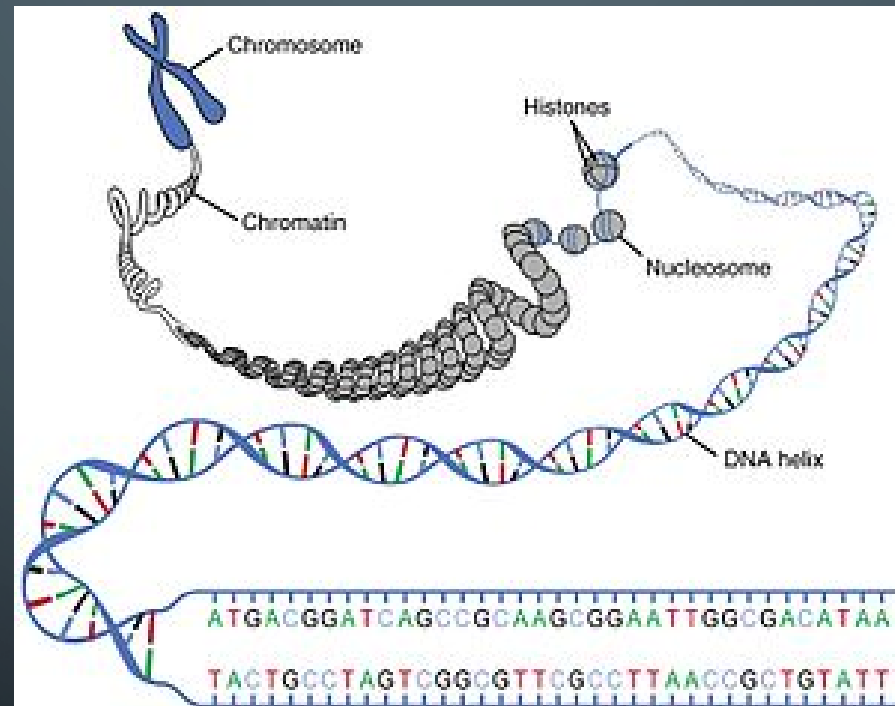


STRINGS

EM C



Prof. Manoel Pereira Junior
IFMG – Campus Formiga

C TEM STRINGS?!?! 😞

- São possíveis usando vetores de caracteres
- Tamanho do char = 1 byte = 8 bits = 256 caracteres

C TEM STRINGS?!?! ☹️

	0	1	2	3	4	5	6	7	8	9
30			sp	!	"	#	\$	%	&	'
40	()	*	+	,	-	.	/	0	1
50	2	3	4	5	6	7	8	9	:	;
60	<	=	>	?	@	A	B	C	D	E
70	F	G	H	I	J	K	L	M	N	O
80	P	Q	R	S	T	U	V	W	X	Y
90	Z	[\]	^	_	`	a	b	c
100	d	e	f	g	h	i	j	k	l	m
110	n	o	p	q	r	s	t	u	v	w
120	x	y	z	{		}	~			

Exemplo:

82	105	110	32	100	101	32	74	97	110	101	105	114	111
R	i	o		d	e		J	a	n	e	i	r	o

STRINGS

Essa definição
nos é
familiar?!?!?

- sequência de caracteres adjacentes na memória do computador
- em outras palavras... array do tipo **char**
- toda string termina com um "caractere especial", o \0

STRINGS

- segue a mesma regra da declaração de arrays convencionais:

```
char str[6];
```

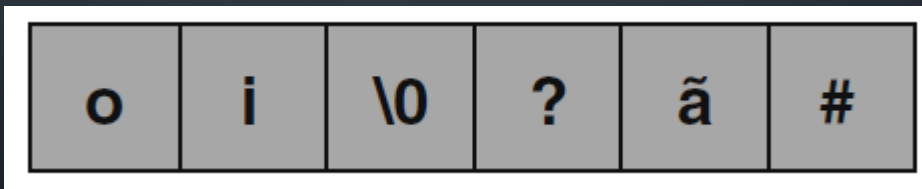
- quantos caracteres cabem em str?

STRINGS

- Qual é mesmo a função do `\0` ?

➔ Limitar a quantidade de posições utilizadas de uma string.

Imagine que você declarou uma string de tamanho 6, mas usou apenas 2...



STRINGS

- Inicializando uma string:

```
char str [10] = { 'J', 'o', 'a', 'o', '\0' };
```

ou

```
char str [10] = "Joao";
```

STRINGS

- Acessando posições da string:

```
char str [10] = "Joao";
```

```
str[0] = 'M';
```

J	o	a	o	\0
M	o	a	o	\0

LEITURA

- Ler uma string:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  int main ()
6  {
7
8      char str1[20];
9      char str2[20];
10
11     scanf("%s",str1);
```

Não se usa o
operador & pois
o próprio nome
da "string" já
é um ponteiro!

LEITURA

- Scanf lê até o primeiro espaço...
- Solução:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  int main(){
5
6      int i;
7      char str1[20];
8      char str2[20];
9
10     scanf("%[^\n]s",str1);
```

LEITURA

- Aceitar apenas letras e espaços...

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  int main ()
6  {
7
8      char str1[20];
9      char str2[20];
10
11     scanf("%[a-z A-Z]s",str1);
12
```

LEITURA

- Modificadores determinam parada de leitura usando scanf

➔ `%[...]` lista entre os colchetes todos os caracteres aceitos na leitura

➔ `%[^...]` lista entre os colchetes todos os caracteres não aceitos na leitura

- Exemplos:

`%[aeiou] | %[^aeiou]`

`%120s`


LEITURA

- Ao usar sucessivos comandos scanf, pode-se ter problemas com buffer
- Use o comando abaixo para limpar o buffer


```
setbuf(stdin, NULL);
```

Onde stdin é o dispositivo de entrada padrão (teclado)

TAMANHO



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 int main(){
5
6     char str1[20];
7
8     scanf("%[^\n]s",str1);
9
10    int i = strlen(str1);
11
12    printf("Tamanho de str1 = %d\n", i);
13
14    return 0;
15 }
```



ATRIBUIÇÃO

- Strings são arrays, portanto não se pode fazer atribuição direta!

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  int main(){
4
5      char str1[20] = "Hello World";
6      char str2[20];
7
8      str1 = str2; //ERRADO!
9
10     return 0;
11
12 }
```

ATRIBUIÇÃO

- Como fazer então?

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  int main(){
4
5      int i;
6      char str1[20] = "Hello World";
7      char str2[20];
8
9      for (i = 0; str1[i] != '\0'; i++)
10         str2[i] = str1[i];
11
12     str2[i] = '\0';
13
14     return 0;
15 }
```


Bem prático,
não?

ATRIBUIÇÃO


- Como fazer então de forma prática?

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  int main(){
5
6      char str1[100], str2[100];
7
8      printf("Entre com uma string: ");
9      scanf("%[^\n]s",str1);
10
11     strcpy(str2, str1);
12
13     printf("String 1: %s\n",str1);
14     printf("String 2: %s\n",str2);
15
16     return 0;
17 }
```



CONCATENAÇÃO



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 int main(){
5
6     char str1[15] = "bom ";
7     char str2[15] = "dia";
8
9     strcat(str1,str2);
10
11     printf("%s",str1);
12
13     return 0;
14 }
```



COMPARAÇÃO



```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  int main(){
5      char str1[100], str2[100];
6      printf("Entre com uma string: ");
7      scanf("%[^\n]s",str1);
8      setbuf(stdin, NULL);
9
10     printf("Entre com outra string: ");
11     scanf("%[^\n]s",str2);
12
13     if(strcmp(str1,str2) == 0)
14         printf("Strings iguais\n");
15     else
16         printf("Strings diferentes\n");
17
18     return 0;
19
20 }
```

Case
sensitive

LOCALIZANDO UM CHARACTER NA STRING

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 int main(){
5
6     char str1[100] = "Exemplo de uso da função strchr";
7
8     char *subcadeia;
9
10    subcadeia = strchr(str1, 'p');
11
12    printf("%s", subcadeia);
13
14    return 0;
15 }
```

Running `"/home/ubuntu/workspace/testes_c/strnigs.c"`
Exemplo de uso da função strchr

LOCALIZANDO SUBSTRING NA STRING

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  int main(){
5
6      char str1[100] = "Exemplo de uso da função strstr";
7
8      char *subcadeia;
9
10     subcadeia = strstr(str1, " de");
11
12     printf("%s",subcadeia);
13
14     return 0;
15 }
```

Running "/home/ubuntu/workspace/testes_c/strnigs.c"
de uso da função strstr

QUEBRANDO UMA STRING POR TOKEN

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include "manoel.h"
5
6 int main()
7 {
8     char str[] = "exemplo de string que será separada por espaços";
9
10    //cada chamada a strtok retorna um token apenas
11    char* token = strtok(str, " ");
12
13    //se precisar achar todos os tokens, use chamadas sucessivas em um laço
14    while (token != NULL) {
15        printf("%s\n", token);
16        token = strtok(NULL, " ");
17    }
18    return 0;
19 }
```

exemplo
de
string
que
será
separada
por
espaços

NULL aqui indica
que quero procurar
na mesma string

MAIS FUNÇÕES

https://www.gnu.org/software/libc/manual/html_node/String-and-Array-Utilities.html#String-and-Array-Utilities

• Referências:

