

FCT/Unesp – Presidente Prudente
Departamento de Matemática e Computação

ATP I

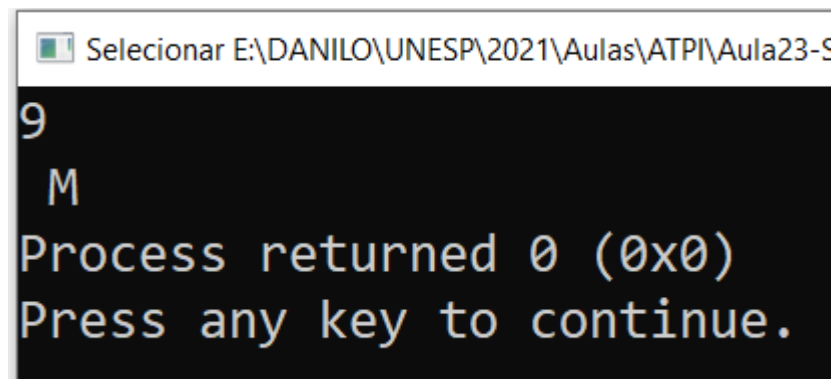
cadeia de caracteres ou string

Prof. Danilo Medeiros Eler
danilo.eler@unesp.br

Tipo de Dado Caractere

- O tipo caractere permite armazenar qualquer informação que pode ser representada em 1 byte
- Exemplo:

```
char n = '9';  
char s = 'M';  
char pular = '\n';  
printf("%c %c %c",n,pular,s);
```



```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-S  
9  
 M  
Process returned 0 (0x0)  
Press any key to continue.
```

Tipo de Dado Caractere

- A leitura de um caractere é realizada com a especificação de formato %c

- Exemplo:

```
char sexo, simbol;  
printf("Sexo: ");  
scanf("%c",&sexo);  
printf("Simbolo: ");  
scanf("%c",&simbol);  
printf("\nSimbolo: %c --- Sexo: %c\n",simbol,sexo);
```

```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\caracter  
Sexo: f  
Simbolo:  
Simbolo:  
--- Sexo: f  
  
Process returned 0 (0x0)  
Press any key to continue.
```

Tipo de Dado Caractere

- A leitura de um caractere é realizada com a especificação de formato %c

- Exemplo:

```
char sexo, simbol;
```

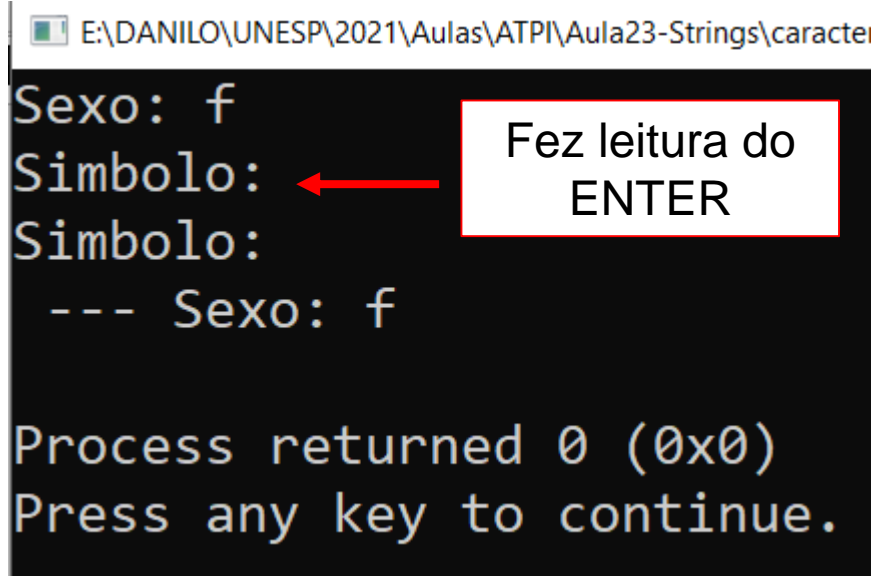
```
printf("Sexo: ");
```

```
scanf("%c",&sexo);
```

```
printf("Simbolo: ");
```

```
scanf("%c",&simbol);
```

```
printf("\nSimbolo: %c --- Sexo: %c\n",simbol,sexo);
```



```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\caracter
Sexo: f
Simbolo: ← Fez leitura do ENTER
Simbolo:
--- Sexo: f

Process returned 0 (0x0)
Press any key to continue.
```

Tipo de Dado Caractere

- A leitura de um caractere é realizada com a especificação de formato %c

- Exemplo:

```
char sexo, simbol;  
printf("Sexo: ");  
scanf("%c",&sexo);
```

```
fflush(stdin);  
//ou setbuf(stdin, NULL);
```

```
printf("Simbolo: ");  
scanf("%c",&simbol);  
printf("\nSimbolo: %c --- Sexo: %c\n",simbol,sexo);
```

```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\c  
Sexo: f  
Simbolo: @  
  
Simbolo: @ --- Sexo: f  
  
Process returned 0 (0x0)  
Press any key to continue.
```

Tipo de Dado Caractere

- O caractere tem a limitação de armazenamento de 1 byte
- Não conseguimos, por exemplo, armazenar um nome

```
char nome;  
printf("Digite seu nome: ");  
scanf("%c",&nome);  
printf("Seu nome: %c",nome);
```

```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\cara  
Digite seu nome: danilo  
Seu nome: d  
Process returned 0 (0x0)  
Press any key to continue.
```

Cadeia de Caracteres ou String

- A cadeia de caracteres é uma maneira de armazenar um conjunto de caracteres
- Geralmente, em algumas linguagens, a cadeia de caracteres é representada por um tipo de dado chamado string

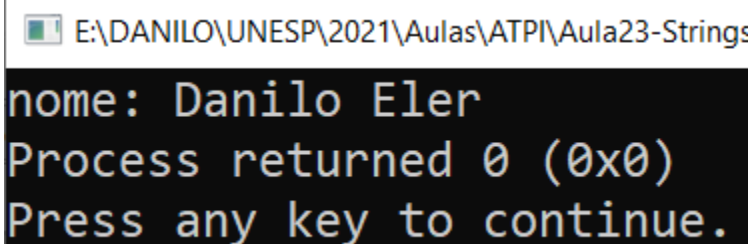
Cadeia de Caracteres ou String

- Na linguagem C, podemos especificar uma cadeia de caracteres desse modo
`char nome[50];`
- Nesse exemplo, indicamos que a cadeia de caracteres terá uma área de memória de 50 bytes
 - Armazenará 50 caracteres

Cadeia de Caracteres ou String

- Podemos inicializar essa variável no momento em que é definida
- A exibição exige a especificação de formato %s, fazendo referência à string

```
char nome[50]="Danilo Eler";  
printf("nome: %s",nome);
```

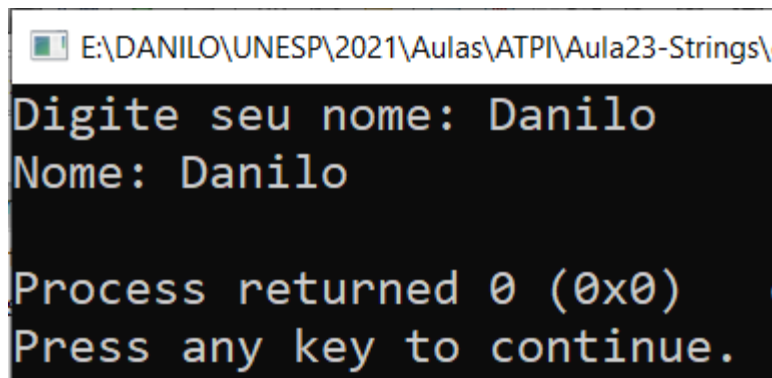


```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings  
nome: Danilo Eler  
Process returned 0 (0x0)  
Press any key to continue.
```

Cadeia de Caracteres ou String

- Para leitura com scanf também podemos usar %s

```
char nome[50];  
printf("Digite seu nome: ");  
scanf("%s",&nome);  
printf("Nome: %s\n",nome);
```



```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\  
Digite seu nome: Danilo  
Nome: Danilo  
  
Process returned 0 (0x0)  
Press any key to continue.
```

Cadeia de Caracteres ou String

- Uma limitação do scanf é que a leitura para quando espaço é encontrado

```
char nome[50];  
printf("Digite seu nome: ");  
scanf("%s",&nome);  
printf("Nome: %s\n",nome);
```

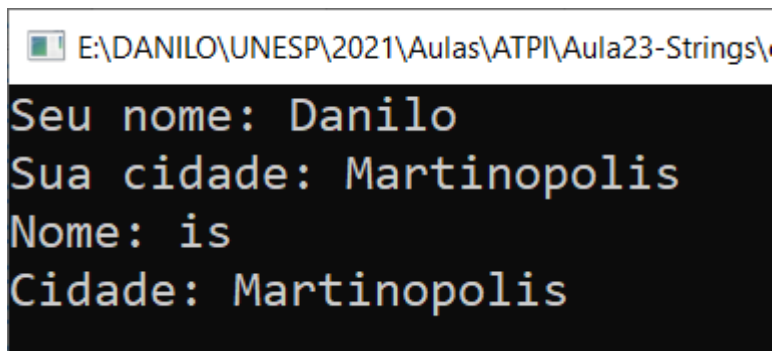
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings

```
Digite seu nome: Danilo Eler  
Nome: Danilo  
  
Process returned 0 (0x0)   exe  
Press any key to continue.
```

Cadeia de Caracteres ou String

- Outra limitação da leitura de strings é quando são informados mais caracteres do que o espaço reservado em memória

```
char nome[10], cidade[10];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```



```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\  
Seu nome: Danilo  
Sua cidade: Martinopolis  
Nome: is  
Cidade: Martinopolis
```

Cadeia de Caracteres ou String

- Apesar da quantidade de caracteres indicada para armazenamento em memória, o último caractere é reservado para um marcador de fim de string
 - Esse marcador é o ‘\0’
- Dessa maneira, a função de exibição sabe quando deverá parar de percorrer a memória
 - Essa também é uma limitação que pode gerar diversos problemas

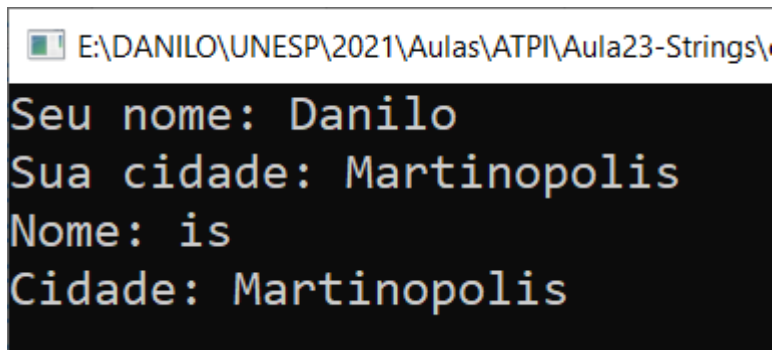
Cadeia de Caracteres ou String

- Na leitura, esse marcador ‘\0’ é inserido pela função de leitura quando o espaço ou quebra de linha (ENTER) é encontrado
- A limitação é que os caracteres lidos são inseridos na memória, sem observar o tamanho reservado para a variável

Cadeia de Caracteres ou String

- Por isso, nesse exemplo, a cadeia de caracteres pode avançar até a área de memória de outras variáveis
 - O mesmo ocorre na exibição

```
char nome[10], cidade[10];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```



```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\  
Seu nome: Danilo  
Sua cidade: Martinopolis  
Nome: is  
Cidade: Martinopolis
```

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

END	DADO
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
.....	

Cadeia de Caracteres ou String

- Exemplo

→ `char nome[5], cidade[5];`
`printf("Seu nome: ");`
`scanf("%s",&nome);`
`fflush(stdin);`
`printf("Sua cidade: ");`
`scanf("%s",&cidade);`
`printf("Nome: %s\n",nome);`
`printf("Cidade: %s\n",cidade);`

	cidade →	END	DADO
		0	
		1	
		2	
		3	
		4	
	nome →	5	
		6	
		7	
		8	
		9	
		10	
		

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
→ printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

Seu nome:

cidade →

nome →

END	DADO
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
.....	

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
→ scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

Seu nome: Lara

cidade	→	END	DADO
		0	
		1	
		2	
		3	
		4	
nome	→	5	
		6	
		7	
		8	
		9	
		10	
		

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
→ scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

Seu nome: Lara

	END	DADO
cidade →	0	
	1	
	2	
	3	
	4	
nome →	5	L
	6	a
	7	r
	8	a
	9	\0
	10	
	

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
→ printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

Seu nome: Lara
Sua cidade:

	END	DADO
cidade →	0	
	1	
	2	
	3	
	4	
nome →	5	L
	6	a
	7	r
	8	a
	9	\0
	10	
	

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
→ scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

Seu nome: Lara
Sua cidade: Indiana

	END	DADO
cidade →	0	
	1	
	2	
	3	
	4	
nome →	5	L
	6	a
	7	r
	8	a
	9	\0
	10	
	

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
→ scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

Seu nome: Lara
Sua cidade: Indiana

cidade	END	DADO
	0	I
	1	n
	2	d
	3	i
	4	a
nome	5	n
	6	a
	7	\0
	8	a
	9	\0
	10	
	

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
→ printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

Seu nome: Lara
Sua cidade: Indiana
Nome: na

	END	DADO
cidade →	0	l
	1	n
	2	d
	3	i
	4	a
nome →	5	n
	6	a
	7	\0
	8	a
	9	\0
	10	
	

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
→ printf("Cidade: %s\n",cidade);
```

```
Seu nome: Lara  
Sua cidade: Indiana  
Nome: na  
Cidade: Indiana
```

Não para no limite
da alocação da
variável

cidade

nome

END	DADO
0	I
1	n
2	d
3	i
4	a
5	n
6	a
7	\0
8	a
9	\0
10	
.....	

Cadeia de Caracteres ou String

- Exemplo

```
char nome[5], cidade[5];  
printf("Seu nome: ");  
scanf("%s",&nome);  
fflush(stdin);  
printf("Sua cidade: ");  
scanf("%s",&cidade);  
printf("Nome: %s\n",nome);  
→ printf("Cidade: %s\n",cidade);
```

```
Seu nome: Lara  
Sua cidade: Indiana  
Nome: na  
Cidade: Indiana
```

Para somente
quando encontra
o marcador

cidade

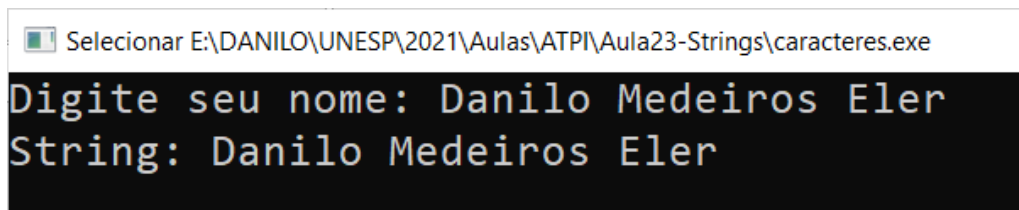
nome

END	DADO
0	I
1	n
2	d
3	i
4	a
5	n
6	a
7	\0
8	a
9	\0
10	
.....	

Cadeia de Caracteres ou String

- Podemos utilizar outras funções para fazer a leitura de strings
- Por exemplo, a função gets faz a leitura até encontrar o fim de linha

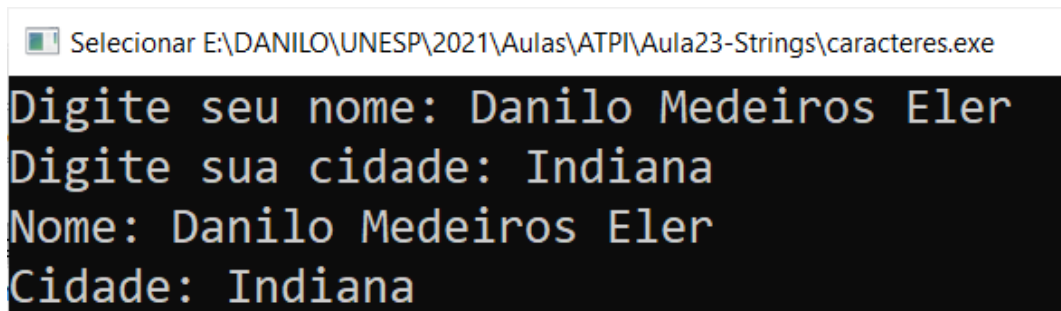
```
char nome[50];  
printf("Digite seu nome: ");  
gets(nome);  
printf("String: %s\n",nome);
```



Cadeia de Caracteres ou String

- A função gets também tem a mesma limitação da leitura realizada com scanf, ou seja, pode ultrapassar os limites da área alocada para a variável

```
char nome[50], cidade[10];  
printf("Digite seu nome: ");  
gets(nome);  
printf("Digite sua cidade: ");  
gets(cidade);  
printf("Nome: %s\n", nome);  
printf("Cidade: %s\n", cidade);
```

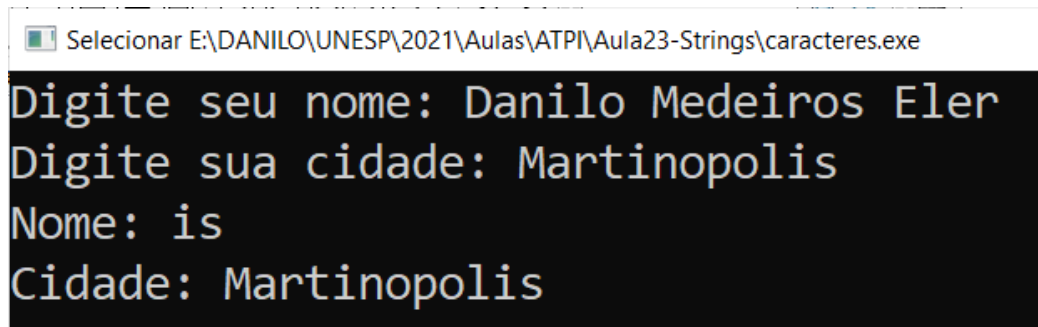


```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\caracteres.exe  
Digite seu nome: Danilo Medeiros Eler  
Digite sua cidade: Indiana  
Nome: Danilo Medeiros Eler  
Cidade: Indiana
```

Cadeia de Caracteres ou String

- A função gets também tem a mesma limitação da leitura realizada com scanf, ou seja, pode ultrapassar os limites da área alocada para a variável

```
char nome[50], cidade[10];  
printf("Digite seu nome: ");  
gets(nome);  
printf("Digite sua cidade: ");  
gets(cidade);  
printf("Nome: %s\n", nome);  
printf("Cidade: %s\n", cidade);
```



```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\caracteres.exe  
Digite seu nome: Danilo Medeiros Eler  
Digite sua cidade: Martinopolis  
Nome: is  
Cidade: Martinopolis
```

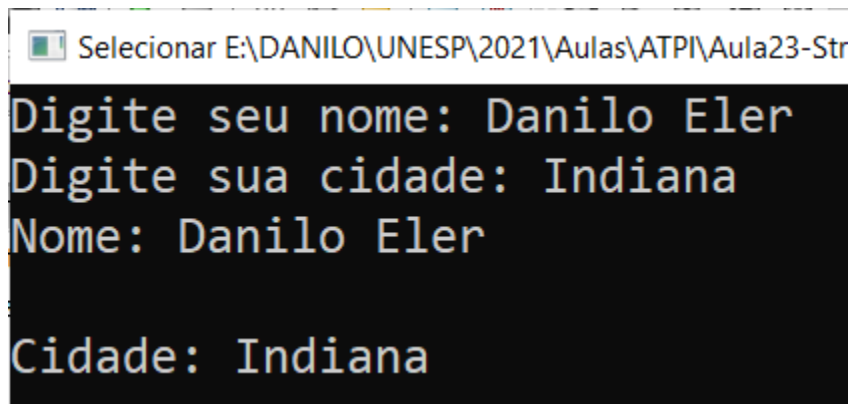
Cadeia de Caracteres ou String

- Uma alternativa é limitar a quantidade de caracteres que serão lidos na entrada de dados
- Para tanto, uma opção é a função *fgets*, que, de modo simplificado, tem a seguinte assinatura
 fgets(variavelString, quantidade, origem);
 - variavelString: nome da variável para armazenar os caracteres em memória;
 - quantidade: número de caracteres que serão lidos da entrada;
 - origem: indica a entrada de dados para a função efetuar a leitura.
- O marcador '\0' é inserido após o termino da leitura
 - Após o último caractere lido por quebra de linha ou pela quantidade
 - O '\n' também é inserido na string, caso haja espaço

Cadeia de Caracteres ou String

- Exemplo com fgets
 - Sem exceder tamanho ou quantidade

```
char nome[50], cidade[10];  
printf("Digite seu nome: ");  
fgets(nome,50,stdin);  
printf("Digite sua cidade: ");  
fgets(cidade,10,stdin);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

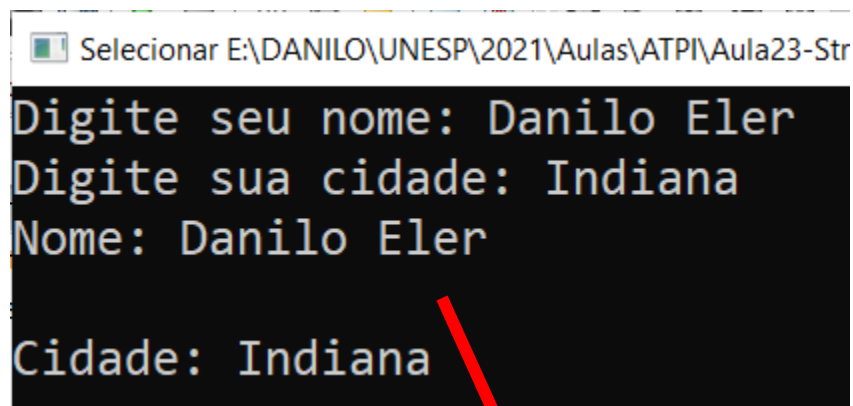


```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Str  
Digite seu nome: Danilo Eler  
Digite sua cidade: Indiana  
Nome: Danilo Eler  
Cidade: Indiana
```

Cadeia de Caracteres ou String

- Exemplo com fgets
 - Sem exceder tamanho ou quantidade

```
char nome[50], cidade[10];  
printf("Digite seu nome: ");  
fgets(nome,50,stdin);  
printf("Digite sua cidade: ");  
fgets(cidade,10,stdin);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```



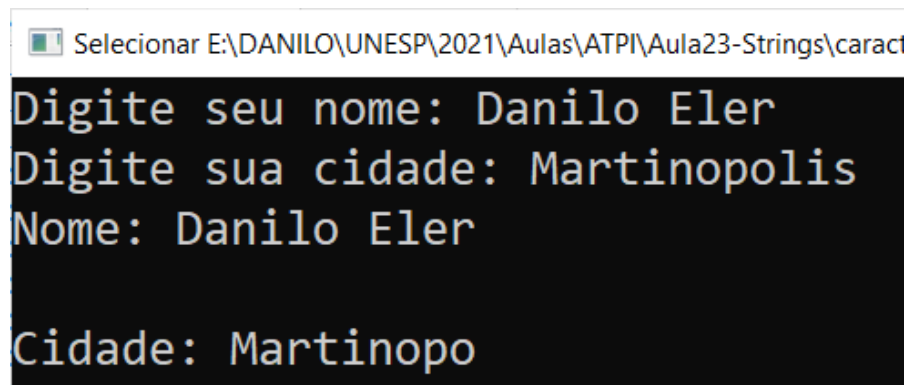
```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Str  
Digite seu nome: Danilo Eler  
Digite sua cidade: Indiana  
Nome: Danilo Eler  
Cidade: Indiana
```

Note que a quebra de linha ocorre por causa do armazenamento do '\n'

Cadeia de Caracteres ou String

- Exemplo com fgets
 - Excedendo tamanho e quantidade

```
char nome[50], cidade[10];  
printf("Digite seu nome: ");  
fgets(nome,50,stdin);  
printf("Digite sua cidade: ");  
fgets(cidade,10,stdin);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```

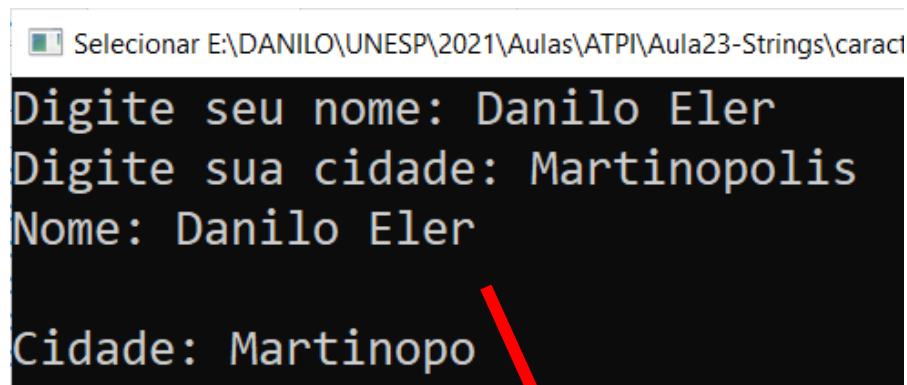


```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\caract  
Digite seu nome: Danilo Eler  
Digite sua cidade: Martinopolis  
Nome: Danilo Eler  
Cidade: Martinopo
```

Cadeia de Caracteres ou String

- Exemplo com fgets
 - Excedendo tamanho e quantidade

```
char nome[50], cidade[10];  
printf("Digite seu nome: ");  
fgets(nome,50,stdin);  
printf("Digite sua cidade: ");  
fgets(cidade,10,stdin);  
printf("Nome: %s\n",nome);  
printf("Cidade: %s\n",cidade);
```



```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\caract  
Digite seu nome: Danilo Eler  
Digite sua cidade: Martinopolis  
Nome: Danilo Eler  
Cidade: Martinopo
```

Note que a quebra de linha ocorre por causa do armazenamento do '\n'

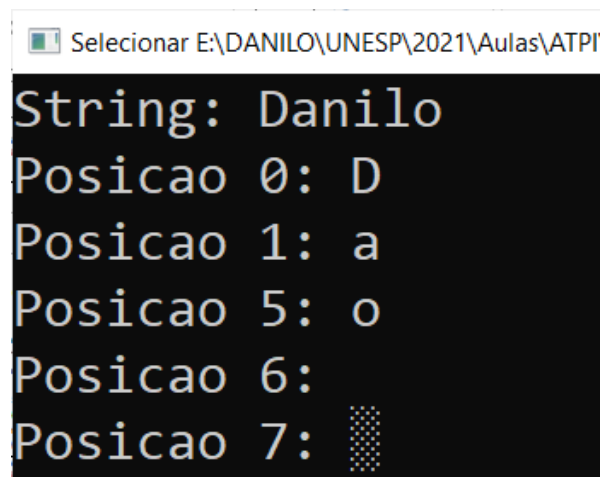
Cadeia de Caracteres ou String

- A cadeia de caracteres é um *array* ou vetor com elementos do tipo `char`
- Por isso, podemos acessar cada elemento que a cadeia armazena. Para tanto, basta indicar o índice do elemento que desejamos acessar

- Exemplo:

```
char nome[7] = "Danilo";  
printf("String: %s\n",nome);  
printf("Posicao 0: %c\n",nome[0]);  
printf("Posicao 1: %c\n",nome[1]);  
printf("Posicao 5: %c\n",nome[5]);  
printf("Posicao 6: %c\n",nome[6]);  
printf("Posicao 7: %c\n",nome[7]);
```

0	1	2	3	4	5	6
D	a	n	i	l	o	\0



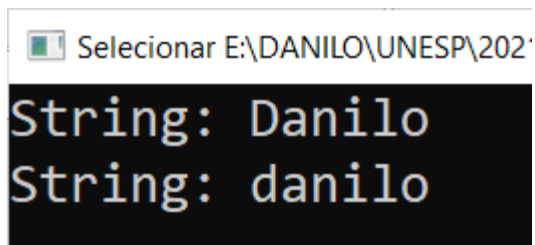
```
Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI  
String: Danilo  
Posicao 0: D  
Posicao 1: a  
Posicao 5: o  
Posicao 6:  
Posicao 7:
```

Cadeia de Caracteres ou String

- A cadeia de caracteres é um *array* ou vetor com elementos do tipo char
- Por isso, podemos acessar cada elemento que a cadeia armazena. Para tanto, basta indicar o índice do elemento que desejamos acessar

- Exemplo:

```
char nome[7] = "Danilo";  
printf("String: %s\n",nome);  
nome[0] = 'd';  
printf("String: %s\n",nome);
```



```
Selecionar E:\DANILO\UNESP\202  
String: Danilo  
String: danilo
```

0	1	2	3	4	5	6
D	a	n	i	l	o	\0



alteração do elemento da posição 0

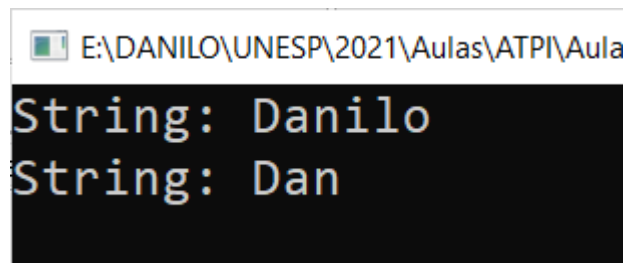
0	1	2	3	4	5	6
d	a	n	i	l	o	\0

Cadeia de Caracteres ou String

- A cadeia de caracteres é um *array* ou vetor com elementos do tipo char
- Por isso, podemos acessar cada elemento que a cadeia armazena. Para tanto, basta indicar o índice do elemento que desejamos acessar

- Exemplo:

```
char nome[7] = "Danilo";  
printf("String: %s\n",nome);  
nome[3] = '\0';  
printf("String: %s\n",nome);
```



```
E:\DANILO\UNESP\2021\Aulas\ATPI\Aula  
String: Danilo  
String: Dan
```

0	1	2	3	4	5	6
D	a	n	i	l	o	\0



alteração do elemento da posição 3

0	1	2	3	4	5	6
D	a	n	\0	l	o	\0



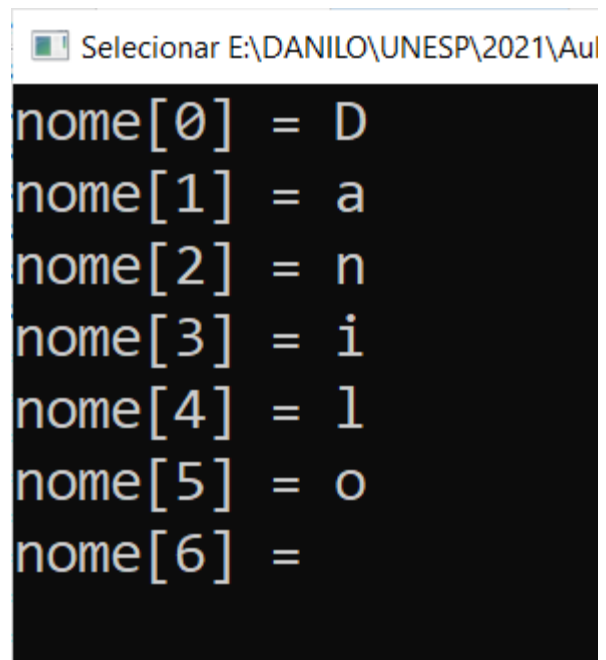
marcador de final de string

Cadeia de Caracteres ou String

- Podemos utilizar estruturas de repetição para percorrer a string
- Exemplo:

0	1	2	3	4	5	6
D	a	n	i	l	o	\0

```
char nome[7] = "Danilo";  
for(int i=0; i<=6; i++){  
    printf("nome[%d] = %c\n",i, nome[i]);  
}
```



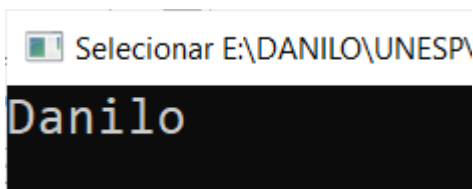
```
nome[0] = D  
nome[1] = a  
nome[2] = n  
nome[3] = i  
nome[4] = l  
nome[5] = o  
nome[6] =
```

Cadeia de Caracteres ou String

- Podemos utilizar estruturas de repetição para percorrer a string
- Exemplo:

0	1	2	3	4	5	6
D	a	n	i	l	o	\0

```
char nome[7] = "Danilo";  
int cont=0;  
while(nome[cont]!='\0'){  
    printf("%c",nome[cont]);  
    cont++;  
}
```



Cadeia de Caracteres ou String

- Contar quantas letras 'a' estão na palavra
- Exemplo:

```
char palavra[20] = "abacate";  
int cont=0, i=0;  
while (palavra[i]!='\0'){  
    if (palavra[i] == 'a'){  
        cont++;  
    }  
    i++;  
}  
printf("Quantidade de letras a: %d\n",cont);
```

0	1	2	3	4	5	6	7	...	19
a	b	a	c	a	t	e	\0	...	

Selecionar E:\DANILO\UNESP\2021\Aulas\ATPI\Aula23-Strings\carac

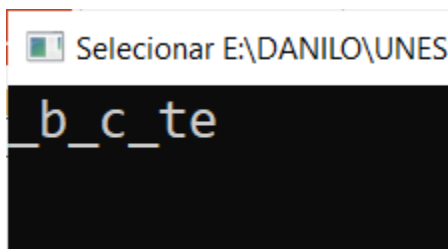
```
Quantidade de letras a: 3
```


Cadeia de Caracteres ou String

- Exibir somente as letras diferentes de 'a'
- Exemplo:

```
char palavra[20] = "abacate";  
int i=0;  
while (palavra[i]!='\0'){  
    if (palavra[i] != 'a'){  
        printf("%c",palavra[i]);  
    }else{  
        printf("_");  
    }  
    i++;  
}
```

0	1	2	3	4	5	6	7	...	19
a	b	a	c	a	t	e	\0	...	



Exercícios

- 1) Ler o nome, idade e endereço de uma pessoa. Em seguida, exibir na tela.
- 2) Verificar se uma string contém um determinado caractere.
- 3) Ler um número binário e substituir o caractere '0' por um '*'.
- 4) Exibir os 3 primeiros caracteres de uma string.
- 5) Exibir os 3 últimos caracteres de uma string.

Bibliografia Básica

1. ASCENCIO, A. F. G.; CAMPOS, E. A. V. C. **Fundamentos da programação de computadores**: algoritmos, pascal e C/C++. Pearson Prentice Hall, 2003. 355p.
2. KERNINGHAN, B. W.; Ritchie, D. M. **C: a Linguagem de Programação padrão ANSI**. Rio de Janeiro: Editora Campus, 1990. 289p.
3. KERNINGHAN, B. W.; Pike, R. **A Prática de Programação**. Rio de Janeiro: Editora Campus, 2000, 280p.
4. LOPES, A.; GARCIA, G. **Introdução à Programação**: 500 exercícios resolvidos. Rio de Janeiro: Editora campus, 2002. 469p.
5. MANZANO, J. A. N. G.; OLIVEIRA, J. F. **Algoritmos**: lógica para desenvolvimento de programação de computadores. 10ª ed. Editora Érica Ltda. 2000, 236p.
6. MIZRAHI, V. V. **Treinamento em Linguagem C – Curso Completo – Módulo 1**. São Paulo: MAKRON Books do Brasil Editora Ltda, 1990. 241p.
7. MEDINA, M.; FERTIG, C. **Algoritmos e programação**: teoria e prática. São Paulo: Novatec Editora. 384p. 2005.
7. PUGA, S.; RISSETTI, G. **Lógica de programação e estruturas de dados**: com aplicações em JAVA. São Paulo: Pearson Education do Brasil, 254p. 2004.
8. SCHILDT H. **C Completo e Total**. 3ª ed. São Paulo: MAKRON Books do Brasil editora Ltda. 1997. 827p.
9. XAVIER, G. F. C. **Lógica de Programação**. São Paulo: Editora SENAC. 1998. 378p.

Bibliografia Complementar

1. BROOKSHEAR, J. G. **Ciência da computação**: uma visão abrangente. 5ª ed., Bookman Editora, 2000. 499p.
2. CORMEN, T.H., Leiserson, C.E., Rivest R.L., Stein, C. **Algoritmos**: teoria e Prática. Rio de janeiro: Editora Campus, 2002. 916p.
3. PLAUGER, P. L. **A Biblioteca Standard C**. Rio de Janeiro: Editora Campus, 1994. 614p.
4. PRATA, S. **C primer plus**, 4ª ed. SAMS Publishing, 2002. 931p.
5. OLIVEIRA, U. **Programando em C**, vol. I – fundamentos. Editora Ciência Moderna, 2008, 743p.