Vetores, Strings e Matrizes

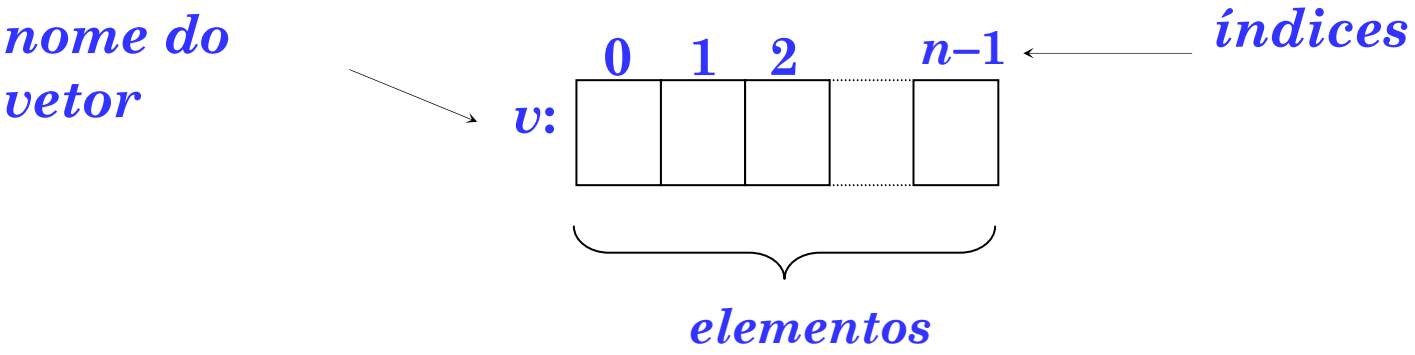
Prof. Dr. Silvio do Lago Pereira

Departamento de Tecnologia da Informação

Faculdade de Tecnologia de São Paulo

Vetores

- Um vetor é um tipo de dados agregado homogêneo, cujos itens são identificados por índices.
- Vetores são declarados com um sufixo [n] , indicando que o vetor tem capacidade para n itens;
- A indexação inicia-se sempre em zero.



Uso de índices inadequados

```
#include <stdio.h>


void main(void) {
    int x[3], y[4];

    x[2] = y[0] = 1;
    x[3] = 2;
    y[-1] = 3;

    printf("%d %d", x[2], y[0]);
}
```

Exercício

Codifique um programa para ler 5 números e, depois, exibi-los na ordem inversa.

```
#include <stdio.h>

void main(void) {
   int i, v[5];

   for(i=0; i<5; i++) {
      printf("%do. valor? ",i+1);
      scanf("%d",&v[i]);
   }

   printf("\nOrdem inversa: ");

   for(i=4; i>=0; i--) printf("%d ", v[i]);
}
```

Inicialização de vetores

- O vetor deve ser global ou estático.
- Os valores iniciais devem ser constantes.

```
#include <stdio.h>


void main(void) {
    static float moeda[5] = {
        1.00, 0.50, 0.25, 0.10, 0.05
    };
    ...
}
```

Valores iniciais insuficientes

```
#include <stdio.h>

#define max 5

void main(void) {
   static int A[max] = {9, 3, 2, 7};
   auto int i;

   for(i=0; i<max; i++)
      printf("%d", A[i]);
}
```

Tamanho implícito

```
#include <stdio.h>

void main(void) {
    static char ds[] = {
        'D', 'S', 'T', 'Q', 'Q', 'S', 'S'
    };
    ...
}
```

Parâmetros do tipo vetor

- O nome do vetor representa seu endereço;
- Em geral, parâmetros são passados por valor;
- Exceto vetores, que são passados por referência.

```
#include <stdio.h>

void main(void) {
   int x[3], y[4];
   printf("\n x =  %p e y =  %p", x, y);
}
```

Exemplo: temperatura média

Problema:

Dadas as temperaturas registradas diariamente, durante uma semana, determine em quantos dias a temperatura esteve acima da média.

Solução:

① obter os valores das temperaturas;

② calcular a média entre esses valores;

③ contabilizar os valores acima da média.

Exemplo: programa principal

```
#include <stdio.h>

#define max 7

void main(void) {
    float t[max], m;

    obtem(t);
    m = media(t);
    printf("Estatística: %d", conta(t,m));
}
```

Exemplo: obtendo as temperaturas

```
void obtem(float t[]) {
    int i;

    puts("Informe as temperaturas: ");

    for(i=0; i<max; i++) {
        printf("%dº valor? ",i+1);
        scanf("%f", &t[i] );
    }
}
```

Exemplo: calculando a média

```
float media(float t[]) {
    int i;
    float s=0;

    for(i=0; i<max; i++)
        s += t[i];

    return s/max;
}
```

Exemplo: contabilizando

```
int conta(float t[], float m) {
    int i, c=0;

    for(i=0; i<max; i++)
        if( t[i]>m )
            c++;

    return c;
}
```

Exercício

Codifique a função histograma(t).


```
void histograma(float t[]) {
   static char d[7]={'D','S','T','Q','Q','S','S'};
   int i, j;

   for(i=0; i<max; i++) {
      printf("\n%c: ",d[i]);
      for(j=1; j<=t[i]; j++) putch(220);
   }
}
```

Ordenação “Bubble Sort”

- **Estratégia: Permutar pares de itens consecutivos fora de ordem, enquanto for possível.**

$v[0]$	$v[1]$	$v[2]$	$v[3]$	$v[4]$	$v[5]$	$v[6]$	$v[7]$	$v[8]$
71	**63**	46	80	39	92	55	14	27
63	**71**	**46**	80	39	92	55	14	27
63	46	**71**	**80**	39	92	55	14	27
63	46	71	**80**	**39**	92	55	14	27
63	46	71	39	**80**	**92**	55	14	27
63	46	71	39	80	**92**	**55**	14	27
63	46	71	39	80	55	**92**	**14**	27
63	46	71	39	80	55	14	**92**	**27**
63	46	71	39	80	55	14	27	**92**

Observações sobre o método

- Em cada fase, o maior item é deslocado para sua posição definitiva.
- Se uma lista tem n itens, após a primeira fase haverá n-1 itens a ordenar.
- Após, n-1 etapas haverá apenas 1 item a ordenar.
- Numa fase i são realizadas n-i comparações.

Ordenação completa

fase	*i*	*j*	*v*[0]	*v*[1]	*v*[2]	*v*[3]	*v*[4]
1º	1	0	**4 6**	**3 9**	5 5	1 4	2 7
		1	3 9	**4 6**	**5 5**	1 4	2 7
		2	3 9	4 6	**5 5**	**1 4**	2 7
		3	3 9	4 6	1 4	**5 5**	**2 7**
			3 9	4 6	1 4	2 7	**5 5**
2º	2	0	**3 9**	**4 6**	1 4	2 7	**5 5**
		1	3 9	**4 6**	**1 4**	2 7	**5 5**
		2	3 9	1 4	**4 6**	**2 7**	**5 5**
			3 9	1 4	2 7	**4 6**	**5 5**
3º	3	0	**3 9**	**1 4**	2 7	**4 6**	**5 5**
		1	1 4	**3 9**	**2 7**	**4 6**	**5 5**
			1 4	2 7	**3 9**	**4 6**	**5 5**
4º	4	0	**1 4**	**2 7**	**3 9**	**4 6**	**5 5**
			1 4	**2 7**	**3 9**	**4 6**	**5 5**

Algoritmo Bubble Sort

```
void bsort(int v[], int n) {
    int i, j;

    for(i=1; i<n; i++)
        for(j=0; j<n-i; j++)
            if( v[j]>v[j+1] ) {
                int x = v[j];
                v[j] = v[j+1];
                v[j+1] = x;
            }
}
```

Exercício

Utilize bsort(), para ordenar um vetor.

```
void main(void) {
    static int v[5]={46,39,55,27,16};
    int i;

    bsort(v,5);

    printf("\nVetor ordenado: ");

    for(i=0; i<5; i++)
        printf("%d ",v[i]);
}
```

Busca linear

- **Estratégia: examinar os itens um a um.**

```
int blin(int x, int v[], int n) {

   int i;

   for(i=0; i<n; i++)
      if( x == v[i] )
         return 1;

   return 0;
}
```

Busca binária

- **<u>Estratégia</u>: examinar primeiro o item do meio e, depois, desprezar metade dos itens (ordenados).**

Passo	x	i	f	m	$L[0]$	$L[1]$	$L[2]$	$L[3]$	$L[4]$	$L[5]$	$L[6]$	$L[7]$	$L[8]$
1º	63	0	8	4	14	27	39	46	**55**	63	71	80	92
2º	63	5	8	6						63	**71**	80	92
3º	63	5	5	5						**63**			

Fracasso da busca

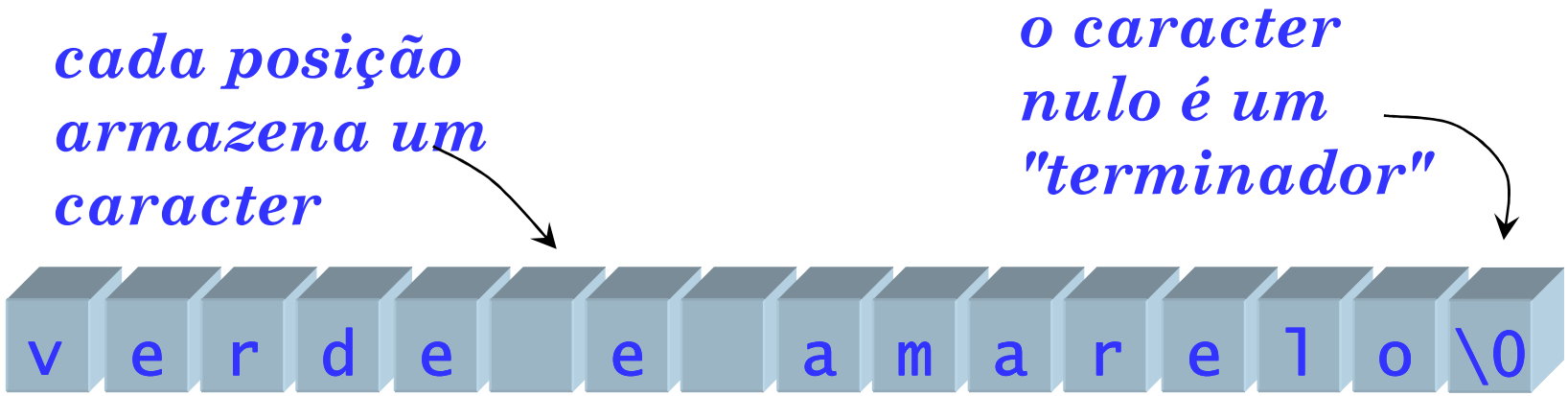
Passo	x	i	f	m	$L[0]$	$L[1]$	$L[2]$	$L[3]$	$L[4]$	$L[5]$	$L[6]$	$L[7]$	$L[8]$
1º	40	0	8	4	14	27	39	46	**55**	63	71	80	92
2º	40	0	3	1	14	**27**	39	46					
3º	40	2	3	2			**39**	46					
4º	40	3	3	3				46					
5º	40	**3**	**2**	?									

Algoritmo de busca binária

```
int bbin(int x, int v[], int n) {
    int i=0, f=n-1, m;
    while( i<=f ) {
        m = (i+f)/2;
        if( x == v[m] ) return 1;
        if( x <  v[m] ) f = m-1;
        else            i = m+1;
    }
    return 0;
}
```

Strings

- uma string é uma série de caracteres terminada com um byte nulo, representado por '\0';
- cuidado:
  - '\0' = ASCII 0
  - '0' = ASCII 48



Importante

- Não esquecer de reservar espaço para o ‘\0’.
- Para string constante, o espaço para o '\0' é alocado automaticamente pelo compilador.

```
#include <stdio.h>

void main(void) {
   printf("\nEspaço alocado = %d bytes",
          sizeof("verde e amarelo") );
}
```

Inicialização de string

```
#include <stdio.h>

void main(void) {
    char x[] = {'u', 'm'};
    char y[] = "dois";

    printf("%s %s", x, y);
}
```

Leitura de string via teclado

```
#include <stdio.h>

void main(void) {
    char n[21];

    printf("Qual o seu nome? ");

    gets(n);

    printf("Olá, %s!",n);
}
```

Manipulação de string

- Como string não é um tipo primitivo, operadores de atribuição e comparação não podem ser usados.

```
#include <stdio.h>

void main(void) {
    char x[] = "um";
    char y[] = "um";

    if( x==y ) printf("iguais");
    else printf("diferentes");
}
```

Exemplo: comparação entre strings

```
int strcmp(char s[], char t[]) {
    int i=0;

    while( s[i]==t[i] && s[i]!='\0' )
        i++;

    return s[i]-t[i];
}
```

Exemplo: atribuição entre strings

```
void strcpy(char s[], char t[]) {
    int i=0;
    while( s[i]=t[i] ) i++;
}

void main(void) {
    char a[10]="um", b[10] = "dois";
    printf("\nAntes: %s, %s",a,b);
    strcpy(a,b);
    printf("\nDepois: %s, %s",a,b);
}
```

Exercício

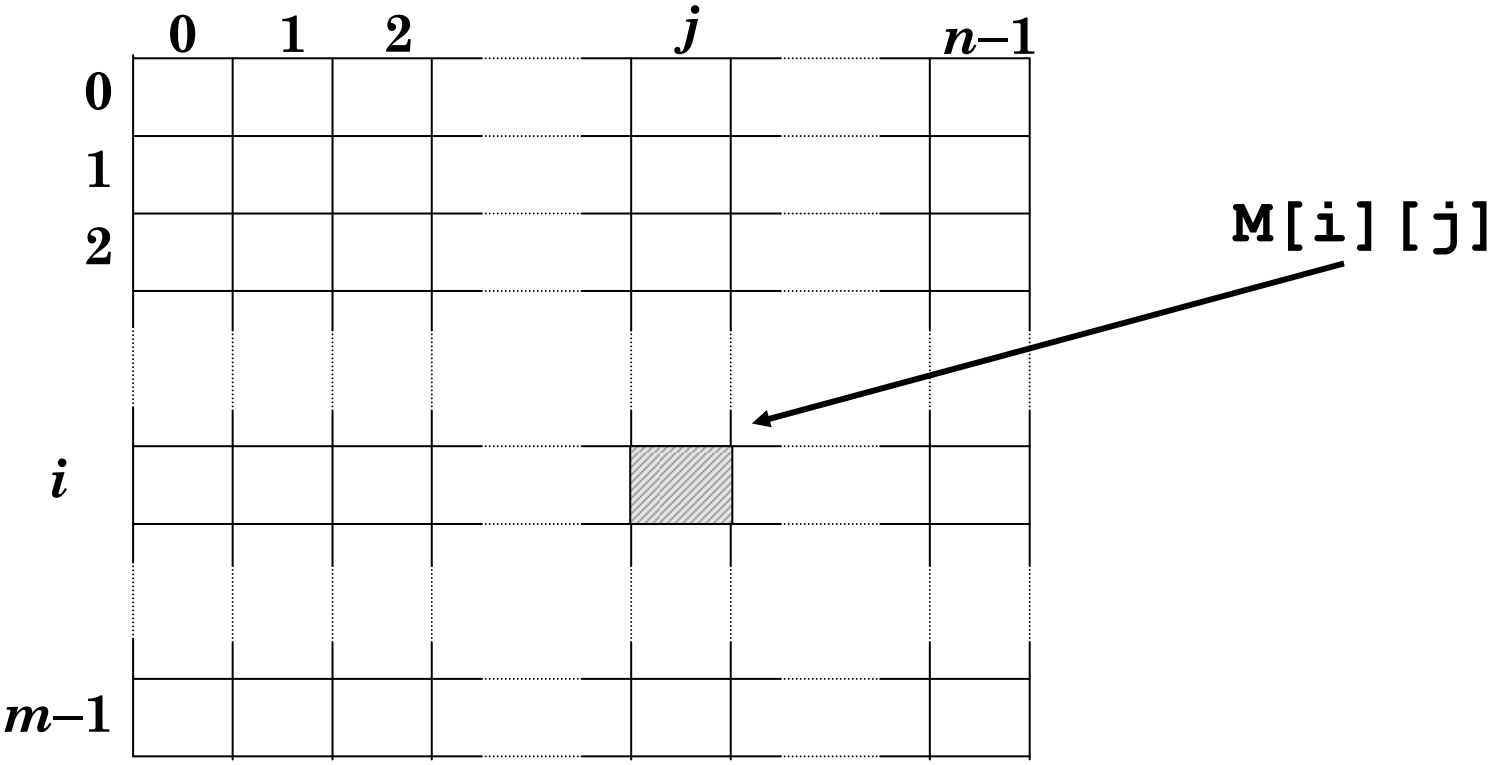
Codifique strlen(s), que devolve o comprimento de s.

```
int strlen(char s[]) {
    int i;
    for(i=0; s[i]!='\0'; i++ );
    return i;
}

void main(void) {
    char a[10]="teste";
    printf("\nComprimento: %d", strlen(a));
}
```

Matrizes

- Uma matriz é um vetor cujos itens são vetores.



Inicialização de matriz

```
static int labirinto[8][10] = {
        {1,1,1,1,1,1,1,1,1,1},
        {0,0,1,0,1,0,1,0,1,1},
        {1,0,1,0,1,0,0,0,0,1},
        {1,0,1,1,1,0,1,1,0,1},
        {1,0,0,0,0,0,1,0,0,1},
        {1,0,1,0,0,1,1,0,1,1},
        {1,0,0,1,0,1,0,0,0,0},
        {1,1,1,1,1,1,1,1,1,1}
};
```

Matriz de strings

```
#include <stdio.h>

void main(void) {
    char n[5][11];
    int i;
    for(i=0; i<5; i++) {
        printf("%dº nome: ",i+1);
        gets(n[i]);
    }
    ...
}
```

Matrizes como parâmetro

- O nome da matriz representa seu endereço;
- matrizes são passadas por referência;
- a segunda dimensão é necessária para calcular os endereços dos itens.

Exercício

Adapte bsort() para ordenar um vetor de strings.

```
void bsort(int v[][20], int n) {
    int i, j;

    for(i=1; i<n; i++)
        for(j=0; j<n-i; j++)
            if( strcmp(v[j],v[j+1])>0 ) {
                char x[20];
                strcpy(x, v[j]);
                strcpy(v[j], v[j+1]);
                strcpy(v[j+1], x);
            }
}
```

Exercício

Crie um programa para jogar o "jogo-da-velha".

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

void main(void) {
   static char t[3][3] = {
      {' ', ' ', ' '},
      {' ', ' ', ' '},
      {' ', ' ', ' '}
   };
   int j=0, e, s;
   char v;
```

Programa principal

```
clrscr();
printf("\nPar (0) ou impar (1)?");
scanf("%d",&e);
randomize();
s = random(2);
do {
   mostra(t);
   if( e==s ) usuario(t);
   else computador(t);
   v = vencedor(t);
   s = !s;
} while( ++j<9 && v==' ');
```

Programa principal

```
  mostra(t);

  switch( v ) {
    case ' ': puts("\nEmpate"); break;
    case 'x': puts("\nVoce venceu"); break;
    case 'o' : puts("\nEu venci"); break;
  }

  getch();
}
```

Exibindo o tabuleiro

```
void mostra(char t[3][3]) {
    int i;
    clrscr();
    for(i=0; i<3; i++) {
      printf("\n %c | %c | %c ",
              t[i][0], t[i][1], t[i][2] );
      if( i<2 )
        printf("\n---+---+---");
    }
}
```

Exibindo o tabuleiro

```
void mostra(char t[3][3]) {
   clrscr();
   printf("\n %c | %c | %c ",
          t[0][0], t[0][1], t[0][2] );
   printf("\n---+---+---");
   printf("\n %c | %c | %c ",
          t[1][0], t[1][1], t[1][2] );
   printf("\n---+---+---");
   printf("\n %c | %c | %c ",
          t[2][0], t[2][1], t[2][2] );
}
```

Jogada do usuário

```
void usuario(char t[3][3]) {
   int L, C;

   do {
      gotoxy(1,10);
      printf("Posicao? ");
      scanf("%d,%d",&L,&C);
   } while( L<0 || L>2 ||
            C<0 || C>2 ||
            t[L][C]!=' ');

   t[L][C] = 'x';
}
```

Jogada do computador

```
void computador(char t[3][3]) {
    int L, C;

    do {
        L = random(3);
        C = random(3);
    } while( t[L][C] != ' ' );

    t[L][C] = 'o';
}
```

Vencedor

```
char vencedor(char t[3][3]) {
   int i;

   for(i=0; i<3; i++)
     if( t[i][0]==t[i][1] &&
         t[i][1]==t[i][2] && t[i][0]!= ' ')
         return t[i][0];

   for(i=0; i<3; i++)
     if( t[0][i]==t[1][i] &&
         t[1][i]==t[2][i] && t[i][0]!= ' ')
          return t[0][i];
```

Vencedor

```
    if( t[0][0]==t[1][1] &&
        t[1][1]==t[2][2] && t[i][0]!= ' ')
        return t[0][0];

    if( t[0][2]==t[1][1] &&
        t[1][1]==t[2][0] && t[i][0]!= ' ')
        return t[0][2];

    return ' ';
}
```

Exercício para avaliação (EP2)

Modificar a função computador() para implementar a seguinte estratégia:

1o. Computador tenta ganhar o jogo.

2o. Computador tenta impedir adversário de ganhar.

3o. Computador faz uma jogada aleatória.

<u>Bônus</u>: em vez de fazer uma jogada aleatória, o computador pode fazer a "melhor jogada".

Fim