Funções

Introdução à Ciência da Computação I

Prof. Denis F. Wolf

Função

- Agrupa um conjunto de comandos e associa a ele um nome (identificador).
- O emprego deste nome como um comando corresponde a uma chamada a função para sua execução
- Após sua execução, o programa retoma sua execução no ponto imediatamente após ao momento da chamada à sub-rotina
  - O momento de retorno ao ponto em que a função foi invocada é chamado de retorno.

Parâmetros de uma Função

- Durante a chamada de uma função pode informar valores (argumentos) para o processamento da função
- Argumentos = lista de expressões
  - Lista pode ser vazia
  - Lista aparece entre parênteses após o nome da função
- Ex: **int** soma (**int** x, **int** y) {
  }

O Retorno da Função

- No seu retorno, uma função pode entregar resultados ao programa que a chamou
  - Exemplo: **return (resultados);**
  - O valor da variável local *resultados* é passado de volta como o valor da função
- Valores de qualquer tipo podem ser retornados
  - Funções predicado: funções que retornam valores
  - Procedimentos: funções que não retornam valores
  - Exemplo: **void** function (**int** x)

Elementos que compõem uma função

- Funções são definidas de acordo com a seguinte sintaxe:

  tipo_de_resultado nome (lista de parâmetros) {
  /* corpo de função */
  }
- Onde:
  - Tipo de resultado:
    - Quando a função é um procedimento, usa-se a palavra chave **void**
    - Procedimentos não retornam valor
  - Lista de parâmetros
    - Funcionam como variáveis locais com valores iniciais
    - Quando função não recebe parâmetros, a lista de parâmetros é substituída pela palavra **void**

Exemplo de Função

```
/* definição da função soma */
int soma(int a, int b) {
    int res;
    res = a + b;
     return (res);
}

void main() {
    int resultado;
    resultado = soma(12,8);  /* chamada */
    printf("O valor da soma é: %d\n", resultado );
    return 0;
}
```

Realizando uma chamada

- Quando invocamos uma função:
  - Cada expressão na lista de argumentos é avaliada
  - O valor da expressão é convertido, se necessário, para o tipo do parâmetro formal
  - Este tipo é atribuído ao parâmetro formal correspondente no início do corpo da função
  - O corpo da função é executado
  - Ao término, o fluxo de execução retorna imediatamente após o ponto de chamada

7

Exercícios

1) Criar uma função que recebe as dimensões da base e da altura de um triângulo e calcula sua área.
2) Criar uma função que converte a temperatura de Celsius para Fahrenheit.
3) Crie um programa que leia as coordenadas de 2 pontos no plano cartesiano e retorne a distância entre eles através de uma função

Realizando uma chamada

Os comandos do corpo da função são executados até que:

a) Se encontre um comando **return**
b) Não existirem mais comandos para serem executados

O valor da expressão **return**, se ele existir, é avaliado e retornado como valor da função

O programa que chamou continua sua execução

8

Exercícios

4) Crie uma função que receba um número N e retorne a somatória de 1 até N (1+2+3+4...+N).

5) Crie duas funções que recebam 3 números inteiros e retornem o máximo divisor comum (m.d.c.) e o mínimo múltiplo comum (m.m.c.) desses números. Crie um programa que leia 3 números inteiros e mostre o mmc e o mdc desses números.

Protótipos

- Antes de usar uma função em C, é aconselhável declará-la para que o compilador tome conhecimento de sua existência e verifique se estão corretas as chamadas
  - A declaração apenas indica a *assinatura* da função.
- Esse tarefa é realizada especificando seu protótipo
  - Tem a mesma forma que a função, só que substitui o corpo por um (;)
  - Nomes das variáveis de um parâmetro são opcionais
    - Fornecê-los ajuda a leitura do programa

9