Introdução à Computação II – AULA 06

BCC Noturno - EMA896115B

Prof. Rafael Oliveira
olivrap@gmail.com

Universidade Estadual Paulista
“Júlio de Mesquita Filho”
UNESP

Rio Claro 2014 (Sem 2)

Recordando

- Comando while – repetição
- Funções do tipo **void** – funções que não retornam valor
- Strings – usados para guardar cadeias de carcteres, como nome, endereço, etc

Recordando – strings

- Um string é uma sequência de caracteres legíveis
- O primeiro parâmetros do `scanf` e do `printf` são sempre um string
- Podemos declarar variáveis e guardar strings nelas
- Para isso usá-se o tipo `char []`
- Por exemplo char nome[50];
- O número representa o tamanho máximo do string que se pode colocar na variável

Entrada e saída

- Para ler e escrever um string usa-se o formato `"\%s"`
- ```
  char nome[50];
  scanf("%s", nome); // sem o & !!!!
  ```
- ```
  printf("O nome lido foi: %s", nome);
  ```

String – exemplo

```
#include <stdio.h>

int main()
{
char nome[50];

   printf("Digite seu nome ==> ");
   scanf("%s", nome);
   printf("O nome lido foi %s\n", nome);
}
```

String – vetor de caracteres

- Na verdade, um string é o que chamamos um “vetor” de caracteres (char)
- Um caractere é um número, entre 0 e 255
- Um vetor é uma sequência de caracteres juntinhos
- Delamaro
  - | ’D’ | ’e’ | ’l’ | ’a’ | ’m’ | ’a’ | ’r’ | ’o’ | ’\0’ |
  - | 68 | 101 | 108 | 97 | 109 | 97 | 114 | 111 | 0 |
- O que define o código a ser usado é o ambiente de execução (Windows)

Tabela de codificação

- Como descobrir a codificação usada no seu sistema?
-

```
#include <stdio.h>

int main()
{
int c;

   c = 0;
   while ( c < 256 )
   {
      printf("%d ==> %c\n", c, (char) c);
      c++;
   }
}
```

Elementos de um vetor

- É possivel acessar cada elemnto de um vetor, de forma individual
- Para isso é preciso dizer qual é o deslocamento (índice) do elemento desejado
- O primeiro elemento tem deslocamento 0, o segundo, 1, e assim por diante
- `char nome[30]`
-

0	1	2	3	4	5	6	7	8	...	29
D	e	l	a	m	a	r	o	\0	...	...

- nome[0], nome[1], nome[2],... nome[29];

Elementos – exemplo

```
#include <stdio.h>

/* Vamos contar o numero de letras A ou a */

int main()
{
char nome[30];
int i, contador;

   scanf("%s", nome);
   ...
}
```

Elementos – exemplo

```
#include <stdio.h>

/* Vamos contar o numero de letras A ou a */

int main()
{
char nome[30];
int i, contador;

   scanf("%s", nome);
   i = 0;
   contador = 0;
   while ( i < 30 )
   {
       if ( nome[i] == 'a' || nome[i] == 'A' )
          contador++;
       i++;
   }
   printf("Achei %d letras A\n", contador);
}
```

Cuidado

- O nome (string) só é significativo até que apareça o marcador de final
- Ou seja, até que apareça o caractere '\0'
- então, como fazemos para contar o número de letras A, se não sabemos até onde ir procurando?

Elementos – exemplo

```
#include <stdio.h>

/* Vamos contar o numero de letras A ou a */

int main()
{
char nome[30];
int i, contador;

   scanf("%s", nome);
   i = 0;
   contador = 0;
   while ( nome[i] != '\0' )
   {
       if ( nome[i] == 'a' || nome[i] == 'A' )
          contador++;
       i++;
   }
   printf("Achei %d letras A\n", contador);
}
```

String não é tipo simples

- Infelizmente, string não é manipulado de forma tão simples como os tipos simples (double, int, char...)
- Coisas que **não** podemos fazer
  - Atribuir valor: `nome = "Delamaro"`
  - Comparar: `if ( nome == "Delamaro" )`
  - Concatenar: `nome = nome + "Delamaro"`
- Mas para tudo isso existem funções específicas

Funções de strings

- Comparar dois strings: `strcmp(s1, s2)`
  - retorna 0 se s1 é igual a s2
  - retorna valor negativo se s1 < s2
  - retorna valor positivo se s1 > s2
  - ```
    if (strcmp(nome, "Delamaro") == 0)
          printf("%s\n", nome);
    ```
- Atribuir um string para a variável: `strcpy(s1, s2)`
  - `strcpy(nome, "Delamaro");`
- Concatenar: `strcat(s1, s2)`
  - ```
    strcpy(nome, "Marcio");
    strcat(" Delamaro");
    ```

Algumas observações

- Os vetores (strings) podem ser declarados como variáveis locais ou globais
- Se o vetor for muito grande, procure declará-lo como variável global
- Valem as mesmas regras de inicialização das variáveis escalares
  - Vetores globais são inicializados com zero
  - Vetores locais não são inicializados, portanto seu valor inicial é desconhecido
- Um string não é um valor. É um espaço de memória...

Que quer dizer isso?

- Não dá para retornar um string, por exemplo.
- ```
  char [] le_string()
  {
  char nome[30];

     scanf("%s", nome);
     return nome;
  }
  ```
  ```

Que quer dizer isso?

- Dá para modificar os elementos de um vetor dentro de uma função
- ```
  void le_string(char s[])
  {
     scanf("%s", s);
  }

  int main()
  {
  char nome[30];

      le_string(nome);
  }
  ```
- Está dizendo que a variável “nome” vai ser usada dentro da chamada da função. Não o seu valor
  ```

Exemplo – parâmetros

```
#include <stdio.h>
#include <string.h>

void  f(char s[], int m)
{
   s[0] = 'd';
   m = 10;
}

int main()
{
char nome[30];
int m;

    strcpy(nome, "Delamaro");
    m = 0;
    f(nome, m);
    printf("%s -- %d\n", nome, m);
}
```

Exercício

- Escreva uma função que substitui todos os espaços em branco de um string por um asterisco.

Exercício

- Escreva uma função que substitui todos os espaços em branco de um string por um asterisco.
- ```
  void  troca_espaco(char s[])
  {
  int k = 0;

     while ( s[k] != '\0' )
     {
          if ( s[k] == ' ' )
            s[k] = '*';
          k++;
     }
  }
  ```
  ```

Muito além do string

- Vetores podem ser usados para agrupar outros tipos de dados, além de **char**
- Valores de inteiros e floats (double) por exemplo podem ser colocados em vetores
- É um recurso muito utilizado
- Permite que uma série de dados seja armazenada e processada de forma semelhante
- `double nota[30]`
-

0	1	2	3	4	5	6	7	8	...	29
7.4	8.5	8.0	4.9	5.5	7.7	3.8	8.3	0.1	...	6.5

- nota[0], nota[1], nota[2],... nota[29];

Vetores – exemplo

```
int main()
{
double notas[30];
int n, i;
    printf("Quantas notas serao digitadas? ");
    scanf("%d", &n);
    i = 0;
    while (i < n )
    {
        printf("Digite a nota %d ==> ", i);
        scanf("%lf", &notas[i]);
        i++;
    }
    i = 0;
    printf("As notas digitadas foram: ");
    while (i < n )
    {
        printf("%5.2lf ", notas[i]);
        i++;
    }
    printf("\n");
}
```

Exercícios

- Escreva uma função para computar a soma dos elementos de um vetor de `doubles`
- Escreva uma função para computar a média dos elementos de um vetor de `doubles`
- Escreva uma função para computar o devio-padrão dos elementos de um vetor de `doubles`

Soluções

```
#include <stdio.h>
#include <math.h>
double soma(double[], int);
double media(double[], int);
double devp(double[], int);

int main()
{
double notas[30];
int n, i;
    printf("Quantas notas serao digitadas? ");
    scanf("%d", &n);
    i = 0;
    while (i < n )
    {
        printf("Digite a nota %d ==> ", i);
        scanf("%lf", &notas[i]);
        i++;
    }
    printf("As soma é: %lf\n", soma(notas, n));
    printf("As media é: %lf\n", media(notas, n ));
    printf("O desvio-padrao é: %lf\n", devp(notas, n));
}
```

Soluções

```
double soma(double x[], int k)
{
int i = 0;
double s = 0.0;

    while ( i < k )
    {
        s = s + x[i];
        i++;
    }
    return s;
}
```

Soluções

```
double media(double x[], int k)
{
    return soma(x,k) / k;
}
```

Soluções

```
double devp(double x[], int k)
{
int i = 0;
double med, dv, aux;
    dv = 0.0;
    med = media(x, k);
    while ( i < k )
    {
        aux = x[i] - med;
        dv = dv + (aux * aux);
        i++;
    }
    return sqrt(dv / (k-1));
}
```