Linguagem C

Strings

Strings em Linguagem “C”

- **Definição de string**
  - Strings são seqüências de caracteres diversos. São conhecidos por “literais” na teoria de algoritmos estruturados, sendo representados entre aspas. Alguns exemplos de strings:
  - “Fulano da Silva”,
  - “? Interrogação? “,
  - “1,234”,
  - “0”.
  - Em C, strings são representadas através de vetores de caracteres, terminados com o caractere de fim de string cujo valor na tabela ASCII é zero (0 ou \0).

Strings em Linguagem “C”

- **Declaração de string**
- Um vetor em C que pretenda armazenar uma string **n** caracteres deve ser alocado com **n+1** posições do tipo *char* para conter o terminador de string. A inicialização de uma string pode ser efetuada com o uso de uma seqüência de caracteres entre aspas.
- Exemplos de declarações de string:

  char frase[] = “Primeira frase”; /*Inicialização sem a dimensão */

  char frase[15] = “Primeira frase”;

  char frase[6] = {‘T’, ‘e’, ‘s’, ‘t’, ‘e’, 0); /* inicializado como

- um vetor de caracteres comum, ‘forçando’ o caracter terminador */
- No caso do primeiro e do segundo exemplo, a representação do vetor da string *frase* é:
- ‘P’ ‘r’ ‘i’ ‘m’ ‘e’ ‘i’ ‘r’ ‘a’ ‘ ’ ‘f‘ ‘r’ ‘a’ ‘s’ ‘e’ 0
- Onde cada quadrado representa um byte de memória (tamanho de um *char*).

Strings em Linguagem “C”

- **Operações sobre string**
- String não é um tipo primitivo da linguagem C, por isso as seguintes operações **NÃO** são válidas:

```
char str1[10];
char str2[] = “Palavra 2”;
str1 = str2 /* ERRO! Não copia str2 em str1 */
if (str1 == str2) /* ERRO! Não compara str1 com str2 */
{
....
}
```

- Para operar sobre strings são utilizadas funções da biblioteca **string.h**.
- Dezenas de funções com diversas variações.....

Strings em Linguagem “C”

- **strlen**
- Protótipo:

  int strlen (char *string)
- Descrição: Retorna o número de caracteres de uma string (exceto o caractere de fim de string).
- Exemplo:

  char nome[] = “Fulano”;

  printf (“O nome possui %d letras”, strlen (nome));

  .....

Strings em Linguagem “C”

- **strcpy**
- Protótipo:

  char *strcpy (char *string1, char *string2)
- Descrição: Copia o conteúdo de *string2* em *string1* e retorna o endereço de string.
- Exemplo:

  char str1[10];

  char str2[] = “Palavra”;

  strcpy (str1, str2); /* Agora str1 também contém “Palavra” */

Strings em Linguagem “C”

- **strcmp**
- Protótipo:

  int strcmp (char *string1, char *string2)
- Descrição: Compara os conteúdos de string1 e string2 caracter a caracter e retorna
  - 0 se string1 = string2
  - <0 se string1 < string2
  - >0 se string1 > string2
- Exemplo:

  char nome1[] = “Fulano”

  char nome2[] = “Beltrano”;

  if (strcmp (nome1, nome2) == 0)

  {printf (“Nomes são iguais”);}

  else

  {printf (“Nomes são diferentes);}

Strings em Linguagem “C”

- **strcat**
- Protótipo:

  strcat (char *string_destino, char *string_origem)
- Descrição: Concatena os conteúdos de string1 e string2
- Exemplo:

  char nome1[] = “Fulano”

  char nome2[] = “da Silva”;

  char nome_completo[30]=“”;

  strcat(nome_completo, nome1);

  printf (“%s\n”, nome_completo);

  strcat(nome_completo, nome2);

  printf (“%s\n”, nome_completo);

Este programa imprimirá:

Fulano

Fulano da Silva

Strings em Linguagem “C”

- **atoi**
- Protótipo:

  int atoi(char *string)
- Descrição: Converte o string em um número inteiro
- Exemplo:

```
int a;
a = atoi(“10”);
```

- **atof**
- Protótipo:

  double atof(char *string)
- Descrição: Converte o string em um número em ponto flutuante(double)
- Exemplo:

```
double d;
d = atof(“101.15”);
```

Strings em Linguagem “C”

- **gets**
- Protótipo:

  void gets (char *string1)
- Descrição: Recebe uma string via teclado e armazena em *string1*. Os caracteres são armazenados até que o enter seja pressionado.
- Exemplo:

  char nome[10];

  gets (nome);

Strings em Linguagem “C”

- Observações:a função gets() permite que o usuário forneça mais caracteres do que os que podem ser armazenados no vetor(o que pode causar um erro). Para evitar este problema, pode-se utilizar a função *fgets*:

```
char nome[10];
fgets(nome, 10, stdin); /* ‘stdin’ é um arquivo aberto por padrão, relacionado aos dados
    digitados via teclado */
```

- No exemplo mostrado, *fgets* receberia 9 caracteres (ou até que o usuário teclasse enter) e armazenaria os dados digitados na string *nome,* adicionando o caracter terminador de string. É importante observar que, caso o usuário digitasse enter antes de 9 caracteres, o caracter de nova linha (‘\n’) também seria armazenado no vetor.

```
scanf (“%s”, str); /* Recebe uma string até que o primeiro espaço seja inserido */
```

Funções em Linguagem “C”

- **Exercício 1:**
  - Criar um programa capaz de calcular o tempo entre dois horários quaisquer de um determinado dia.
  - O programa deve ler dois horários, no formato HH:MM:SS(dois dígitos para hora, dois para minutos e dois para segundos). O programa deve verificar se o horário é válido(HH entre 00 e 23, MM entre 00 e 59, SS entre 00 e 59).
  - O programa deve ter uma função para calcular a quantidade de segundos em um horário, e outra função para calcular e imprimir a quantidade de horas, minutos e segundos em uma quantidade de segundos;

Funções em Linguagem “C”

- **Exercício 2:**
- Faça um programa que dadas 2 palavras, determine:
  - Se as palavras são iguais;
  - Caso as palavras sejam diferentes, qual delas tem maior comprimento (não esquecer a possibilidade de existirem palavras diferentes de mesmo tamanho);
  - Verifique se a segunda palavra é uma sub string da primeira:
  - Exemplo: Palavra 1= **casa**mento

    Palavra 2 = **casa**

Funções em Linguagem “C”

- **Exercício 3:**
- Faça um programa que leia 10 nomes completos do teclado, e ao final apresente uma listagem dos nomes no formato:
  - Sobrenome, Primeiro_Nome;

Exemplo:

João Alberto Fabro

José da Silva

Josué dos Santos

Fabro, João

Silva, José

Santos, Josué