

**INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PERNAMBUCO**

CAMPUS AFOGADOS DA INGAZEIRA

Introdução à Programação

Strings

Prof. Dr. Diego Rodrigues de Almeida
diego.rodrigues@afogados.ifpe.edu.br
<https://sites.google.com/site/ifpediego>

String

- ▶ Como já sabemos uma declaração do tipo
 - ▶ `char Cad[10];`
- ▶ Define um conjunto de dez posições de memória para armazenar 10 valores do tipo `char`
- ▶ Um vetor cujas componentes são do tipo *char* constitui uma *cadeia de caracteres*

String

- ▶ Uma String é uma cadeia de caracteres terminado por um caractere nulo, que geralmente é especificado como ‘\0’
- ▶ Ou seja, Strings são vetores de **chars** terminados por ‘\0’.
Nada mais e nada menos.
- ▶ “Afogados da Ingazeira”

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
A	f	o	g	a	d	o	s		d	a		l	n	g	a	z	e	i	r	a	\0

- ▶ “Próxima Linha\n”

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
P	r	ó	x	i	m	a		L	i	n	h	a	\n	\0

ATENÇÃO!!!

- ▶ Devemos ficar atentos para o fato de que as strings têm o seu último elemento como um '\0'.
- ▶ Devemos lembrar que o tamanho da string deve incluir o '\0' no final
- ▶ Por exemplo, para armazenar um cadeia de 40 caracteres deve-se reservar um vetor de 41 de caracteres

ATENÇÃO!!!

► “Afogados da Ingazeira”

- Letras = 19
- Espaços em Branco = 2
- \0 = 1
- Total = 22

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
A	f	o	g	a	d	o	s		d	a		l	n	g	a	z	e	i	r	a	\0

► Próxima linha\n

- Letras = 13 (letras + \n)
- Espaços em Branco = 1
- \0 = 1
- Total = 15

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
P	r	ó	x	i	m	a		L	i	n	h	a	\n	\0

Delimitação

- ▶ Enquanto um caractere é delimitado por aspas simples, em C as strings são delimitadas por aspas duplas
- ▶ Por exemplo:
 - ▶ “Programando em C”
- ▶ Não é necessário a colocação do caractere nulo ao final da cadeia

gets()

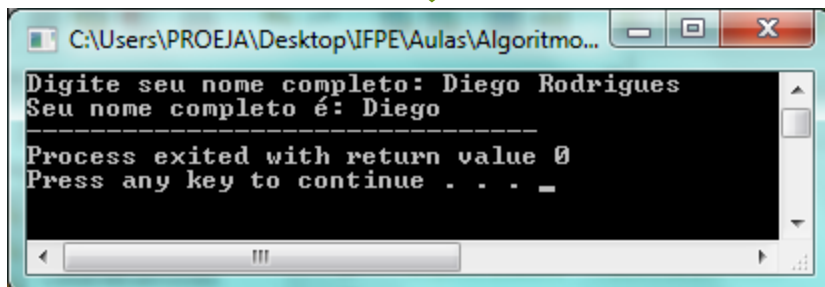
- ▶ A função `scanf()` possui uma limitação
- ▶ Ao ler uma string, a função `scanf()` só lê até o espaço em branco
- ▶ O código no próximo slide ilustra um exemplo

gets()

- ▶ O nome digitado foi “Diego Rodrigues”, no entanto só aparece “Diego” como resultado!!!



```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <locale.h>
4
5  int main(int argc, char *argv[]) {
6      setlocale(LC_ALL, "Portuguese");
7      char nome[40];
8      int i;
9      printf("Digite seu nome completo: ");
10     scanf("%s", nome);
11     printf("Seu nome completo é: %s", nome);
12     return 0;
13 }
```



```
C:\Users\PROEJA\Desktop\IFPE\Aulas\Algoritmo...
Digite seu nome completo: Diego Rodrigues
Seu nome completo é: Diego
-----
Process exited with return value 0
Press any key to continue . . . _
```


gets()

- ▶ Quando na string houver espaços em branco, uma alternativa para o scanf é a função gets()
- ▶ O código anterior deve ser modificado da seguinte forma

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    char nome[50];
    printf("Digite seu nome: ");
    gets(nome);
    printf("Nome: %s", nome);
    return 0;
}
```

fgets()

► Atenção

- A função `gets()` é deprecated, ou seja, seu uso foi descontinuado e é desencorajado.
- A função `gets()` possui uma falha de segurança que pode ser explorada por hackers.
- Portanto, pode-se utilizar `gets` para fins acadêmicos, mas não para fins de uso profissional.
- Uma alternativa segura para o `gets()` é o `fgets()` que veremos no próximo slide

fgets()

- ▶ O código anterior utilizando o fgets()

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    char nome[50];
    printf("Digite seu nome: ");
    fgets(nome, 50, stdin);
    printf("Nome: %s", nome);
    return 0;
}
```

fgets()

► Sintaxe

► fgets(str, n, stream)

- str: Ponteiro para o vetor onde a string será armazenada.
- n: Número máximo de caracteres a serem lidos (incluindo o caractere \0)
- stream: Origem dos dados (geralmente stdin para entrada padrão).

► Atenção

- Enquanto a função gets adiciona apenas “\0” ao final da string, a função fgets adiciona “\n” na penúltima posição e “\0” na última.

Exemplo

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    char frase[100];
    printf("Digite uma frase");
    fgets(frase, 100, stdin);
    printf("A frase digitada foi %s", frase);
    return 0;
}
```

Exercício

- ▶ Crie um programa em que você digita seu nome e o seu sexo. Depois disso, o programa deve dizer:
 - ▶ Bem vindo SEU NOME, caso você seja do sexo masculino
 - ▶ Bem vinda SEU NOME, caso você seja do sexo feminino

Resolução 1

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    system("clear");
    char sexo;
    char nome[51];
    printf("Digite o seu sexo (m, f): ");
    scanf("%c", &sexo);
    fflush(stdin);
    printf("Digite seu nome: ");
    fgets(nome, 51, stdin);
    if(sexo == 'm'){
        printf("Bem vindo %s", nome);
    }else{
        printf("Bem vinda %s", nome);
    }
    return 0;
}
```

Resolução 2

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    system("clear");
    char sexo;
    char nome[51];
    printf("Digite o seu sexo (m, f): ");
    scanf("%c", &sexo);
    fflush(stdin);
    printf("Digite seu nome: ");
    fgets(nome, 51, stdin);
    char mensagem[10] = "Bem vindo";
    if(sexo == 'f'){
        mensagem[8] = 'a';
    }
    printf("%s %s", mensagem, nome);
    return 0;
}
```


Exercício

- ▶ Faça um programa em que você digita uma palavra de **EXATAMENTE** 5 letras e o programa diz qual é a primeira e a última letra

Resolução

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    system("clear");
    char palavra[6];
    printf("Digite uma palavra: ");
    fgets(palavra, 6, stdin);
    printf("A primeira letra da palavra é %c\n", palavra[0]);
    printf("A última letra da palavra é %c\n", palavra[4]);
    return 0;
}
```

Exercício

- ▶ Faça um programa em que você digita uma palavra de **ATÉ** 20 letras e o programa diz qual é a primeira e a última letra
- ▶ Se a palavra pode ter **ATÉ** 20 letras então pode ter 1, 2, 3, 4, 5, 6, ..., 19 ou 20 letras

Resolução 1

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    system("clear");
    char palavra[21];
    printf("Digite uma palavra: ");
    gets(palavra);
    printf("A primeira letra é %c\n", palavra[0]);
    int indice = 0;
    while(palavra[indice] != '\0'){
        indice++;
    }
    printf("A última letra é %c\n", palavra[indice - 1]);
    return 0;
}
```

Resolução 2

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

int main() {
    setlocale(LC_ALL, "Portuguese");
    system("clear");
    char palavra[21];
    printf("Digite uma palavra: ");
    fgets(palavra, 20, stdin);
    printf("A primeira letra é %c\n", palavra[0]);
    int indice = 0;
    while(palavra[indice] != '\0'){
        indice++;
    }
    printf("A última letra é %c\n", palavra[indice - 2]);
    return 0;
}
```

Exercício

- ▶ Fazer programa em que você digita uma palavra e o programa diz quais as posições das vogais na palavra

Exemplo

- ▶ Crie um programa em que o usuário digita uma frase de até 100 letras e o programa conta quantas palavras existem na frase

Exercício

- ▶ Faça um programa em que você digita seu nome e o programa diz como é seu nome em ordem inversa

Obrigado

