sequências de caracteres
Strings

Nada é tão fácil quanto parece, nem tão difícil quanto a explicação do manual.

Lei de Murphy

Sumário

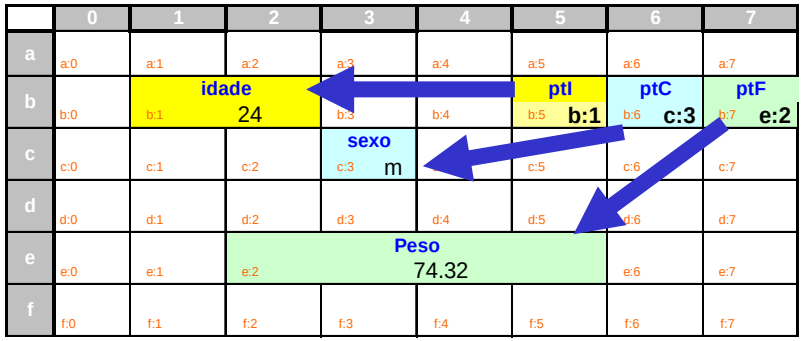
- Revisões sobre ponteiros
- O que é uma string
- Caracteres e strings
- Declaração de strings
- O caracter terminador ‘\0’
- Incialização automática de strings
- Vectores de caracteres e strings
- Leitura e escrita de strings
- Passagem de strings para funções
- Principais funções de manipulação de strings
- Biblioteca <string.h>
- Vectores de strings
- Parametros da linha de comando

Apontadores em C

revisões

Ponteiros

```
<tipo_variavel> *<nome>;
<tipo_variavel> *<nome> = endereço;
```



Declaração

```
int idade;
char sexo;
float peso;
int *ptI;
char *ptC;
float *ptF;
```

Atribuição

```
idade =24;
sexo = 'm';
peso=74.32;
*ptI = &idade;
*ptC = &sexo;
*ptF = &peso;
```

dados

```
idade = 18;
*pti = 18;
```

O operador & devolve o endereço de uma variável
O nome de uma variável devolve o seu conteúdo

O operador * devolve o conteúdo apontado por um apontador

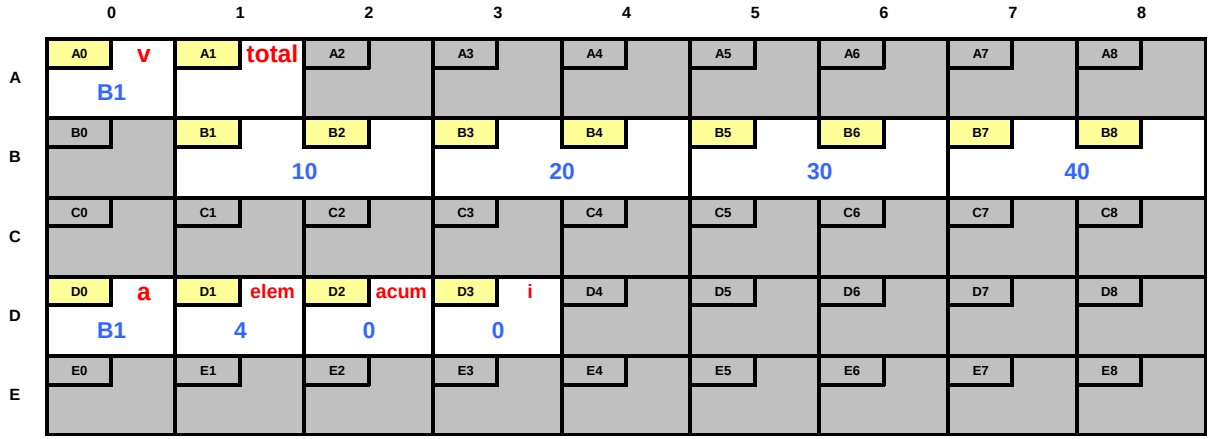
Apontadores e vectores

Programa

```
int v[4] = {10 , 20, 30, 40 };
int total = soma( v , 4 );
```

Funçoes

```
Int soma( int a[] , int elem){
  int acum =0, i;
  for( i=0 ; i< elem ; i++)
    acum += a[i];
  return acum;
}
```



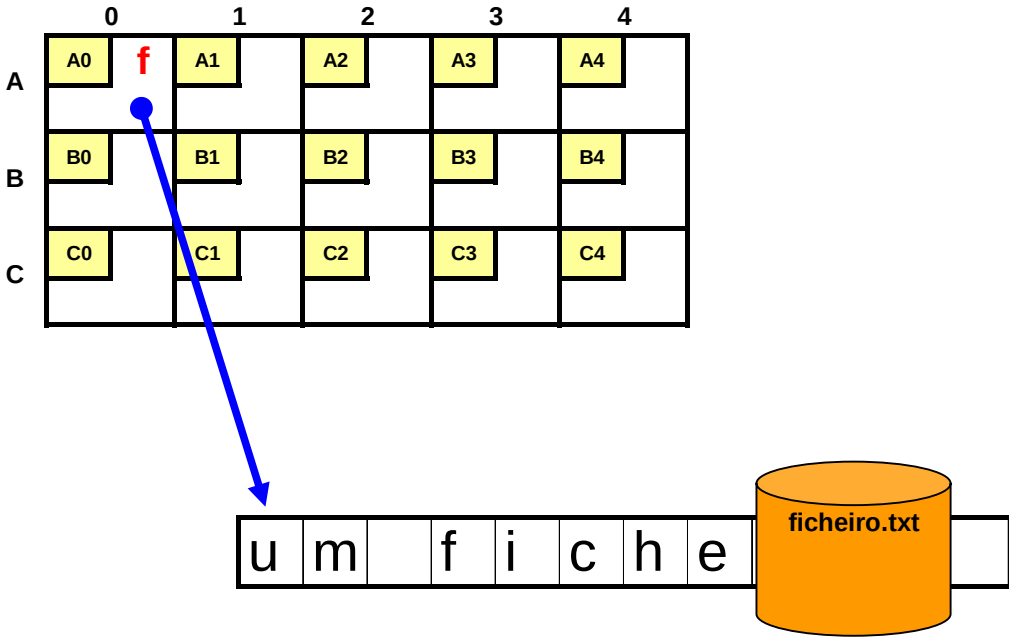
NOTA:

O nome de um vector corresponde ao endereço do primeiro dos seus elementos

NOTA:
Nos parâmetros das funções
não é necessário o & porque o vector é um ponteiro

Ficheiros

- FILE *f
  - Streams de periféricos
- Utilização de streams
  - Abrir a stream – fopen
    - Texto / binário
    - Leitura / escrita
  - Ler
    - fget , fscanf , fread
  - Escrever
    - fputc, fprinf , fwrite
  - Movimentar
    - fseek
  - Fechar a stream
    - Esvaziar o buffer - fflush
    - Fechar a stream - close



Strings

Ponteiros para caracteres

Motivação

- A linguagem C
  - manipula com facilidade
    - Inteiros
    - Reais
    - Caracteres

Manipulação de reais

```
float n1,n2,n3;
printf("Introduza dois números:");
scanf("%f %f",&n1, &n2);
n3 = n1 + n2;
printf(" Soma = %f ", n3);
```

  - Limitações na manipulação
    - vectores
      - unidimensionais
    - matrizes
      - bidimensionais

Manipulação de Vectores

- **Escrever funções**
  - **ler vectores**
  - **escrever vectores**
  - **somar vectores**
- **Chamar funções**

Manipulação de vectores

Ler Vector

```
void LerVector( int v[] , int elementos){
  int i;
  for( i=0 ; i < elementos ; i++){
    printf(" %d elem :", i+1);
    scanf("%d",&v[i]);
  }
}
```

Escrever Vector

```
void EscreverVector( int v[] , int elementos){
  int i;
  for( i=0 ; i < elementos ; i++){
    printf(" %d ", v[i]);
  }
}
```

Manipulação de vectores

Soma Vector

```
void SomaVector( int v1[] ,int v2[] ,int soma[] , int elementos){
  int i;
  for( i=0 ; i < elementos ; i++){
     soma[i] = v1[i] + v2[i];
  }
}
```

Programa

```
. . .
int vector1[5];
 int vector2[5];
 int soma[5];
 LerVector( vector1, 5);
 LerVector( vector2, 5);
 SomaVector( vector1, vector2,
soma, 5);
 EscreverVector( soma, 5);. . .
```

Exercício

Exercício

Construa um programa que solicite o nome do utilizador e imprima um cumprimento com o seu seu nome

Resultado

```
Nome : António
Olá António
```

Programa

```
int main(int argc, char* argv[])
{
   char nome[80]
   LerVectorCaracteres(nome, 80);
   printf("Ola ");
   EscreveVectorCaracteres(nome);
   return 0;
}
```

- Problemas
  - LerVectorCaracteres
  - EscreveVectorCaracteres
  - Número variável de caracteres no nome
  - O array de caracteres é uma estrutura muito utilizada

Definição de string

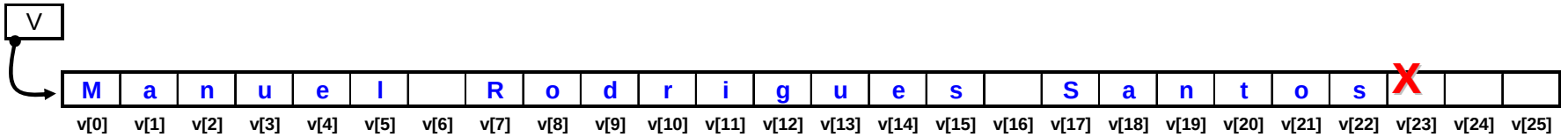
String

Uma string é um conjunto de caracteres armazenados num vector

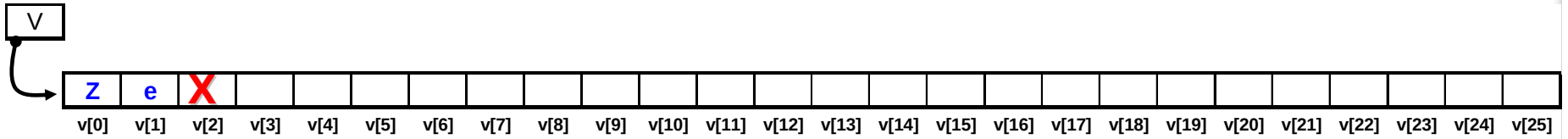
- Vector de caracteres com facilidade de:
  - Leitura
    - scanf("%s",variavel);
    - gets(variavel);
  - Escrita
    - printf("%s",variavel);
    - puts(variavel);
  - Inicialização
    - char nome[80] = “Bolo de chocolate”
    - char *disciplina = “Introdução à programação”

Vectores de caracteres dinâmicos

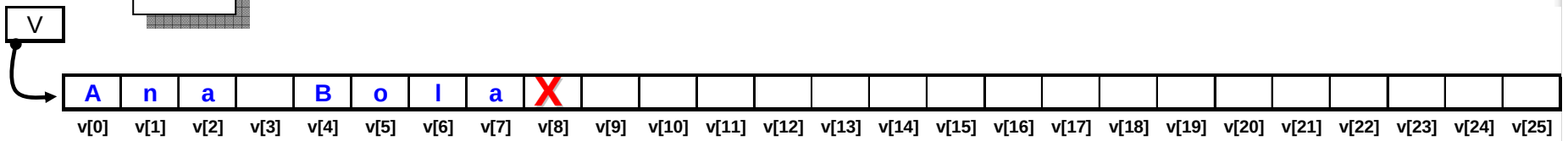
Nome de uma Pessoa => char v[25];



Manuel Rodrigues Santos



Ze



Ana Bola

Que caracter escolher ?

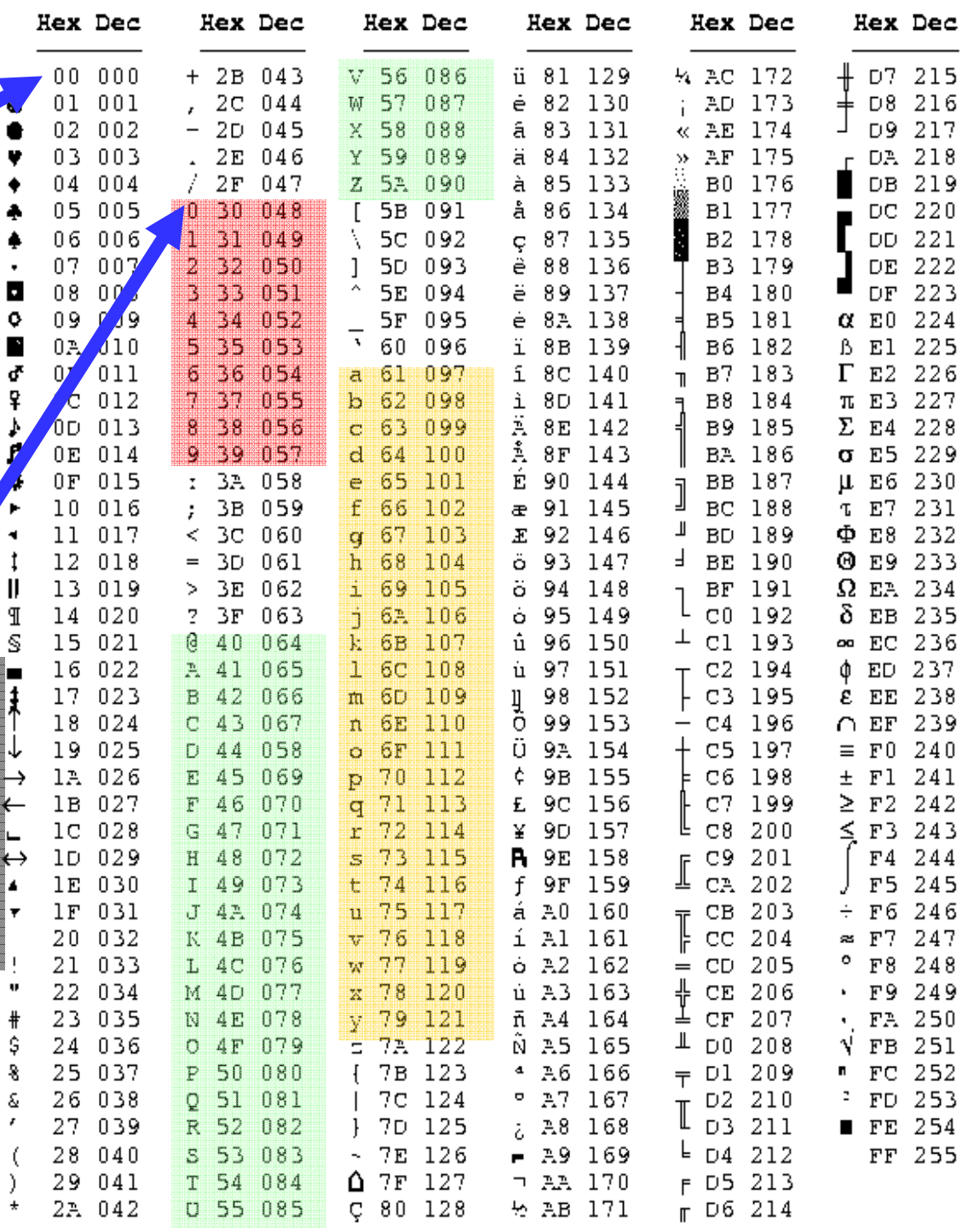
Que caracter ?

- Caracter terminador
  - '\0'

Nota

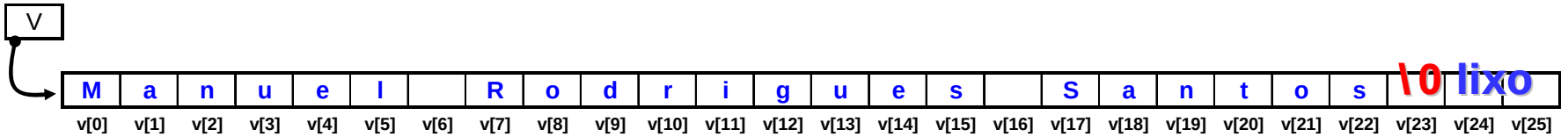
'\0' – caracter com o valor Ascii 0 (zero)

'0' – caracter com o valor Ascii 38

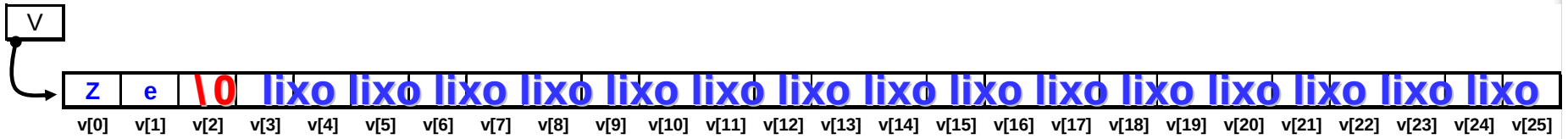


Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec
00	000	+ 2B	043	V 56	086	ü 81	129	¼ AC	172	╫ D7	215
01	001	, 2C	044	W 57	087	é 82	130	¡ AD	173	╪ D8	216
02	002	- 2D	045	X 58	088	â 83	131	« AE	174	┘ D9	217
03	003	. 2E	046	Y 59	089	ä 84	132	» AF	175	┌ DA	218
04	004	/ 2F	047	Z 5A	090	à 85	133	░ B0	176	▌ DB	219
05	005	0 30	048	[5B	091	å 86	134	▒ B1	177	▐ DC	220
06	006	1 31	049	\ 5C	092	ç 87	135	▓ B2	178	▄ DD	221
07	007	2 32	050	] 5D	093	ê 88	136	│ B3	179	▀ DE	222
08	008	3 33	051	^ 5E	094	ë 89	137	┤ B4	180	DF	223
09	009	4 34	052	_ 5F	095	è 8A	138	╡ B5	181	α E0	224
0A	010	5 35	053	` 60	096	ï 8B	139	╢ B6	182	ß E1	225
0B	011	6 36	054	a 61	097	î 8C	140	╖ B7	183	Γ E2	226
0C	012	7 37	055	b 62	098	ì 8D	141	╕ B8	184	π E3	227
0D	013	8 38	056	c 63	099	Ä 8E	142	╣ B9	185	Σ E4	228
0E	014	9 39	057	d 64	100	Å 8F	143	║ BA	186	σ E5	229
0F	015	: 3A	058	e 65	101	É 90	144	╗ BB	187	µ E6	230
10	016	; 3B	059	f 66	102	æ 91	145	╝ BC	188	τ E7	231
11	017	< 3C	060	g 67	103	Æ 92	146	╜ BD	189	Φ E8	232
12	018	= 3D	061	h 68	104	ô 93	147	╛ BE	190	Θ E9	233
13	019	> 3E	062	i 69	105	ö 94	148	┐ BF	191	Ω EA	234
14	020	? 3F	063	j 6A	106	ò 95	149	└ C0	192	δ EB	235
15	021	@ 40	064	k 6B	107	û 96	150	┴ C1	193	∞ EC	236
16	022	A 41	065	l 6C	108	ù 97	151	┬ C2	194	φ ED	237
17	023	B 42	066	m 6D	109	ÿ 98	152	├ C3	195	ε EE	238
18	024	C 43	067	n 6E	110	Ö 99	153	─ C4	196	∩ EF	239
19	025	D 44	058	o 6F	111	Ü 9A	154	┼ C5	197	≡ F0	240
1A	026	E 45	069	p 70	112	¢ 9B	155	╞ C6	198	± F1	241
1B	027	F 46	070	q 71	113	£ 9C	156	╟ C7	199	≥ F2	242
1C	028	G 47	071	r 72	114	¥ 9D	157	╚ C8	200	≤ F3	243
1D	029	H 48	072	s 73	115	₧ 9E	158	╔ C9	201	⌠ F4	244
1E	030	I 49	073	t 74	116	ƒ 9F	159	╩ CA	202	⌡ F5	245
1F	031	J 4A	074	u 75	117	á A0	160	╦ CB	203	÷ F6	246
20	032	K 4B	075	v 76	118	í A1	161	╠ CC	204	≈ F7	247
21	033	L 4C	076	w 77	119	ó A2	162	═ CD	205	° F8	248
22	034	M 4D	077	x 78	120	ú A3	163	╬ CE	206	∙ F9	249
23	035	N 4E	078	y 79	121	ñ A4	164	╧ CF	207	· FA	250
24	036	O 4F	079	z 7A	122	Ñ A5	165	╨ D0	208	√ FB	251
25	037	P 50	080	{ 7B	123	ª A6	166	╤ D1	209	ⁿ FC	252
26	038	Q 51	081	| 7C	124	º A7	167	╥ D2	210	² FD	253
27	039	R 52	082	} 7D	125	¿ A8	168	╙ D3	211	■ FE	254
28	040	S 53	083	~ 7E	126	⌐ A9	169	╘ D4	212	FF	255
29	041	T 54	084	⌂ 7F	127	¬ AA	170	╒ D5	213		
2A	042	U 55	085	Ç 80	128	½ AB	171	╓ D6	214		

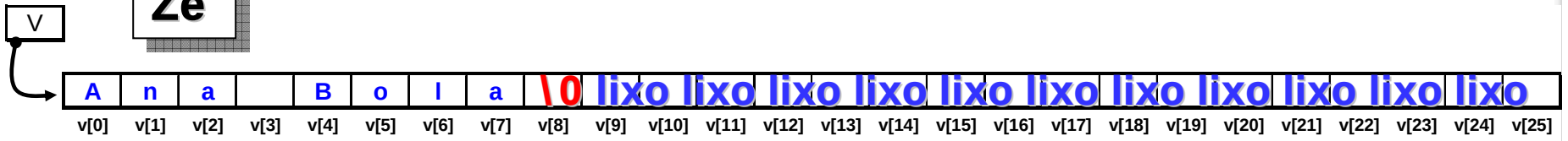
Caracter terminador '\0'



Manuel Rodrigues Santos



Ze



Ana Bola

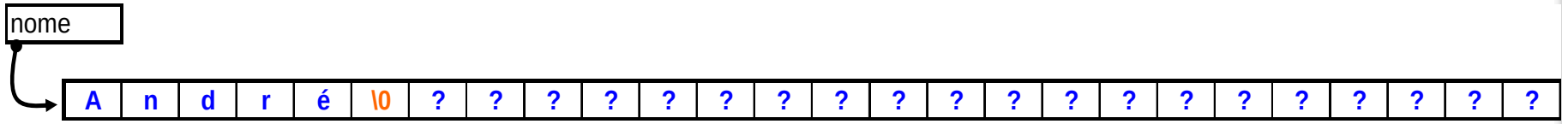
Definição de String

Uma string é um conjunto de caracteres armazenados num vector de caracteres que termina por '\0'

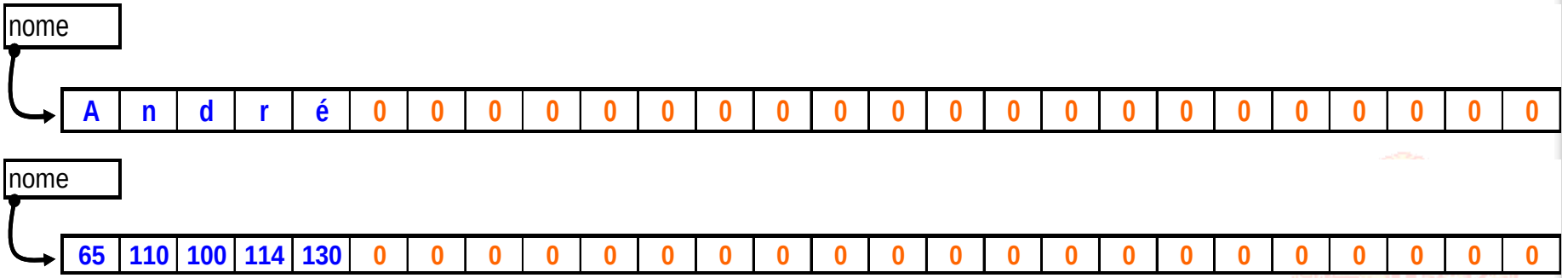
Inicialização automática de strings

Definição do tamanho da string

```
char nome[25] = "André"
```



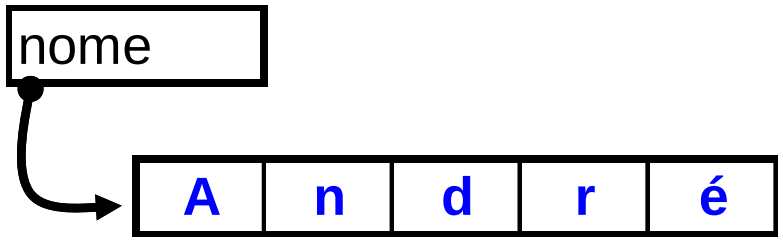

```
char nome[25] = {'A','n','d','r','é'}
```

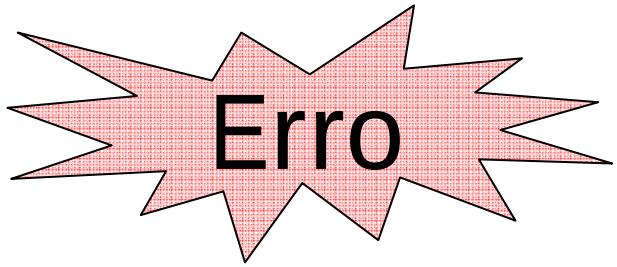


Inicialização automática de strings

Omissão do tamanho da string

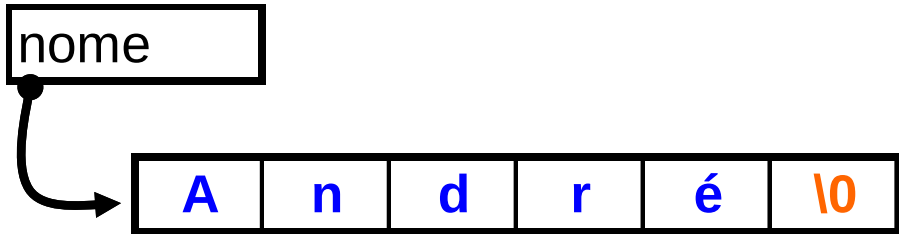
```
char nome[] = {'A','n','d','r','é'}
```






```
char nome[] = "André"
```

```
char *nome = "André"
```



Nota

Um vector de caracteres pode não conter uma string

Leitura e escrita de Strings

- Leitura
  - scanf
- Escrita
  - printf

Exercício

Construa um programa que solicite o nome e o apelido do utilizador e imprima um cumprimento com o seu seu nome completo

Programa

```
int main(int argc, char* argv[]){
char nome[20];
 char apelido[20];
 printf("Qual o seu nome :");
 scanf("%s",nome);
 printf("\nQual o seu apelido :");
 scanf("%s",apelido);
 printf("\nOla %s %s ", nome, apelido);
}
```

Resultado

```
Nome : Manuel
Apelido: Rodrigues
Olá Manuel Rodrigues
```

Leitura e escrita de Strings –Nova tentativa

Programa

```
int main(int argc, char* argv[]){
char nome[20];
char apelido[20];
printf("\nQual o seu nome :");
scanf("%s",nome);
printf("\nQual o seu apelido :");
scanf("%s",apelido);
printf("\nOla %s %s", nome, apelido);
}
```

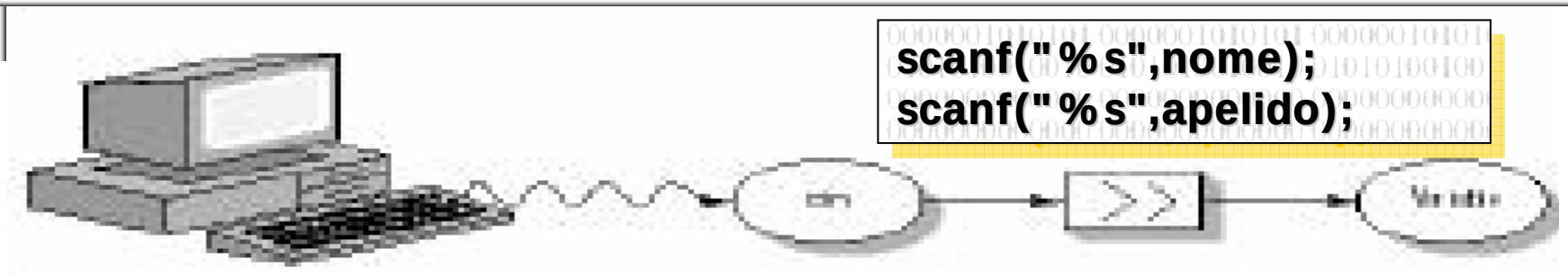
Nova Tentativa

Resultado

Nome : Antonio Manuel
Apelido:

Olá Antonio Manuel

Leitura e escrita de Strings



A	n	t	ó	n	l	0		M	a	n	u	e	l	\n'

Nota

O Caracter espaço serve para separar a entrada de variáveis no scanf

nome

A	n	t	ó	n	l	0	\0'

Apelido

M	a	n	u	e	l	\0'

Função gets

gets(variavel)

Programa

```
int main(int argc, char* argv[]){
 char nome[20];
  char apelido[20];
  printf("Qual o seu nome :");
  gets(nome);
  printf("\nQual o seu apelido :");
  gets(apelido);
  printf("\nOla %s %s ", nome, apelido);
}
```

Resultado

```
Nome : Antonio Manuel
Apelido: Rodrigues Manso

Olá Antonio Manuel Rodrigues Manso
```

Função GetLine – novo Teste

Programa

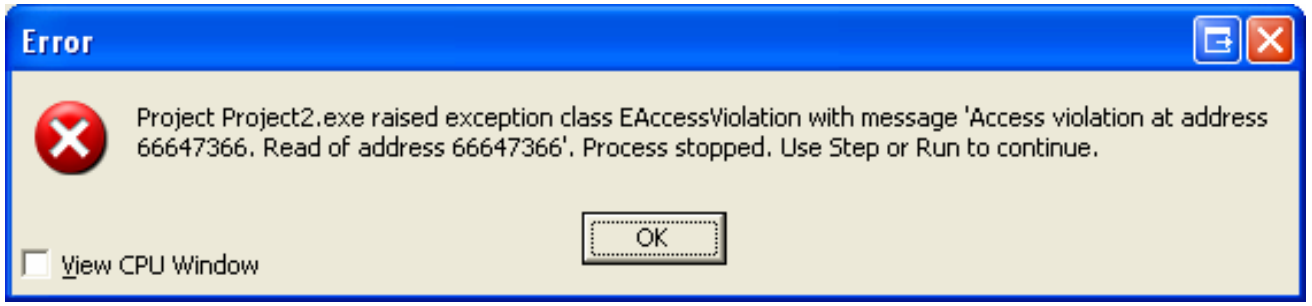
```
int main(int argc, char* argv[]){
  char nome[20];
  char apelido[20];
  cout<<"Nome:";
  cin.getline(nome,19);
  cout<<"Apelido:";
  cin.getline(apelido,19);
```

Nova Tentativa

Resultado

Nome:Tranquilino da Conceicao Peixoto
Apelido: Trindade de Almeida Garrett

Ola Tranquilino da Con Trindade de Almeida Garrett



Strings e Funções

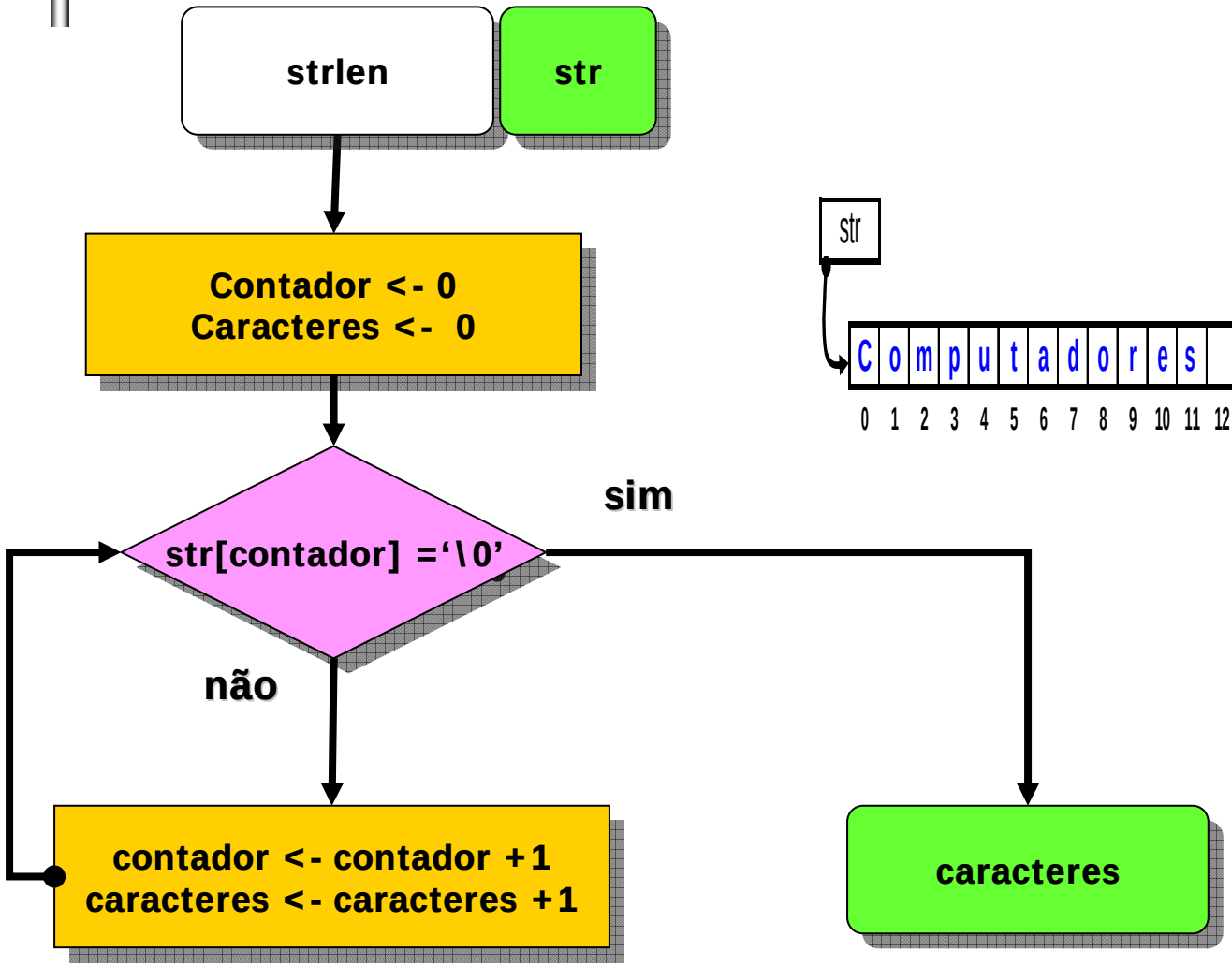
exemplos

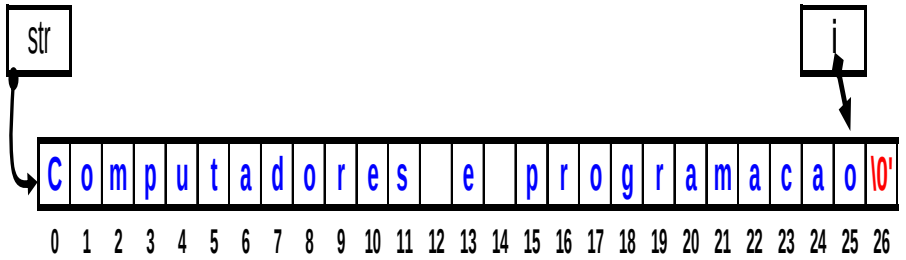
```
int strlen( char *string)

int strcmp( char *str1, char *str2)

char * strcpy( char *destino, char *origem)
```

Nota

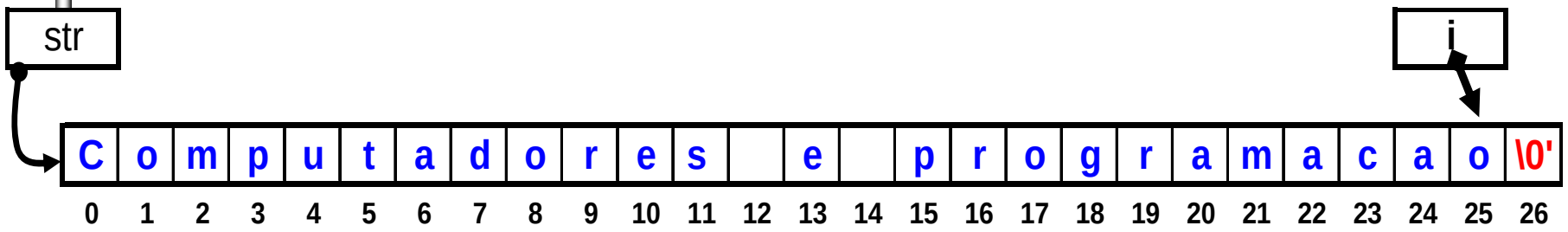
Não é necessário passar o nº de elementos porque a string termina por '\0'

Tamanho da string - fluxograma



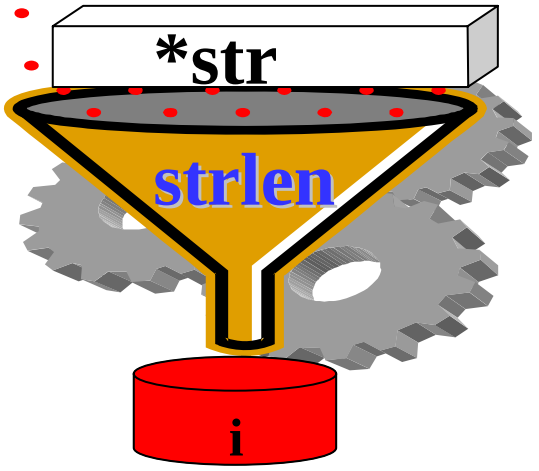


Tamanho da string



Contar o número de caracteres

```
int strlen(char *str)
{
 int i=0;
 while( str[i] != '\0' )
      i++;
 return i;
}
```



Programa

```
char *nome="Computadores e Programacao";
 printf( "\nCaracteres : %d " , strlen(nome) );
```

Resultado

```
Caracteres : 26
```

Esta Vazia

Versão 1

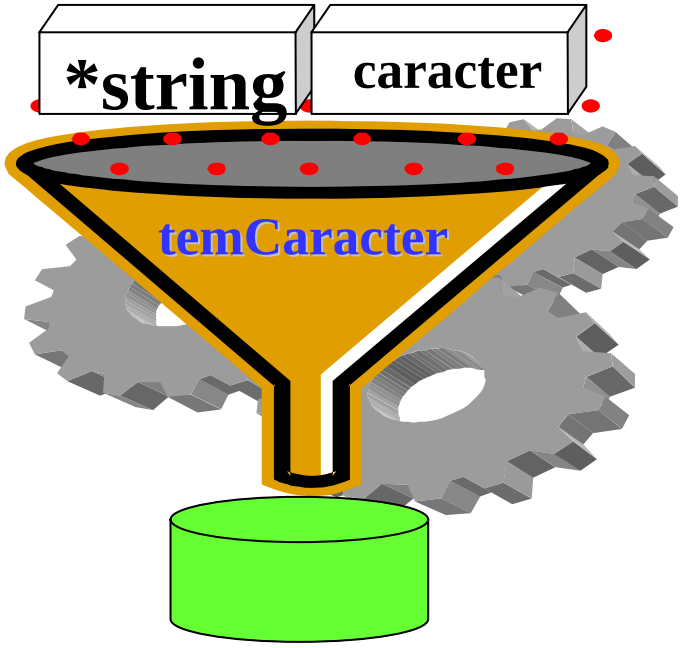
```
int estaVazia(char *str)
{
    return str[0] != '\0';
}
```

Versão 2

```
int  estaVazia(char *str)
{
    return strLen(str)==0;
}
```

Tem Caracter

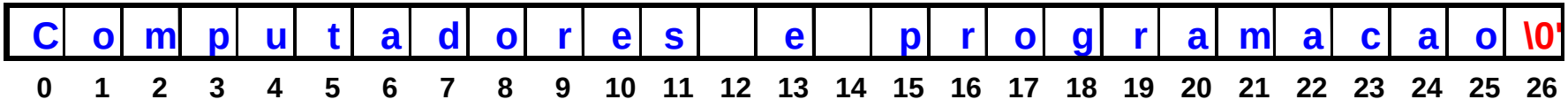
C	o	m	p	u	t	a	d	o	r	e	s		e		p	r	o	g	r	a	m	a	c	a	o	\0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26



Programa

```
int main(int *argc, char *argv[])
{
 char *nome="Computadores e Programacao";
 if( temCaracter( nome , 'a' ) )
    printf(" tem o caracter a ");
 else
    printf(" Não tem o caracter a ");
}
```

Tem Caracter

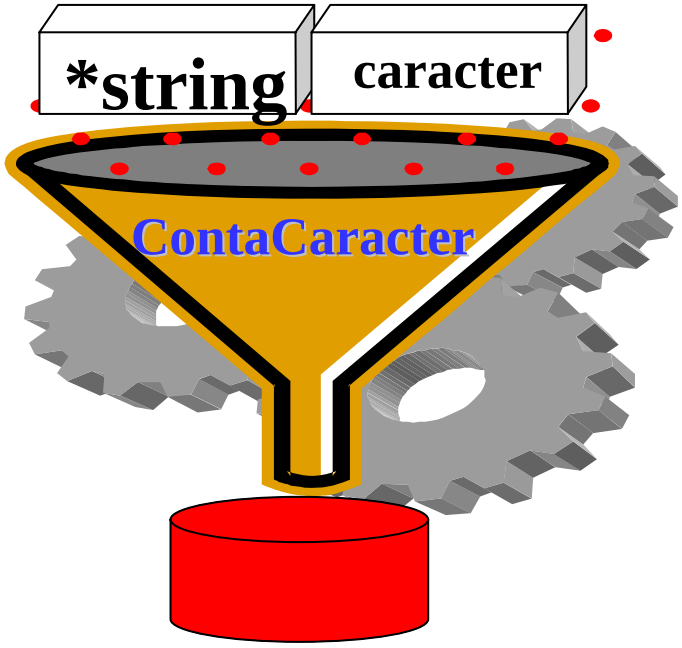


Verifica se a string tem o caracter

```
int  temCaracter(char *str, char ch)
{
 int i=0;
 while( str[i] != '\0')
 {
   if ( str[i] == ch)
     return 1;
   i++;
 }
 return 0;
}
```

ContaCaracter

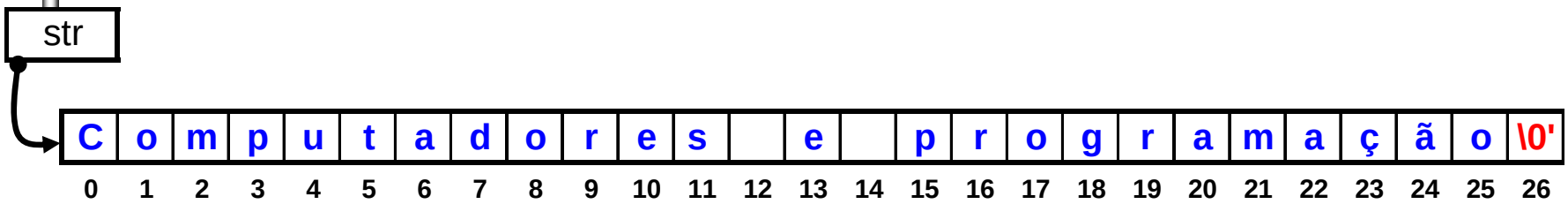
C	o	m	p	u	t	a	d	o	r	e	s		e		p	r	o	g	r	a	m	a	c	a	o	\0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26



Programa

```
int main(int *argc, char *argv[])
{
  char *nome="Computadores e Programacao";
  int num = ContaCarateres( nome, 'a');
  printf("a string tem  %d as",num);
}
```

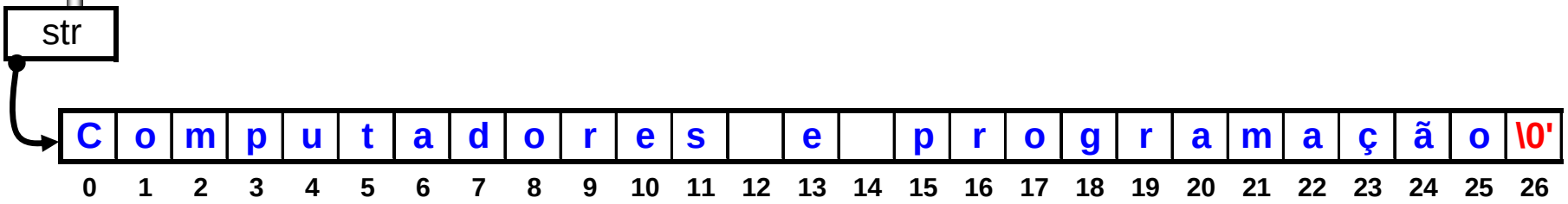
Conta número de Caracteres



Conta o número de caracteres iguais ao parâmetro

```
int  contaCaracter(char *str, char ch)
{
 int contador =0;
 int i=0;
 while( str[i] !='\0')
 {
    if ( str[i] == ch)
      contador ++;
    i++;
 }
 return contador;
}
```

Contar Espaços



Conta o número de espacos

```
int contaEspacao(char *str)
{
   return numCaracteres(str,‘ ‘);
}
```

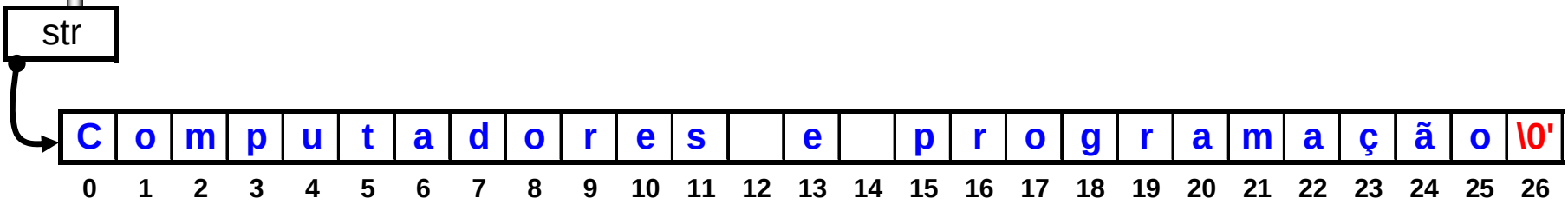
Programa

```
char *nome="Computadores e Programacao";
Printf("\nNum Espacos :  %d“, contaEspacos(nome) );
```

Resultado

```
Num Espacos : 2
```

Contar Vogais



Conta o número de vogais

```
int contaVogais(char *str)
{
return contaCaracter(str,'a') +
       contaCaracter(str,'e') +
       contaCaracter(str,'i') +
       contaCaracter(str,'o') +
       contaCaracter(str,'u') ;
}
```

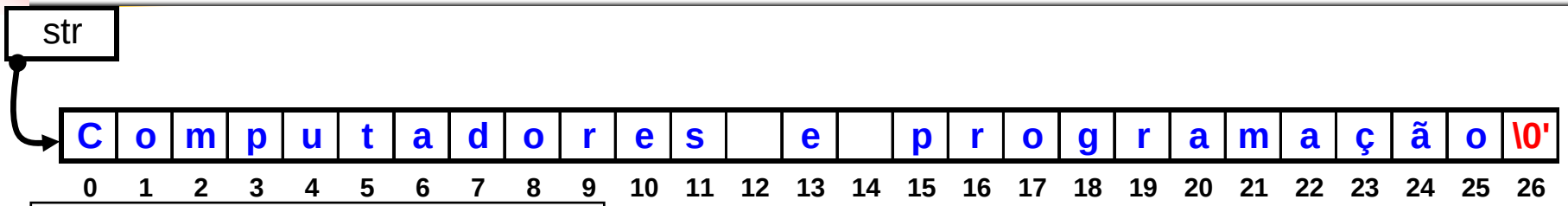
Programa

```
char *nome="Computadores e Programação";
Printf("\nNum Vogais :  %d", contaVogais(nome) );
```

Resultado

```
Num Vogais : 10
```

Contar Vogais



Conta o número de vogais

```
int contaVogais(char *str)
{
  char vogais[]="aeiouAEIOUáéíóúàèìòùãõêÁÉÍÓÚÀÈÌÒÙÃÕÊ";
  int numVogais=0;
  for(int i=0   ;str[i]!='\0' ; str++ )
     if ( temCaracter(vogais , str[i]) )
        numVogais ++;
  return numVogais;
}
```

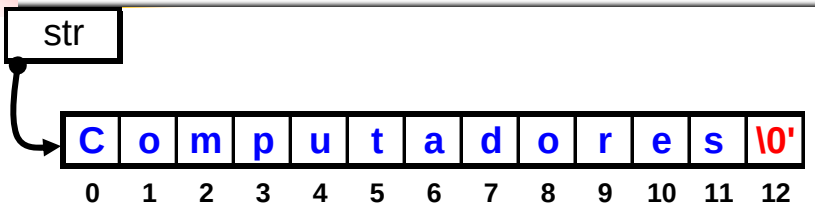
Programa

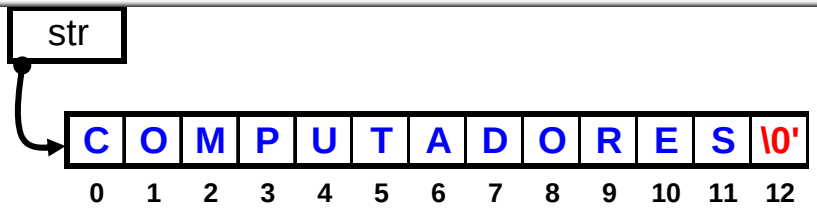
```
char *nome="Computadores e Programação";
Printf("\nNum Vogais :  %d", contaVogais(nome) );
```

Resultado

```
Num Vogais : 11
```

Maiúsculas





Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec
00	000	+ 2B	043	V 56	086	ü 81	129	¼ AC	172	╫ D7	215
01	001	, 2C	044	W 57	087	é 82	130	¡ AD	173	╪ D8	216
02	002	- 2D	045	X 58	088	â 83	131	« AE	174	┘ D9	217
03	003	. 2E	046	Y 59	089	ä 84	132	» AF	175	┌ DA	218
04	004	/ 2F	047	Z 5A	090	à 85	133	▒ B0	176	▒ DB	219
05	005	0 30	048	[5B	091	å 86	134	▒ B1	177	▒ DC	220
06	006	1 31	049	\ 5C	092	ç 87	135	▒ B2	178	▒ DD	221
07	007	2 32	050	] 5D	093	ê 88	136	▒ B3	179	▒ DE	222
08	008	3 33	051	^ 5E	094	ë 89	137	▒ B4	180	▒ DF	223
09	009	4 34	052	_ 5F	095	è 8A	138	▒ B5	181	α E0	224
0A	010	5 35	053	` 60	096	ï 8B	139	▒ B6	182	ß E1	225
0B	011	6 36	054	a 61	097	î 8C	140	▒ B7	183	Γ E2	226
0C	012	7 37	055	b 62	098	ì 8D	141	▒ B8	184	π E3	227
0D	013	8 38	056	c 63	099	Ä 8E	142	▒ B9	185	Σ E4	228
0E	014	9 39	057	d 64	100	Å 8F	143	▒ BA	186	σ E5	229
0F	015	: 3A	058	e 65	101	É 90	144	▒ BB	187	µ E6	230
10	016	; 3B	059	f 66	102	æ 91	145	▒ BC	188	τ E7	231
11	017	< 3C	060	g 67	103	Æ 92	146	▒ BD	189	Φ E8	232
12	018	= 3D	061	h 68	104	ô 93	147	▒ BE	190	Θ E9	233
13	019	> 3E	062	i 69	105	ö 94	148	▒ BF	191	Ω EA	234
14	020	? 3F	063	j 6A	106	ò 95	149	▒ C0	192	δ EB	235
15	021	@ 40	064	k 6B	107	û 96	150	▒ C1	193	∞ EC	236
16	022	A 41	065	l 6C	108	ù 97	151	▒ C2	194	φ ED	237
17	023	B 42	066	m 6D	109	ÿ 98	152	▒ C3	195	ε EE	238
18	024	C 43	067	n 6E	110	Ö 99	153	▒ C4	196	∩ EF	239
19	025	D 44	058	o 6F	111	Ü 9A	154	▒ C5	197	≡ F0	240
1A	026	E 45	069	p 70	112	¢ 9B	155	▒ C6	198	± F1	241
1B	027	F 46	070	q 71	113	£ 9C	156	▒ C7	199	≥ F2	242
1C	028	G 47	071	r 72	114	¥ 9D	157	▒ C8	200	≤ F3	243
1D	029	H 48	072	s 73	115	₧ 9E	158	▒ C9	201	⌠ F4	244
1E	030	I 49	073	t 74	116	ƒ 9F	159	▒ CA	202	⌡ F5	245
1F	031	J 4A	074	u 75	117	á A0	160	▒ CB	203	÷ F6	246
20	032	K 4B	075	v 76	118	í A1	161	▒ CC	204	≈ F7	247
21	033	L 4C	076	w 77	119	ó A2	162	▒ CD	205	° F8	248
22	034	M 4D	077	x 78	120	ú A3	163	▒ CE	206	∙ F9	249
23	035	N 4E	078	y 79	121	ñ A4	164	▒ CF	207	· FA	250
24	036	O 4F	079	z 7A	122	Ñ A5	165	▒ D0	208	√ FB	251
25	037	P 50	080	{ 7B	123	ª A6	166	▒ D1	209	ⁿ FC	252
26	038	Q 51	081	| 7C	124	º A7	167	▒ D2	210	² FD	253
27	039	R 52	082	} 7D	125	¿ A8	168	▒ D3	211	■ FE	254
28	040	S 53	083	~ 7E	126	⌐ A9	169	▒ D4	212	FF	255
29	041	T 54	084	⌂ 7F	127	¬ AA	170	▒ D5	213		
2A	042	U 55	085	Ç 80	128	½ AB	171	▒ D6	214		

Maiúsculas

toupper

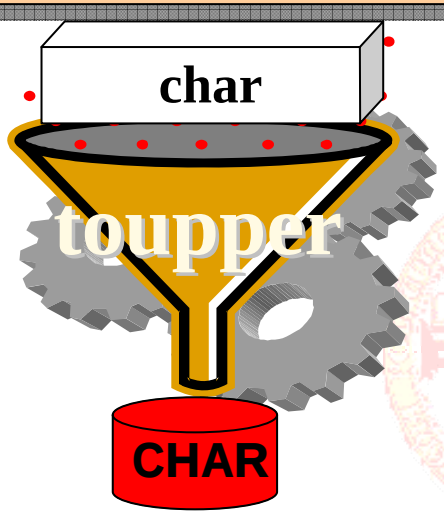
toupper is a function that converts an integer ch (in the range EOF to 255) to its uppercase value (A to Z; if it was lowercase, a to z). All others are left unchanged

tolower

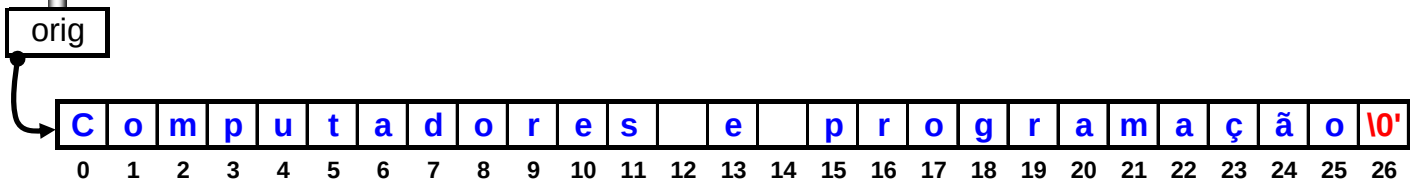
tolower is a function that converts an integer ch (in the range EOF to 255) to its lowercase value (a to z; if it was uppercase, A to Z). All others are left unchanged.

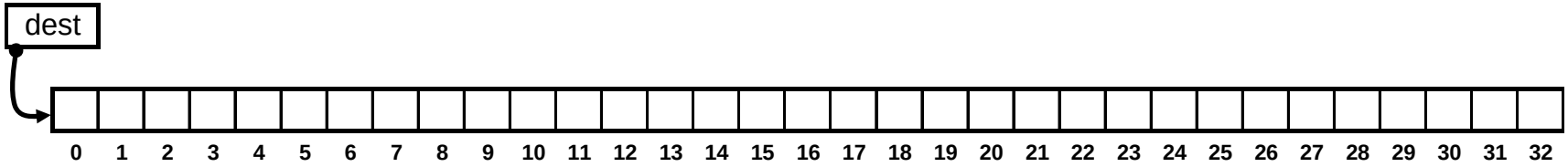
Conversão para maiusculas

```
void maiusculas(char *str)
{
   for( int i=0  ; str[i] !='\0' ;  i++ )
            str[i] = toupper(str[i]);
}
```

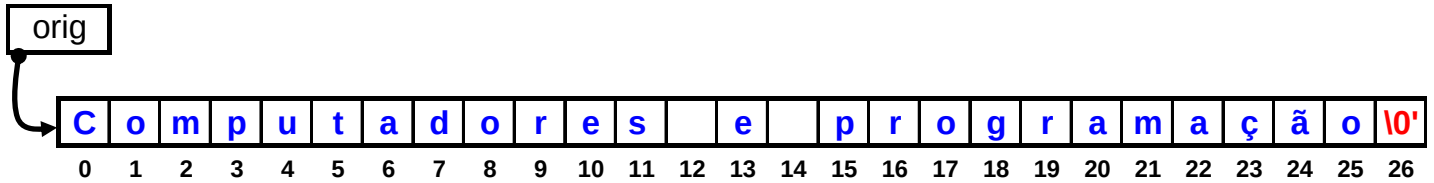


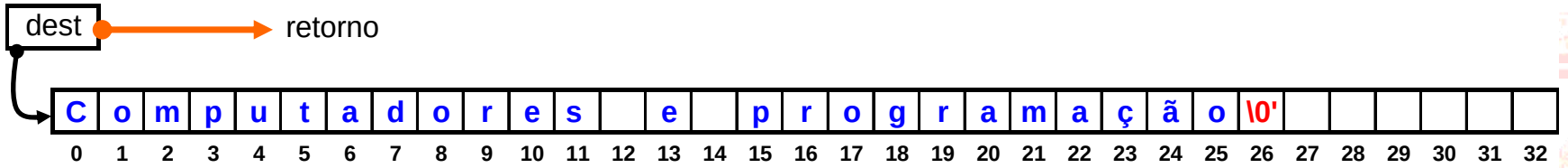
Copiar Strings



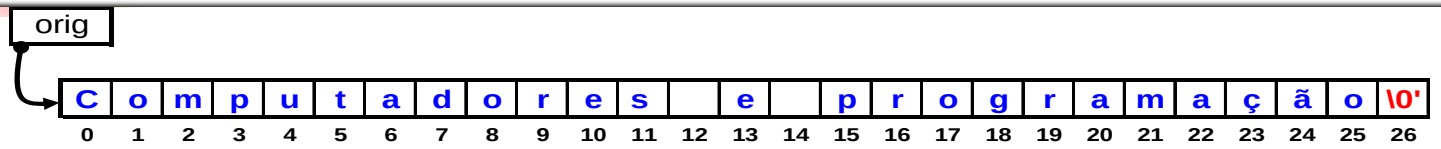


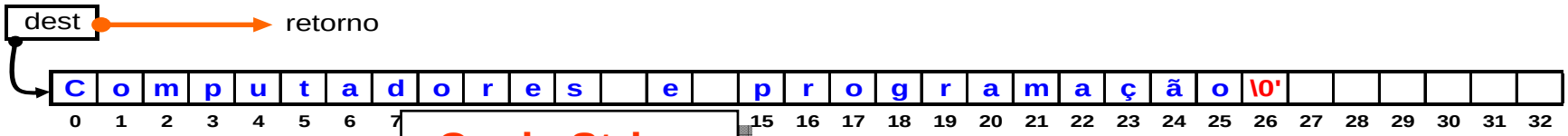
char * strcpy(char *destino, char *origem)





Copiar Strings





Copia Strings

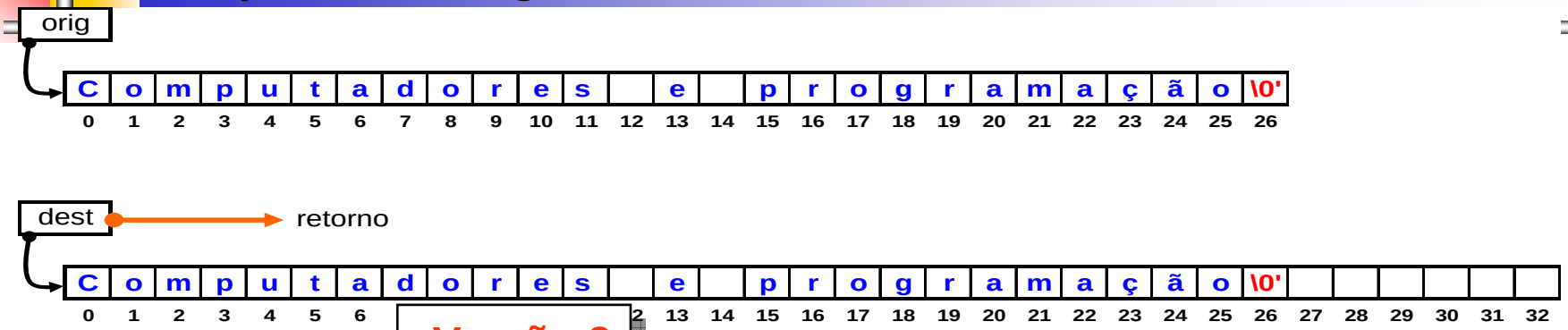
```
char * strcpy1(char *dest, char *orig) {
 for(int i=0; orig[i]!='\0' ; i++)
   dest[i] = orig[i];

 return dest;
}
```

Programa

```
char nome1[50];
char *nome2="Computadores e Programacao";
strcpy1(nome1, nome2);
printf("\nNome 1 : %s “,nome1);
```

Nome 1 : **Computadores e Programacao@**

Copiar Strings



Versão 2

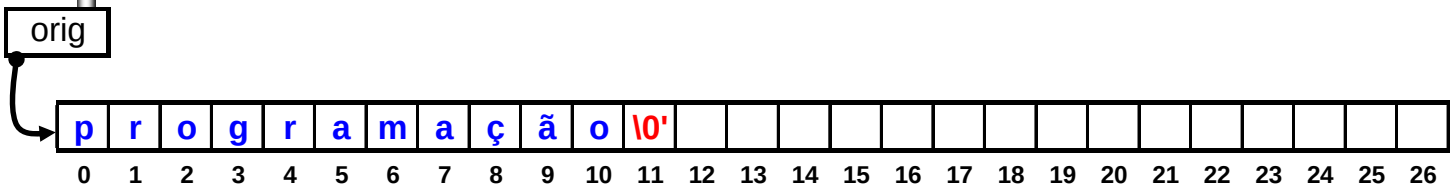
```
char *strcpy2(char *dest, char *orig){
 for(int i=0; i< orig[i]!='\0; i++)
  dest[i] = orig[i];
 dest[ strlen(orig) ]='\0';
 return dest;
}
```

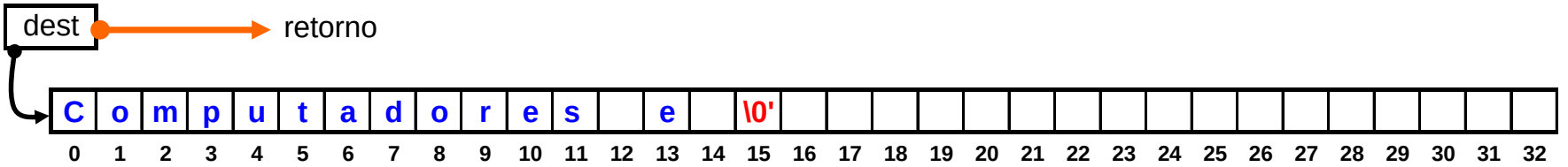
Programa

```
char nome1[50];
char *nome2="Computadores e Programacao";
strcpy2(nome1, nome2);
 printf("\nNome 1 : %s “,nome1);
```

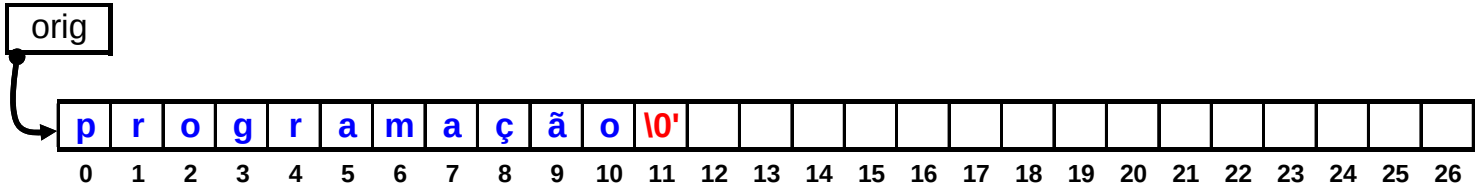
Nome 1 : Computadores e Programacao

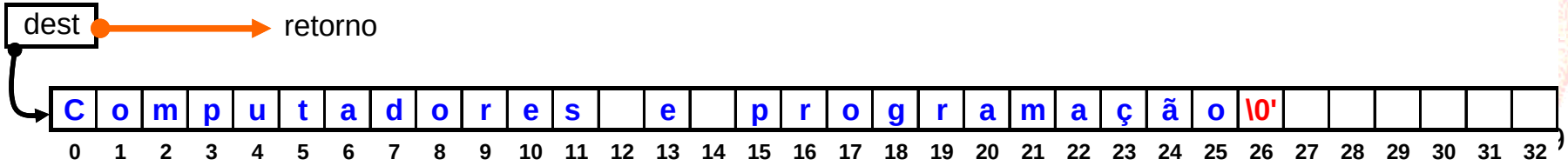
Concatenar Strings



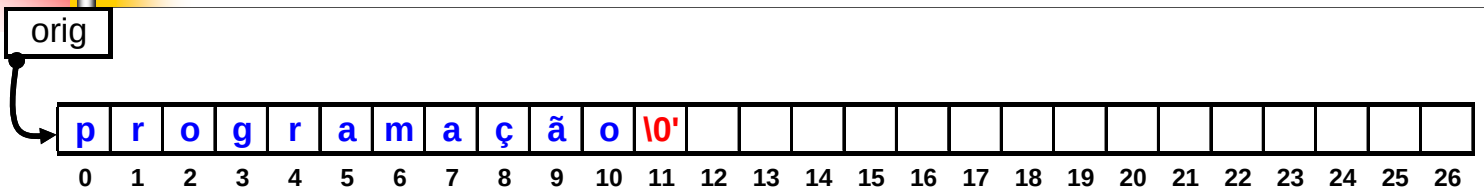


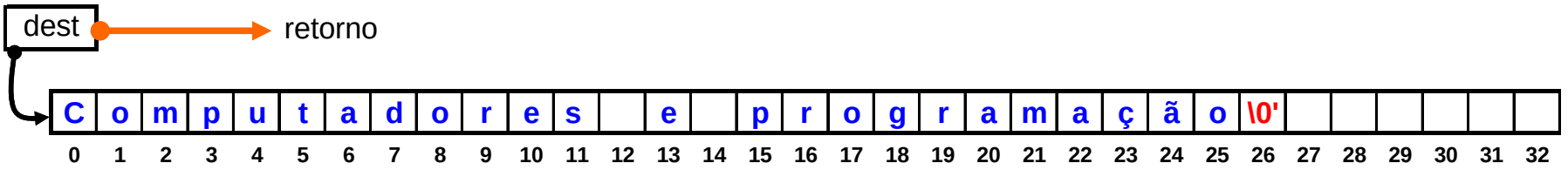
char * strcat(char *destino, char *origem)





Concatenar Strings





Concatenar strings

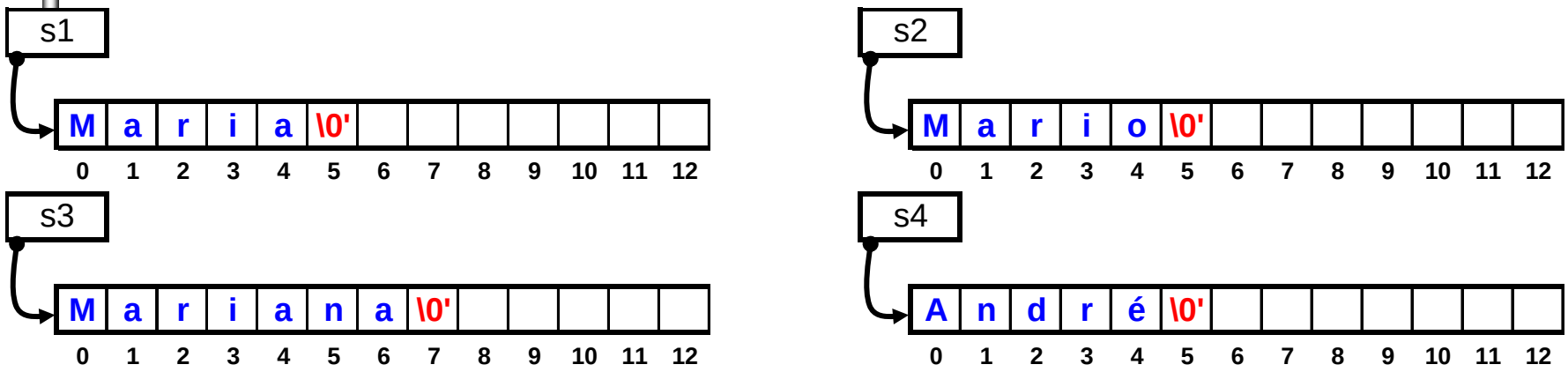
```
char * strcat(char *dest, char *orig)
{
 int i=0, tamDest=strlen(dest);
 while( orig[i] !='\0')
 {
  dest[ tamDest + i ] = orig[i];
  i++;
 }
 dest[ tamDest + i ] ='\0';
 return dest;
}
```

Comparar Strings

- Comparação lexicográfica
  - Iguais (=0)
  - Menor (< 0)
  - Maior (>0)

str1	comparação	str2	Observações
Beatriz	<	Carlos	B < C
abc	<	abxpt	c < x
carlos	=	carlos	São iguais
MARIO	>	MARIA	O > A
Maria	<	Mariana	A primeira está contida na segunda

Comparar Strings



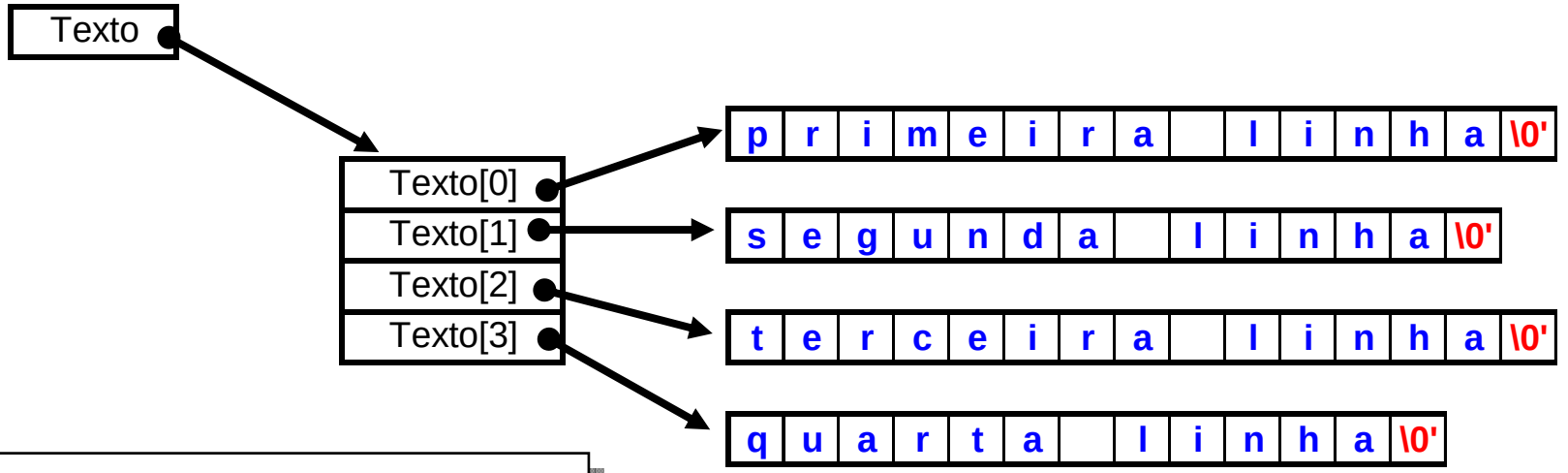
comparar strings

```
int strcmp(char *s1, char *s2 )
{
 int i=0;
 while( s1[i] == s2[i] && s1[i]!=0 )
       i++
 return s1[i] - s2[i];
}
```

Biblioteca string.h

- #include <string.h>
  - strlen
    - Devolve o tamanho da string
  - Strcpy
    - Copia uma string para outra
  - Strcat
    - Concatena strings
  - Strcmp
    - Compara strings
  - Stricmp
    - Compara strings ignorando maiusculas/minusculas
  - Strchr
    - Procura um caracter na string
  - Strstr
    - Procura uma string noutra string

Vectores de Strings



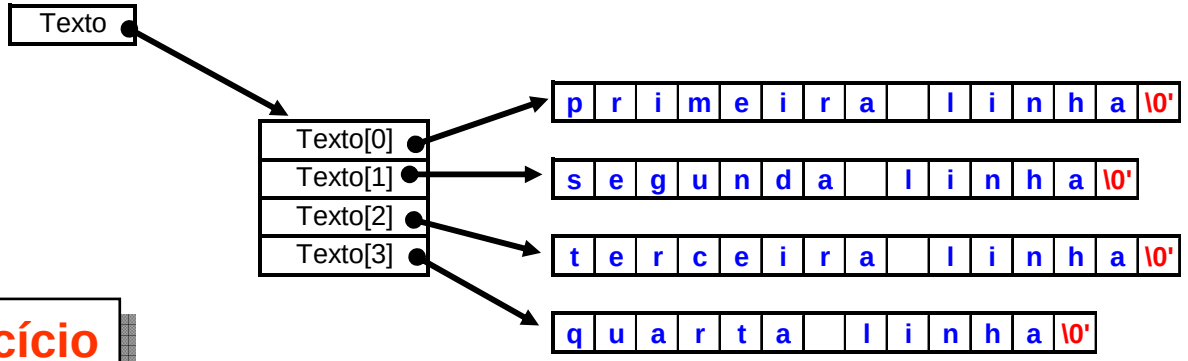
Vectores de strings

```
char  Texto[4][20];
```

Inicialização

```
char texto[2][20]={
                    "primeira linha",
                    "segunda linha“
                  };
```

Vectores de Strings e Funções



Exercício

Construa uma função que calcule o número de caracteres de um texto

Conta Caracteres

```
int contaCaracteres( char *texto[], int linhas)
{
  int soma=0;
  for(int i=0; i < linhas ; i++)
    soma += strlen(texto[i]);
  return soma;
}
```

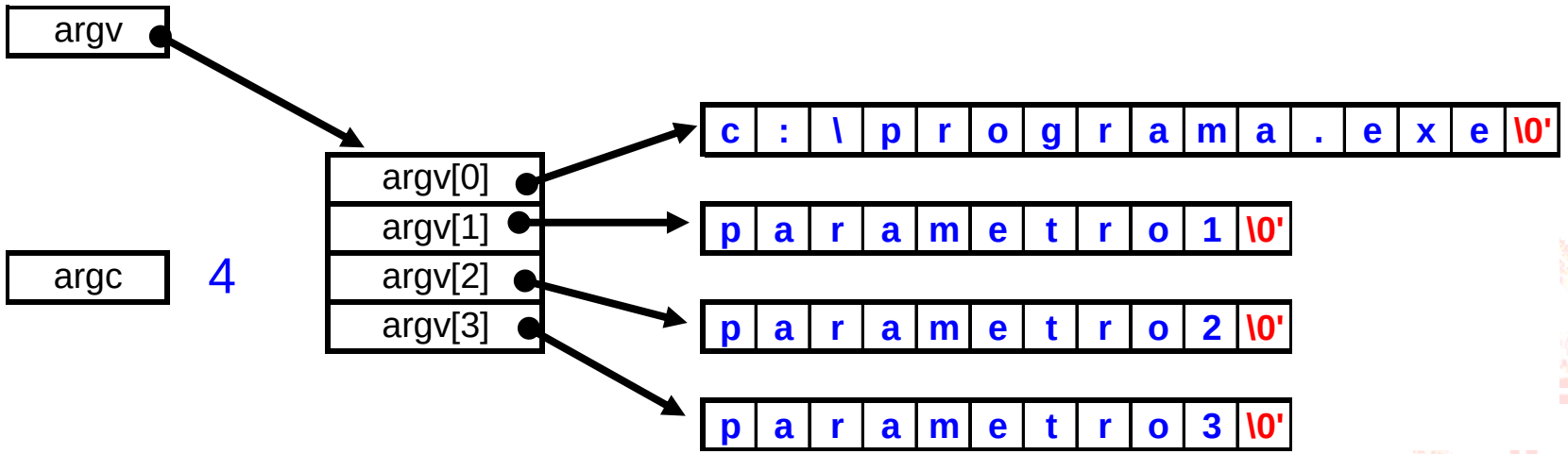
texto

```
char *texto[]
char **texto,
```

Vectores de Strings e Funções

int main(int argc, char* argv[])

C:\> programa parametro1 parametro2 parametro3



Vectores de Strings e Funções

Exercício

Construa um programa que imprima os parametros que são passos por linha de comando

Parametros da função main

```
int main(int argc, char* argv[])
{
  int i;
  printf(" Parametros  %d \n", argc);
  for(i=0; i< argc; i++)
     printf(" %s \n", argv[i]);
  return 0;
}
```

Resumo

- As strings são vectores de caracteres
  - Têm um caracter terminador
  - Os vectores de caracteres podem não ser strings
- Têm tratamento privilegiado em relação aos vectores
  - Inicializadas entre aspas
  - Lidas com gets
  - Impressas com printf ou puts
- Não têm os previlégios dos tipos básicos
  - Não podem ser comparadas directamente
  - Não podem ser copiadas directamente (atribuição)
  - Não podem ser concatenadas directamente (soma)
- Biblioteca <string.h>
  - Funções para o tratamento do strings

FIM

Dúvidas