**STRINGS EM C**

Uma string é uma seqüência de caracteres utilizada para o armazenamento de texto. Em C não existe um tipo string explícito, não existe uma palavra reservada que declare uma variável para armazenar uma string. Na linguagem C strings são representadas por vetores de caracteres que possuem um caracter que indica o término de seu conteúdo, o caracter nulo ‘\0’ . Por serem representadas como vetores de caracteres , a elas se aplicam todas as propriedades e limitações de um vetor. Sendo assim, é possível alterar conteúdo da posições específicas, iterar sobre elas e etc.

**FUNÇÕES DE STRING**

**strlen(string)**

**A função que calcula o tamanho (quantidade de caracteres) da string:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char vetor[20];

scanf("%s",&vetor);

int tamanho=strlen(vetor);

printf("%d", tamanho);

return 0;}

**ENTRADA:**

UNIVERSIDADE

**SAÍDA:**

12

**strcpy(destino, origem)**

**A função que copia o conteúdo de uma string origem para o destino:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char origem[13];

char destino[13];

scanf("%s %s",origem,destino);

printf("DESTINO ORIGINAL: %s \n", destino);

strcpy(destino, origem);

printf("NOVO DESTINO: %s", destino);

return 0;}

**ENTRADA:**

UFU

UNIVERSIDADE

**SAÍDA:**

DESTINO ORIGINAL: UFU

NOVO DESTINO: UNIVERSIDADE

**strncpy(destino, origem, n)**

**A função que copia n caracteres de uma string origem para o destino:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char origem[23];

char destino[10];

scanf("%s",&origem);

strncpy(destino, origem,10);

destino[10] = '\0';

printf("%s", destino);

return 0;}

**ENTRADA:**

RESOLUÇÃO DE PROBLEMAS

**SAÍDA:**

RESOLUÇÃO

**strcat(string1, string2)**

**Realiza a concatenação de string2 em string1**

#include <stdio.h>

#include <stdlib.h>

int main(){

char string1[23]="Resolucao ";

char string2[13]=”de Problemas”;

strcat(string1, string2);

printf("%s\n", string1);

return 0;}

**SAÍDA:**

Resolucao de Problemas

**strncat(string1, string2, n)**

**Realiza a concatenação de string2 em string1 considerando apenas n caracteres:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char string1[23]="Resolucao ";

char string2[13]=”de Problemas”;

strncat(string1, string2, 12);

printf("%s\n", string1);

return 0;}

**SAÍDA:**

Resolucao de Problemas

**strcmp(string1, string2)**

**Faz a comparação alfabética de duas strings:**

**retornando: 1 se a primeira for maior, 0 se forem iguais, -1 se a segunda for maior:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char string1[8]="ABACATE";

char string2[8]=”ARARA”

printf("%d\n", strcmp(string1, string2));

return 0;}

**SAÍDA:**

-1

**strncmp(string1, string2, n)**

**Faz a comparação alfabética de duas strings considerando apenas n caracteres:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char vetor[5]="MARIO";

char vetor2[5]=”MARIA”

printf("%d\n", strncmp(vetor, vetor2, 4));

printf("%d\n", strncmp(vetor, vetor2, 5));

return 0;}

**SAÍDA:**

0

-14

**strrev(string)**

**Inverte a string sobre ela mesma:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char vetor[10]="RESOLUCAO";

printf("%s\n", strrev(vetor));

return 0;}

**SAÍDA:**

OACULOSER

**Para verificar se uma palavra é palidromo:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char vetor[6] = "ARARA";

char vetor2[6];

strcpy(vetor2, vetor);

strrev(vetor2);

if(strcmp(vetor, vetor2) == 0){

printf("%s EH PALINDROMO!", vetor);

}

return 0;}

**SAÍDA:**

ARARA EH PALINDROMO!

**strchr(string, caracter)**

**Considera a string a partir da primeira ocorrência do caracter informado:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char vetor[10]="RESOLUCAO";

printf("%s\n", strchr(vetor, ‘O’));

return 0;}

**SAÍDA**:

OLUCAO

**strstr(string, caracter)**

**Considera a string a partir da primeira ocorrência da substring:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char vetor[10]="RESOLUCAO";

printf("%s\n", strstr(vetor, "OL"));

return 0;}

**SAÍDA:**

OLUCAO

**strtok(string, delimitador)**

**Divide a string, devolvendo a cada chamada a substring até o delimitador:**

#include <stdio.h>

#include <stdlib.h>

int main(){

char vetor[21]="RESOLUCAO-60H-SABADO";

char \*auxiliar;

auxiliar = strtok(vetor, "-");

while(auxiliar != NULL) {

printf("%s\n", auxiliar);

auxiliar = strtok(NULL, "-");

}

return 0;}

**SAÍDA:**

RESOLUCAO

60H

SABADO

**Validações com Expressão Regular**

Para obtermos o resultado de uma entrada do teclado por meio do scanf precisamos utilizar as conversões: %s, %c, %lc, %ls, entre outras.

A sintaxe do scanf é a seguinte: int scanf(“[conversões de formato]”, argumentos);

argumentos: ponteiros para o dado, separados por vírgula.

Uma limitação de utilizar a conversão string %s é que ela lida com os espaços em branco como se fossem ‘\n’.

Para limitarmos a quantidade de caracteres de uma string1 já existente a 20 caracteres sem espaço:

scanf(“%20s”, string1);

Caso deseje se permitir caracteres em branco:

scanf(“%20c”, string1);

Mas não queremos que ‘\n’ (quebra de linha) seja incluída em nossa string1, para isso utilizamos expressões regulares :

scanf(“%20[^\n]\n”, string1);

Há um tipo de conversão “%[“ que não pula espaços em branco, mas permite limitar a entrada a uma expressão regular.

Por exemplo, digamos que seja necessário pagar um número de telefone (composto somente por caracteres numéricos):

scanf(“%[123456789]”, string1);

Ou somente caracteres alfabéticos:

scanf(“%[A-z]”, string1);

Mas devidos a faixa de valores não alfabéticos é melhor utilizar: scanf(“%[A-Za-z]”, string1);

*Os créditos da elaboração deste material são de:  
 - Gustavo Teixeira Perche Mahlow*

*- Marco Antonio da Silva Rodrigues*

*- Tadeu Rodrigues dos Santos Braga*