

DISCIPLINA: ESTRUTURAS DE DADOS

PROGRAMA DE ARTICULAÇÃO DA FORMAÇÃO PROFISSIONAL
MÉDIA E SUPERIOR (AMS)

VETOR DE CARACTERES

- Prof. Alexandre Ponce de Oliveira
alexandreponceo@gmail.com
Aulas 03 e 04

Vetor de Caracteres - Strings

Strings são vetor de caracteres, têm o seu último caractere como um '\0'.

A declaração geral para uma string é:

```
char nome_da_string[tamanho];
```

```
char nome[30];
```

```
char rua[25];
```

O número entre colchetes define o número máximo de caracteres que a string poderá armazenar.

Vetor de Caracteres - Strings

Por exemplo, uma string de 7 posições:

```
char nome[7];
```

E preenchida com a palavra José ficaria da seguinte forma:

J	o	s	é	\0
---	---	---	---	----	-----	-----

Strings – Funções

`gets()`

Recebe uma string do teclado.

Coloca o terminador nulo na string, quando você aperta a tecla "Enter".

`puts()`

Envia uma string para uma saída padrão.

Pula para a próxima linha.

Exemplo – Funções: gets() e puts()

```
#include <stdio.h>
#include <conio.h>
main ()
{
    char nome[30]="Alexandre";
    char teste[100];
    puts(nome);
    puts ("Digite o seu nome: ");
    gets (teste);
    printf ("\n\n Ola %s",teste);
    getch();
}
```

Strings – Funções: putchar()

Imprime um caractere na saída padrão na posição do cursor.

Após a impressão o cursor permanece na posição seguinte ao caractere impresso.

Exemplo:

```
# include <stdio.h>
main ( ) {
    char ch='A';
    putchar(ch);
    putchar (65); //Caracter associado ao código ASCII 65
    putchar ('B');
    getch();
}
```

Strings – Funções: getchar()

Recebe um caractere da entrada padrão.

Exemplo:

```
# include <stdio.h>
main ( ) {
    char ch;
    puts ("Digite um character");
    ch = getchar();
    printf ("\nVoce digitou = %c",ch);
    getch();
}
```

Strings – Funções: getch() e getche()

Recebe um caractere da entrada padrão.

Com esta função getche() não é necessário digitar a tecla “ENTER” após a digitação do caractere.

A função getch() e getche() exige a biblioteca <conio.h>

Diferença:

getch(): não exibe o caractere digitado;

getche(): exibe o caractere digitado;

Exemplo – Funções: getch() e getche()

```
# include <stdio.h>
# include <conio.h>
main ( ) {
    char ch;
    puts ("Digite um character");
    ch = getch(); // ou ch = getche();
    printf ("\nVoce digitou = %c",ch);

    getch();
}
```

Strings e Caractere – Exemplo

```
#include <stdio.h>
#include <conio.h>
main()
{
    char str[10] = "Joao";
    printf("\n\nString: %s", str);
    printf("\nSegunda letra: %c", str[1]);
    str[1] = 'U';
    printf("\nAgora a segunda letra e: %c", str[1]);
    printf("\n\nString resultante: %s", str);
    getch();
}
```

Manipulação de Strings

Para manipular string é necessário utilizar as funções da biblioteca `<string.h>`

A comparação entre strings é feita caractere a caractere, utilizando para isso o valor do código ASCII de cada caractere.

strlen(string): retorna o número de caracteres de uma string qualquer, não levando em conta o caractere nulo `'\0'`.

strcat(string_destino,string_origem)

Adiciona o conteúdo da string origem no final da string destino.

Strings e Caractere – Exemplo strlen()

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
int main ()
{
    char str1[100];
    printf ("Digite seu nome: ");
    gets (str1);
    printf ("\nSeu nome tem %d letras",strlen(str1));
    getch();
}
```

Strings e Caractere – Exemplo strcat()

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
int main ()
{
    char str1[100],str2[100];
    printf ("Digite seu nome: ");
    gets (str1);
    printf ("Digite seu sobrenome: ");
    gets (str2);
    strcat (str1," ");
    strcat (str1,str2);
    printf ("\nSeu nome completo e: %s",str1);
    getch();
}
```

Manipulação de Strings

strcmp(str1,str2)

Compara **string1** com **string2**. Retorna 0 se **string1** é igual a **string2**. Caso **string1** seja menor (em função do código ASCII) que **string2** retorna um número menor que 0, caso contrário (**string1** maior que **string2**) retorna um número maior que 0.

strcpy(string1,string2)

Copia o conteúdo da **string2** para **string1**.

Strings e Caractere – Exemplo strcmp()

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
int main ()
{
    char str1[100],str2[100];
    int teste;
    printf ("Digite o nome de uma cor: ");
    gets (str1);
    printf ("\n\nDigite o nome de uma cor novamente: ");
    gets (str2);
    teste = strcmp(str1,str2);
    if (teste == 0)
        printf ("\n\nAs cores sao iguais!!!");
    else
        printf ("\n\nAs cores sao diferentes");
    getch();
}
```

Strings e Caractere – Exemplo strcpy()

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
int main ()
{
    char str1[100],str2[100];
    printf ("Entre com uma string: ");
    gets (str1);
    strcpy (str2,str1);
    printf ("\nVoce digitou a string: %s",str2);
    getch();
}
```


Funções para Strings

strupr(string): Transforma todos os caracteres da string para maiúsculo.

strlwr(string): Transforma todos os caracteres da string para minúsculo.

toupper(variavel) : Transforma um caractere em maiúsculo.

tolower(variavel): Transforma um caractere em minúsculo.

Para as funções toupper e tolower é necessário o uso da biblioteca <ctype.h>

Strings e Caractere – Exemplo

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
main() {
    char nome[40];
    puts("\nDigite um nome minuscuro:");
    fflush(stdin);
    gets(nome);
   strupr(nome);
    puts(nome);
    strlwr(nome);
    puts(nome);
    getch();
}
```

Strings e Caractere – Exemplo

```
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
main() {
    char a,y,x;
    printf("\n");
    printf("Digite um caracter maiusculo: ");
    a = getchar();
    printf("\n");
    y = tolower(a);
    printf("Caractere convertido com tolower = %c \n", y);
    x = toupper(y);
    printf("Caractere convertido com toupper = %c \n", x);
    printf("\n");
    getch();
}
```

Exercícios

- 1) Escreva um programa que leia duas strings e imprime as duas strings na tela. Imprima também a segunda letra de cada string.
- 2) Faça um programa em C que lê um string de 5 caracteres e escreve a string invertida.
- 3) Faça um programa em C que lê um string de 4 caracteres e inverte a primeira letra do string com a última. O programa deve escrever a string original e a alterada.
- 4) Faça um programa que leia uma string e faça uma copia para uma outra string.