# Programação de Computadores II

## Funções

**Gil Eduardo de Andrade**

# Função

## Definição

- É conjunto de comandos (linhas de código) agrupado num bloco devidamente nomeado, que ao ser invocado executa todas as instruções nele contidas;

## Motivos – Utilização

- Permite reaproveitamento do código construído;
- Evita que um trecho de código seja repetido várias vezes dentro de um mesmo programa;

# Função

## Motivos - Utilização

- Permite a alteração de código de uma forma mais rápida – é preciso alterar apenas dentro da função;
- Blocos do programa não ficam muito grandes sendo mais fáceis de se entender;
- Facilita leitura do código-fonte;
- Separar a lógica do programa em “blocos” que podem ser compreendidos de forma isolada;

# Funções

**Sintaxe:**

- ***tipo*** *nome_da_funcao* (***tipo*** *parametro*);
  - ***tipo***: indica o tipo de retorno da função (*int*, *char*), ou seja, o valor retornado como resultado da execução da função;
  - *nome_da_funcao*: indica o nome pelo qual a função deve ser chamada (invocada);
  - ***tipo*** *parametro:* indica o *tipo* e *nome* do parâmetro (valor) que será passado a função;

# Função

## Implementação / Utilização

- Divide-se em três etapas:
  - ***Declaração ou protótipo:*** efetuada antes função *main()* para indicar que uma nova função está sendo criada adicionalmente as já disponibilizadas pela linguagem;
  - ***Chamada ou invocação:*** efetuada dentro do fluxo de código com intuito de executar a codificação definida para função
  - ***Implementação:*** efetuada após a função *main()* para especificar o código que será executado quando a função for chamada / invocada;

# Função

**SINTAXE**

```
// Protótipo da Função
tipo_da_funcao nome_da_função(parametros);

// Chamada da Função
nome_da_função(parametros);

// Implementação da Função
tipo_da_funcao nome_da_função(parametros)  {
    // Implementação
}
```

# Função

## Exemplo de implementação (sem retorno e sem parâmetro )

```c
#include <stdio.h>

// variáveis globais
int soma, valor_a, valor_b;
// protótipo da função – declaração
void somar();

int main()
{
        printf("Entre com valor a: ");
        scanf("%i", &valor_a);
        printf("Entre com valor b: ");
        scanf("%i", &valor_b);
        // Chamada ou invocação da função
        somar();

        return 0;
}

// implementação da função
void somar() {
        soma = valor_a + valor_b;
        printf("Soma: %i\n", soma);
}
```

Declara as variáveis globais "*valor_a*" e "*valor_b*" que podem ser acessadas por qualquer função (*main()*, *somar()*, etc.)

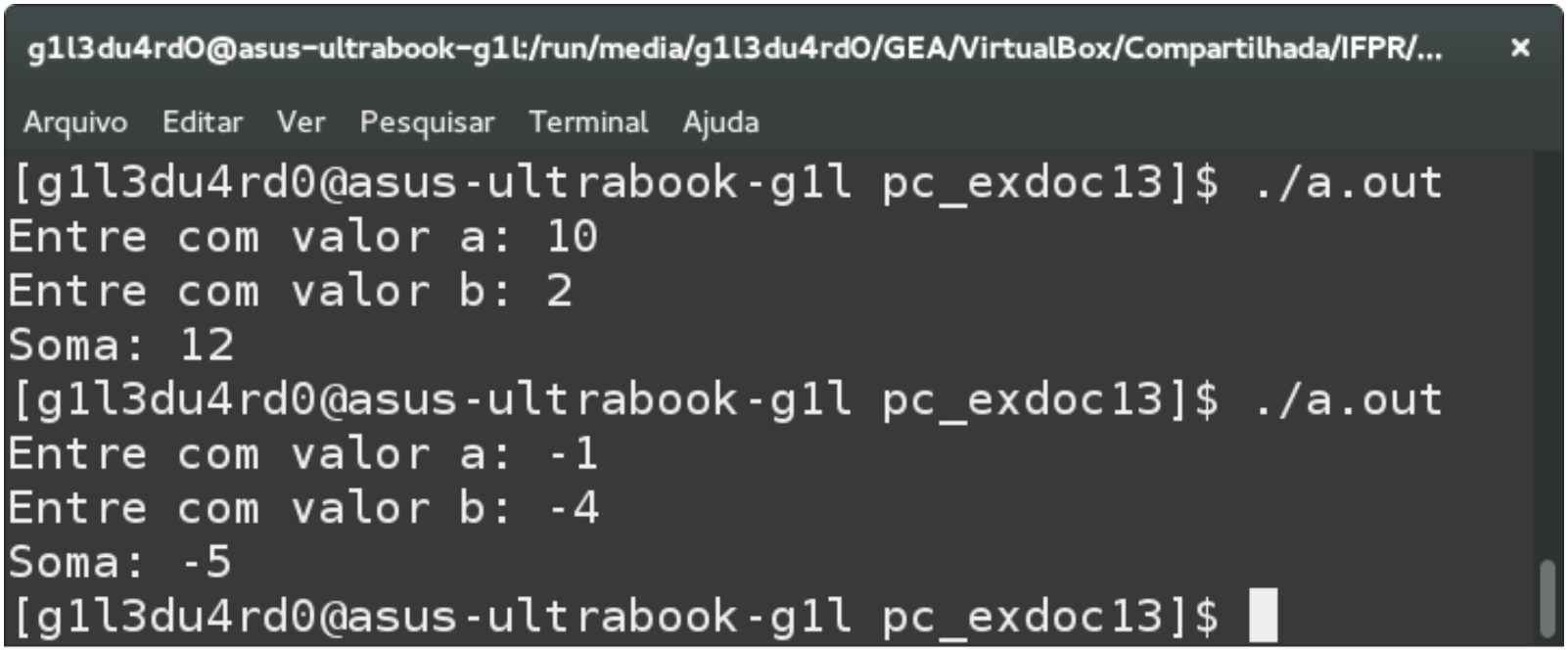
Declara (protótipo) a função *void somar()* que não possui parâmetros *()* e não retorna valor algum (*void*);

Chama (invoca) a função *somar()* dentro da função *main()*, neste momento fluxo de código é direcionado para implementação da função *somar()* logo abaixo da *main()*;

Implementação da função *somar()*, código executado quando ela é chamada. Ao término da execução o fluxo de código retorna para linha logo abaixo da chamada da função (*return 0;*).

# Função

## Exemplo de implementação (sem retorno e sem parâmetro )



# Função

## Exemplo de implementação (sem retorno e com parâmetro )

```c
// protótipo da função – declaração
void somar(int a, int b);

int main()
{
        // variáveis locais
        int valor_a, valor_b;

        printf("Entre com valor a: ");
        scanf("%i", &valor_a);
        printf("Entre com valor b: ");
        scanf("%i", &valor_b);
        // Chamada ou invocação da função
        somar(valor_a, valor_b);

        return 0;
}

// implementação da função
void somar(int a, int b) {

        // variável local
        int soma = a + b;
        printf("Soma: %i\n", soma);
}
```

Declara a função ***void somar(int a, int b)*** que agora possui os parâmetros inteiros "***a***" e "***b***" mas continua sem *retorno*;

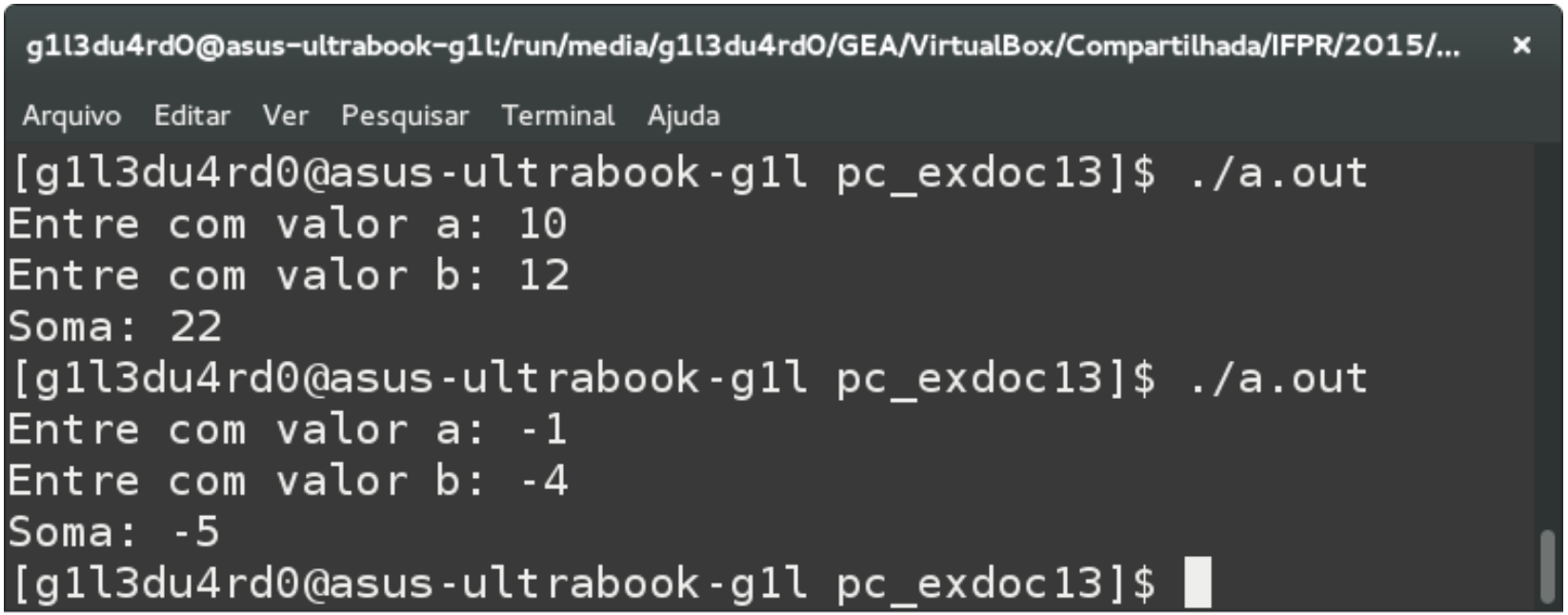
Declara as variáveis locais "***valor_a***" e "***valor_b***" que serão passadas a função ***somar()***, já que não podem ser acessadas pela mesma;

Invoca a função ***somar()*** passando os valores armazenados em "***valor_a***" e "***valor_b***" que serão copiados para dentro dos parâmetros "***a***" e "***b***" ***respectivamente***;

Implementação da função ***somar()***, ela recebe os valores "***a***" e "***b***" utilizados na execução do código. Ao término da sua execução o fluxo de código volta a linha logo abaixo da chamada da função (*return 0;*).

# Função

## Exemplo de implementação (sem retorno e com parâmetro )



# Função

## Exemplo de implementação (com retorno e com parâmetro )

```c
// protótipo da função – declaração
int somar(int a, int b);

int main()
{
        // variáveis locais
        int soma, valor_a, valor_b;

        printf("Entre com valor a: ");
        scanf("%i", &valor_a);
        printf("Entre com valor b: ");
        scanf("%i", &valor_b);
        // Chamada ou invocação da função
        soma = somar(valor_a, valor_b);
        printf("Soma: %i\n", soma);

        return 0;
}

// implementação da função
int somar(int a, int b) {
        // retorna a soma dos valores
        return a + b;
}
```

Declara a função ***int somar(int a, int b)*** que além de possuir os parâmetros inteiros "***a***" e "***b***" agora também retorna um valor ***inteiro***;

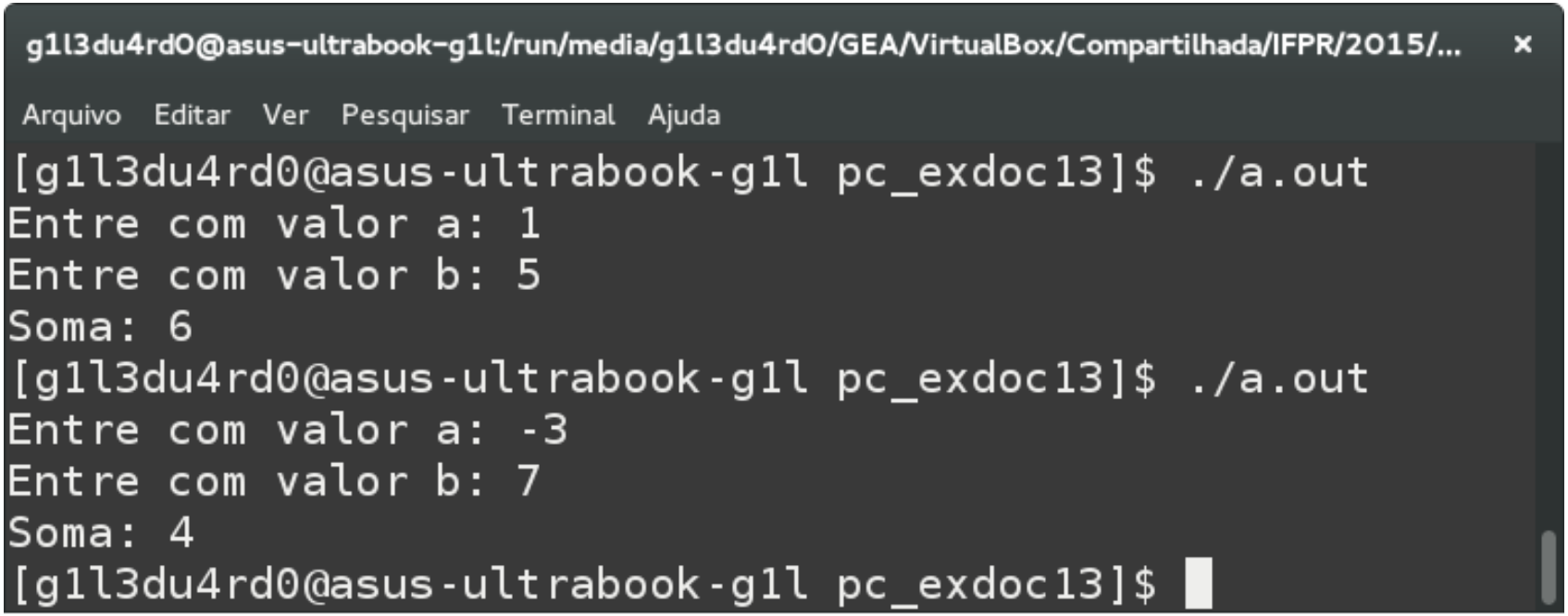
Declara as variáveis locais "***valor_a***" e "***valor_b***" que serão passadas a função ***somar()***, já que não podem ser acessadas pela mesma;

Invoca a função ***somar()*** passando os valores armazenados em "***valor_a***" e "***valor_b***" e armazena o valor retornado por ela dentro da variável local ***soma***;

Implementação da função ***somar()***, ela recebe os valores "***a***" e "***b***" utilizados na execução do código. Ela retorna a soma dos valores armazenados em "a" e "b". Toda vez que o operador ***return*** é chamado a execução da função é finalizada mesmo que ainda existam outras linhas de código abaixo dela.

# Função

## Exemplo de implementação (com retorno e com parâmetro )



# Operadores de Condição

## Exemplos  Utilizados no Documento

http://www.gileduardo.com.br/ifpr/pcii/downloads/pc_exdoc13.zip

## Mais Exemplos sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/pcii/downloads/pc_ex13.zip

## Exercícios sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/pcii/downloads/pc_pratica13.pdf