **forma contínua** em **memória**, ou seja, **um após o outro**, sequencialmente.
  - Podem ser **acessados a partir** de um **índice numérico** de `0` a `N-1`.

Exemplo de *vetor* de inteiros com 10 elementos



Alocação de Vetores

Vetores são alocados através de uma **declaração de variável com** uma ***indicação de multiplicidade***, escrita após o respectivo nome (indentificador).

Exemplo:

Alocação de Vetores

Vetores são alocados através de uma **declaração de variável com** uma ***indicação de multiplicidade***, escrita após o respectivo nome (indentificador).

Exemplo:

```
#include <stdio.h>
int main() {
    int v[10];
}
```

Aloca um vetor de nome **v**, com **10 elementos** do tipo **inteiro**.

Alocação de Vetores

Posso declarar **vetores** de **todos tipos** (exceto `void`), ou seja, posso declarar vetores: **`int`**, **`float`**, **`double`** e **`char`.**

Posso declarar **vetores** de **qualquer tamanho** (desde que exista memória suficiente para ser alocada).

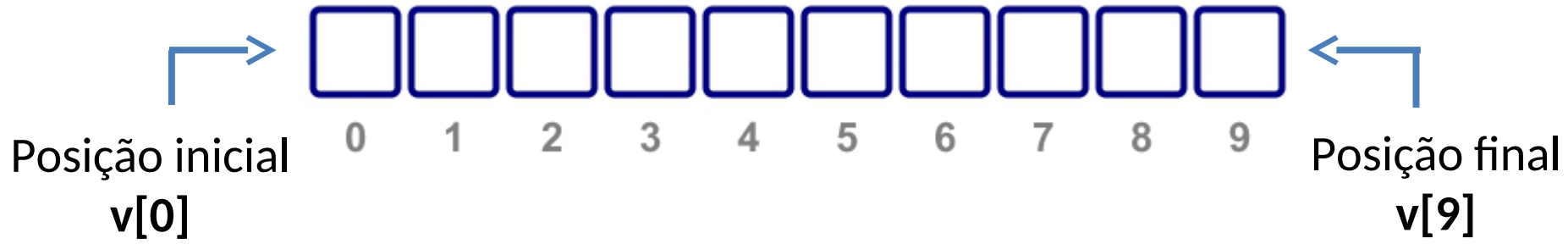
Alocação de Vetores

Posso declarar **vetores** de **todos tipos** (exceto `void`), ou seja, posso declarar vetores: **`int`**, **`float`**, **`double`** e **`char`.**

Posso declarar **vetores** de **qualquer tamanho** (desde que exista memória suficiente para ser alocada).

Representação gráfica de um vetor:

```
int v[10];
```



Inicialização de vetores

Assim como variáveis comuns, os elementos do vetor não são inicializados ao serem alocados. A **declaração abaixo cria** um **vetor** de com 10 inteiros, cujo **conteúdo** é **lixo de memória**.

```
int v[10];
```

Podemos **inicializar** o **vetor com valores iniciais**, colocados entre { } e separados por vírgula.

```
int v[10] = {2,3,5,7,11,13,17,19,23,29};
```

Este trecho de programa aloca um vetor de 10 inteiros, e armazena nele os 10 primeiros números primos.

Inicialização de vetores

A declaração abaixo cria um vetor de 10 inteiros, onde todos os elementos do vetor são inicializados na declaração

```
int p[10] = {2,3,5,7,11,13,17,19,23,29};
```

p	2	3	5	7	11	13	17	19	23	29
	0	1	2	3	4	5	6	7	8	9

Se um vetor de tamanho N é inicializado com M elementos, onde M < N, os demais N – M elementos são inicializados com 0 (zero).

```
int vetor[10] = {6,13,7,10,20,127};
```

vetor	6	13	7	10	20	127	0	0	0	0
	0	1	2	3	4	5	6	7	8	9

Inicialização de vetores

É comum a utilização de laços **`for`** para preencher vetores grandes.

```c
#include <stdio.h>
int main() {
    int a[1000], i;
    for(i = 0; i < 1000; i++) {
        a[i] = i * i;
    }
}
```

Este programa aloca um vetor de 1000 inteiros e armazena em cada posição o quadrado de seu índice.

Inicialização de vetores

Podemos, também, inicializar um vetor com valores digitados pelo usuário:

```c
#include <stdio.h>
int main() {
    float a[10];
    int i;
    for(i = 0; i < 10; i++) {
        printf("Digite o valor da posicao %d: ", i);
        scanf("%f", &a[i]);
    }
    printf("\nVetor preenchido:\n");
    for(i = 0; i < 10; i++) {
        printf("a[%d] -> %f\n", i, a[i]);
    }
}
```

Acesso a elementos do vetor

Cada elemento do vetor se comporta como se fosse uma **variável individual** do tipo especificado.

Para **acessarmos** os **elementos** do **vetor** através do índice, **utilizamos** o **operador** [ ]

```
a[1] = 3;
x = a[0] - 23;
a[2] = a[0] + a[1];
```

Acesso a elementos do vetor

Cada elemento do vetor se comporta como se fosse uma **variável individual** do tipo especificado.

Para **acessarmos** os **elementos** do **vetor** através do índice, **utilizamos** o **operador** [ ]

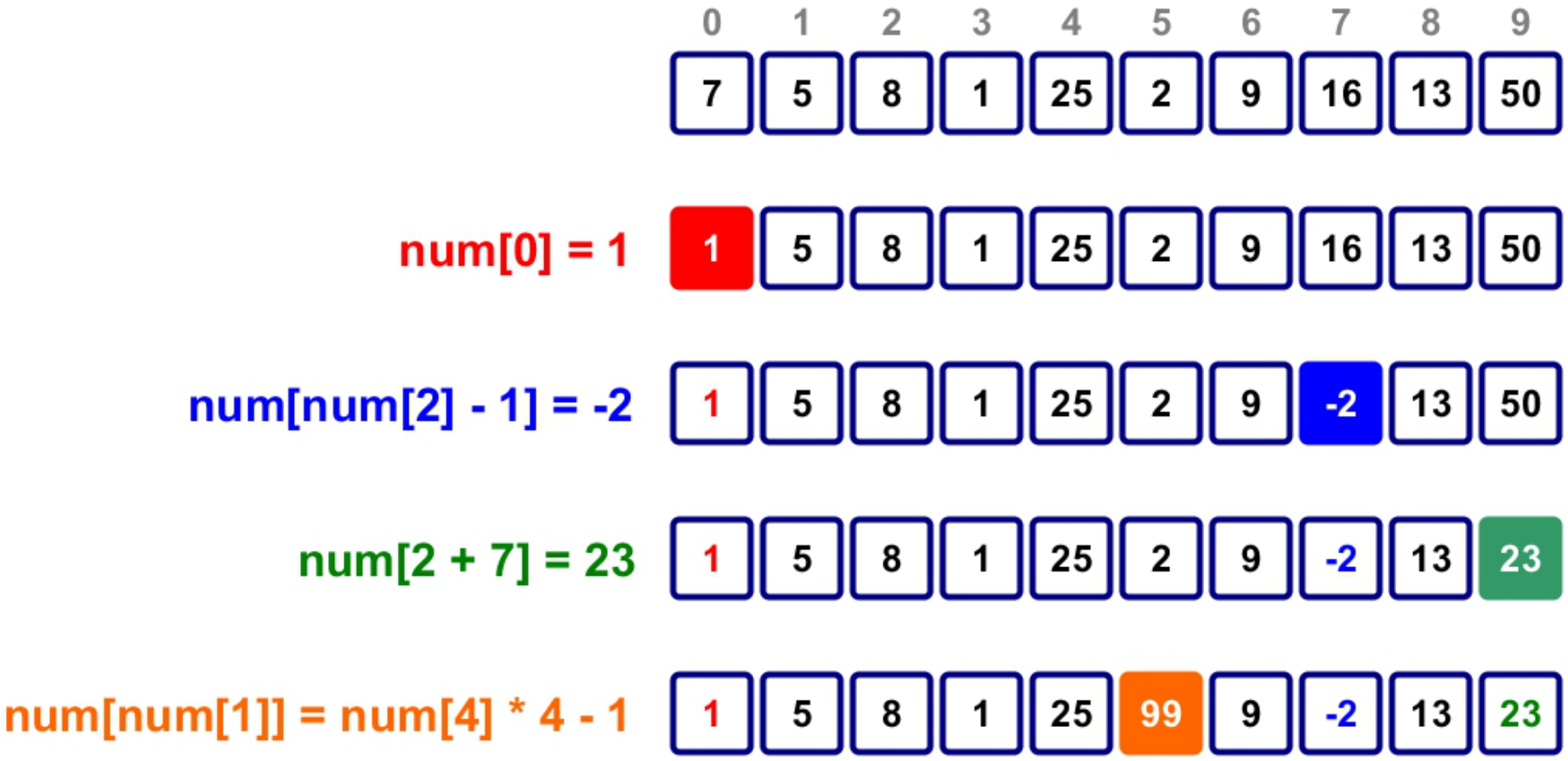
```
a[1] = 3;
x = a[0] - 23;
a[2] = a[0] + a[1];
```

Podemos utilizar uma expressão no lugar do índice de um vetor:

```
x = 10 - 7;
y = a[x + 2];
```

Acesso a elementos do vetor

Atribuir valores a um vetor é igual escrever em uma variável!



Acesso a elementos do vetor

Para acessar todos elementos de um vetor, utilizamos um laço de repetição. **Ex**: somatório elementos vetor:

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

Saída: `Somatório = 30`

Limites do vetor

A linguagem C **não verifica** se estamos tentando **acessar** somente **índices válidos** nos vetores.

A **responsabilidade** é do **programador**!

O poder é de vocês!

```c
#include <stdio.h>
int main() {
    int v[10] = {9,8,7,6,5,4,3,2,1,0},i;
    for(i = 0; i <= 10; i++){
        printf("v[%d]->%d\n",i,v[i]);
    }
}
```

Execute este programa e tente entender o que acontece!

Exemplo:

- Escreva um programa que simula uma eleição com 8 candidatos e N eleitores, onde N é informado pelo usuário.
  - Devemos ler o voto de cada eleitor e contabilizá-lo, ou seja, incrementar a quantidade de votos que um candidato recebeu, ou incrementar a quantidade de votos nulos (caso o eleitor não tenha votado em nenhum candidato).

```c
#include <stdio.h>
int main() {
    int C0, C1, C2, C3, C4, C5, C6, C7, voto, nulos, n, i;
    C0 = C1 = C2 = C3 = C4 = C5 = C6 = C7 = nulos = 0;
    printf("Digite o numero de votantes: ");
    scanf("%d", &n);
    for(i = 1; i <= n; i++) {
        printf("\tDigite o numero do seu candidato: ");
        scanf("%d", &voto);
        switch(voto) {
            case 0: C0++; break;
            case 1: C1++; break;
            case 2: C2++; break;
            case 3: C3++; break;
            case 4: C4++; break;
            case 5: C5++; break;
            case 6: C6++; break;
            case 7: C7++; break;
            default: nulos++;
        }
    }
    printf("\n\nResultado da eleicao:\n\n");
    printf("\tCand0: %d\n\tCand1: %d\n", C0, C1);
    printf("\tCand2: %d\n\tCand3: %d\n", C2, C3);
    printf("\tCand4: %d\n\tCand5: %d\n", C4, C5);
    printf("\tCand6: %d\n\tCand7: %d\n", C6, C7);
    printf("\tNulos: %d\n", nulos);
}
```

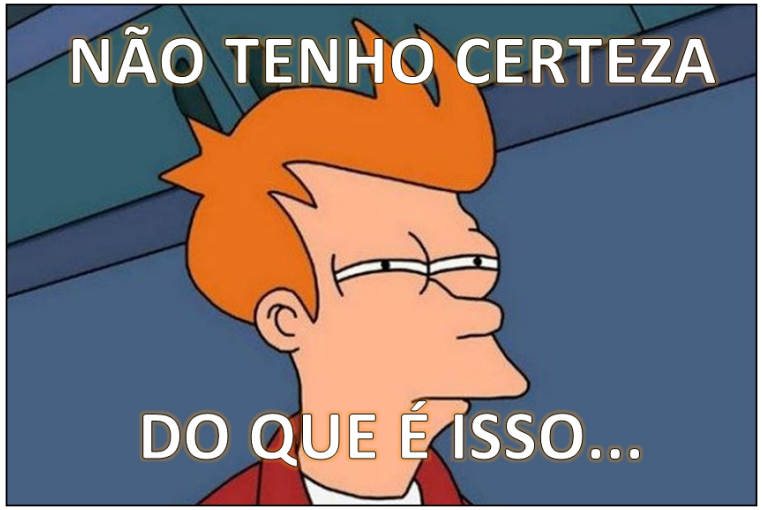
Exemplo:

- Podemos modificar o exemplo anterior para utilizar um vetor:
  - Ao invés de criarmos uma variável para cada candidato, criaremos um vetor para armazenar os votos de todos candidatos!

```c
#include <stdio.h>
int main() {
    int C[8], voto, nulos = 0, n, i;
    for(i = 0; i < 8; i++) {
        C[i] = 0;
    }
    printf("Apuracao da eleicao:\n\tNumero de votantes: ");
    scanf("%d", &n);
    for(i = 1; i <= n; i++) {
        printf("\t\tVoto: ");
        scanf("%d", &voto);
        if(voto >= 0 && voto < 8)
            C[voto]++;
        else
            nulos++;
    }
    printf("\n\nResultado da eleicao:\n\n");
    for(i = 0; i < 8; i++){
        printf("\tCand%d: %d\n", i, C[i]);
    }
    printf("\tNulos: %d\n", nulos);
}
```

Exemplo:

- Imagine agora, que ao invés de 8 candidatos, o programa deve contabilizar os votos de 20 diferentes candidatos...
- Olha só como ficaria o programa **sem vetor**:




```c
#include <stdio.h>
int main() {
    int C0, C1, C2, C3, C4, C5, C6, C7, C8, C9, C10,
        C11, C12, C13, C14, C15, C16, C17, C18, C19,
        voto, nulos, n, i;
    C0 = C1 = C2 = C3 = C4 = C5 = C6 = C7 =
    C8 = C9 = C10 = C11 = C12 = C13 = C14 =
    C15 = C16 = C17 = C18 = C19 = nulos = 0;
    printf("Apuracao da eleicao:\n\tNumero de votantes: ");
    scanf("%d", &n);
    for(i = 1; i <= n; i++) {
        printf("\t\tVoto: ");
        scanf("%d", &voto);
        switch(voto) {
            case 0: C0++; break;
            case 1: C1++; break;
            case 2: C2++; break;
            case 3: C3++; break;
            case 4: C4++; break;
            case 5: C5++; break;
            case 6: C6++; break;
            case 7: C7++; break;
            case 8: C8++; break;
            case 9: C9++; break;
            case 10: C10++; break;
            case 11: C11++; break;
            case 12: C12++; break;
            case 13: C13++; break;
            case 14: C14++; break;
            case 15: C15++; break;
            case 16: C16++; break;
            case 17: C17++; break;
            case 18: C18++; break;
            case 19: C19++; break;
            default: nulos++;
        }
    }
    printf("\n\nResultado da eleicao:\n\n");
    printf("\tCand0: %d\n\tCand1: %d\n", C0, C1);
    printf("\tCand2: %d\n\tCand3: %d\n", C2, C3);
    printf("\tCand4: %d\n\tCand5: %d\n", C4, C5);
    printf("\tCand6: %d\n\tCand7: %d\n", C6, C7);
    printf("\tCand8: %d\n\tCand9: %d\n", C8, C9);
    printf("\tCand10: %d\n\tCand11: %d\n", C10, C11);
    printf("\tCand12: %d\n\tCand13: %d\n", C12, C13);
    printf("\tCand14: %d\n\tCand15: %d\n", C14, C15);
    printf("\tCand16: %d\n\tCand17: %d\n", C16, C17);
    printf("\tCand18: %d\n\tCand19: %d\n", C18, C19);
    printf("\tNulos: %d\n", nulos);
}
```

Exemplo:

- E agora, o mesmo programa para 20 candidatos utilizando um **vetor**:

```c
#include <stdio.h>
int main() {
    int C[20], voto, nulos = 0, n, i;
    for(i = 0; i < 20; i++) {
        C[i] = 0;
    }
    printf("Apuracao da eleicao:\n\tNumero de votantes: ");
    scanf("%d", &n);
    for(i = 1; i <= n; i++) {
        printf("\t\tVoto: ");
        scanf("%d", &voto);
        if(voto >= 0 && voto < 20)
            C[voto]++;
        else
            nulos++;
    }
    printf("\n\nResultado da eleicao:\n\n");
    for(i = 0; i < 20; i++){
        printf("\tCand%d: %d\n", i, C[i]);
    }
    printf("\tNulos: %d\n", nulos);
}
```

Exercícios

1. Desenvolva um programa em C que declara um vetor de tamanho 10, inicializado (na declaração) com valores inteiros quaisquer. O programa deve imprimir todos os valores do vetor utilizando um laço for.
2. Desenvolva um programa em C que declara um vetor de tamanho 8, inicializado (na declaração) com valores inteiros quaisquer. O programa deve imprimir todos os valores do vetor incrementados em 5 unidades cada.
3. Desenvolva um programa em C que declara um vetor de inteiros de tamanho 20 cujo conteúdo segue a regra v[i] = i + 3. Como saída o programa deve imprimir todo o vetor gerado na primeira linha, e na segunda linha deve imprimir apenas os valores múltiplos de 3 pertencentes ao vetor.
4. Desenvolva um programa em C que declara dois vetores (A e B) de tamanho 6 (inicializados na declaração). Um terceiro vetor (C), também de 6 posições deverá ser declarado, e seu conteúdo deverá ser a soma dos vetores A e B.
5. Desenvolva um programa em C que declara dois vetores de tamanho 4, cujos valores são digitados pelo usuário. Os dois vetores devem ser comparados (posição por posição), e o menor dos valores é impresso (para cada posição).
6. Desenvolva um programa em C que percorre um vetor de tamanho 10 com valores inteiros quaisquer (inicializados na declaração); e imprime como resultado o numero de vezes que os números 3 e 1 são encontrados no vetor.
7. Desenvolva um programa em C que descobre o maior dentre 10 valores inteiros quaisquer. Ao invés de declarar 10 variáveis `int`, declare um vetor com 10 posições e o utilize para descobrir seu maior valor.