USP-ICMC-BInfo

Funções em C

SCC501 - ICC-II

2011

Prof. João Luís

Operadores & e *

```
#include <stdio.h>

main()
{
   int x = 15;
   printf("Endereco da variavel x = %d.\n", &x);
   printf("Conteudo do endereco da variavel x =
   %d.\n", *(&x));
}
```

Saída:

```
Endereco da variavel x = 1245052.
Conteudo do endereco da variavel x = 15.
```

Ponteiros e const

```
int gorp = 16;
int chips = 12;
const int * p_snack = &gorp;

*p_snack = 20;    // NÃO
```

não permite a mudança de valor para o qual *p_snack* aponta

```
p_snack = &chips; // OK
```

p_snack pode apontar para uma outra variável

```
int gorp = 16;
int chips = 12;
int * const p_snack = &gorp;

*p_snack = 20;    // OK
```

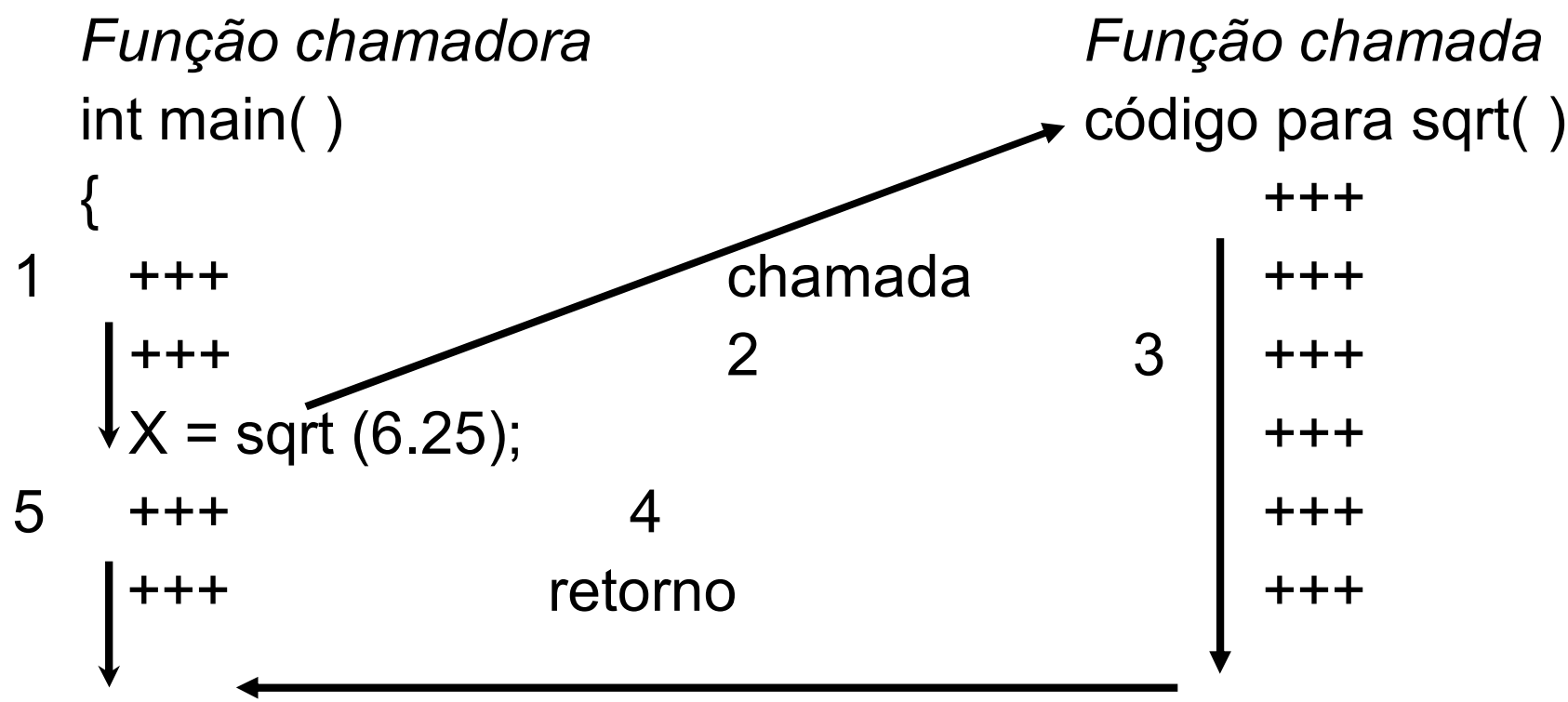
p_snack pode ser usado para mudar valor

```
p_snack = &chips; // NÃO
```

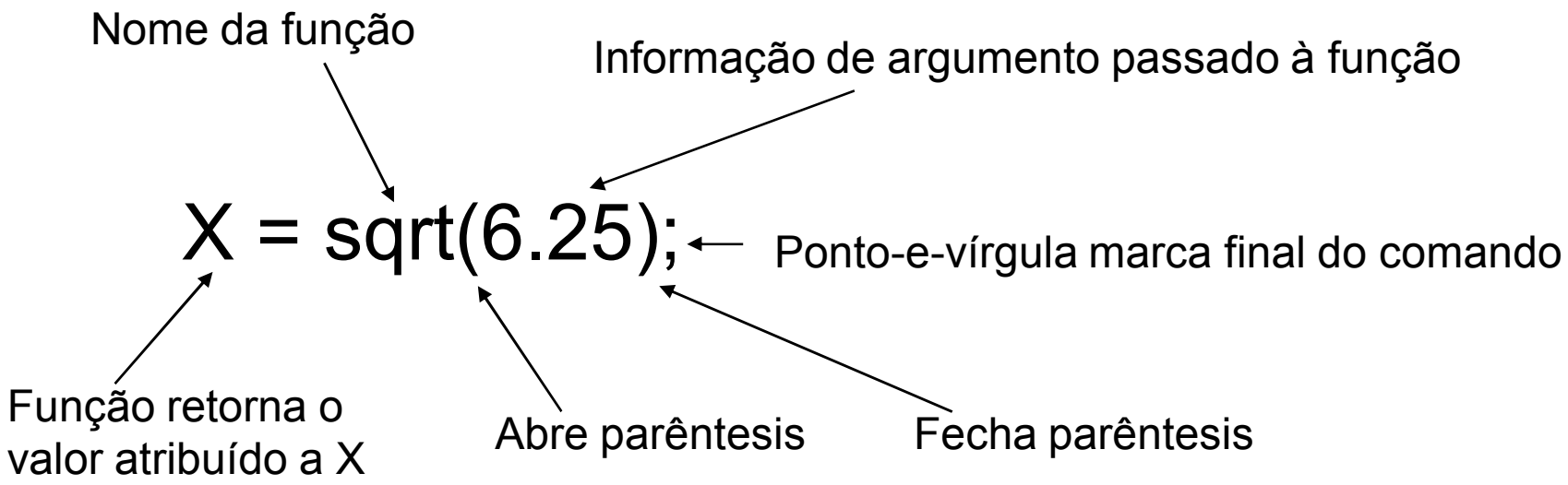
não permite a mudança de variável para o qual *p_snack* aponta

Usando uma função com um valor de retorno

Chamando uma função:



Sintaxe da chamada da função:



```
#include <stdio.h>
#include <math.h>
main()
{
   float cobertura; // usa float para números reais
   float lado;
   printf("Quantos centimetros quadrados de folhas
   voce tem?\n");
   scanf("%g", &cobertura);
   lado = (float) sqrt(cobertura); // chama a função,
                    // atribuindo valor de retorno
   printf("Voce pode cobrir um quadrado com lados de
   %g", lado);
   printf(" centimetros\ncom suas folhas.\n");
}
```

Saída:

```
Quantos centímetros quadrados de folhas você tem?
123.21
Você pode cobrir um quadrado com lados de 11.1
   centímetros
com suas folhas.
```

Funções definidas pelo usuário

```
#include <stdio.h>
void simples(); // protótipo da função

main()
{
   printf("main() chamara a funcao simples():\n");
   simples(); // chamada da função
}

// definição da função
void simples()
{
   printf("Eu sou apenas uma funcao simples.\n");
}
```

```
#include <stdio.h>
void simon(int); // protótipo de função para simon()
void main()
{
   int cont;
   simon(3); // chama a função simon()
   printf("Pegue um inteiro: ");
   scanf("%d", &cont);
   simon(cont); // chama-a de novo
}
void simon(int n) // define a função simon()
{
   printf("Simon diz: toque seus dedos do pe %d
   vezes.\n", n);
}
// funções void não precisam de comando de retorno
```

Saída:

```
Simon diz: toque seus dedos do pé 3 vezes.
Pegue um inteiro: 512
Simon diz: toque seus dedos do pé 512 vezes.
```

protótipos de função

função # 1

função # 2

função # 3

```
#include <iostream.h>
void simon(int);
double taxas(double);
void main( )
{
...
}
void simon(int n)
{
...
}
double taxas (double t)
{
...
return (2 * t);
}
```

Definições de funções ocorrem seqüencialmente em um arquivo

Função definida pelo usuário com um valor de retorno

```
#include <stdio.h>
int kilopgrama(int); // protótipo da função
main()
{
   int kilo, grama;
   printf("Entre com o peso em Quilogramas: ");
   scanf("%d", &kilo);
   grama = kilopgrama(kilo);
   printf("%d Quilogramas sao ", kilo);
   printf("%d gramas.\n", grama);
}
int kilopgrama(int sts)
{
   return 1000 * sts;
}
```

Protótipo de Função e chamadas de função

```
#include <stdio.h>
void saude(int); // protótipo: não retorna valor
float cubo(float x); // protótipo: returna um double
void main(void)
{
   float lado, volume;
   saude(5);              // chamada da função
   printf("De-me um numero: ");
   scanf("%g", &lado);
   volume = cubo(lado); // chamada da função
   printf("Um cubo de %g cm tem um volume de ",
   lado);
   printf("%g cm cubicos.\n", volume);
   saude((int) cubo(2));
}
```

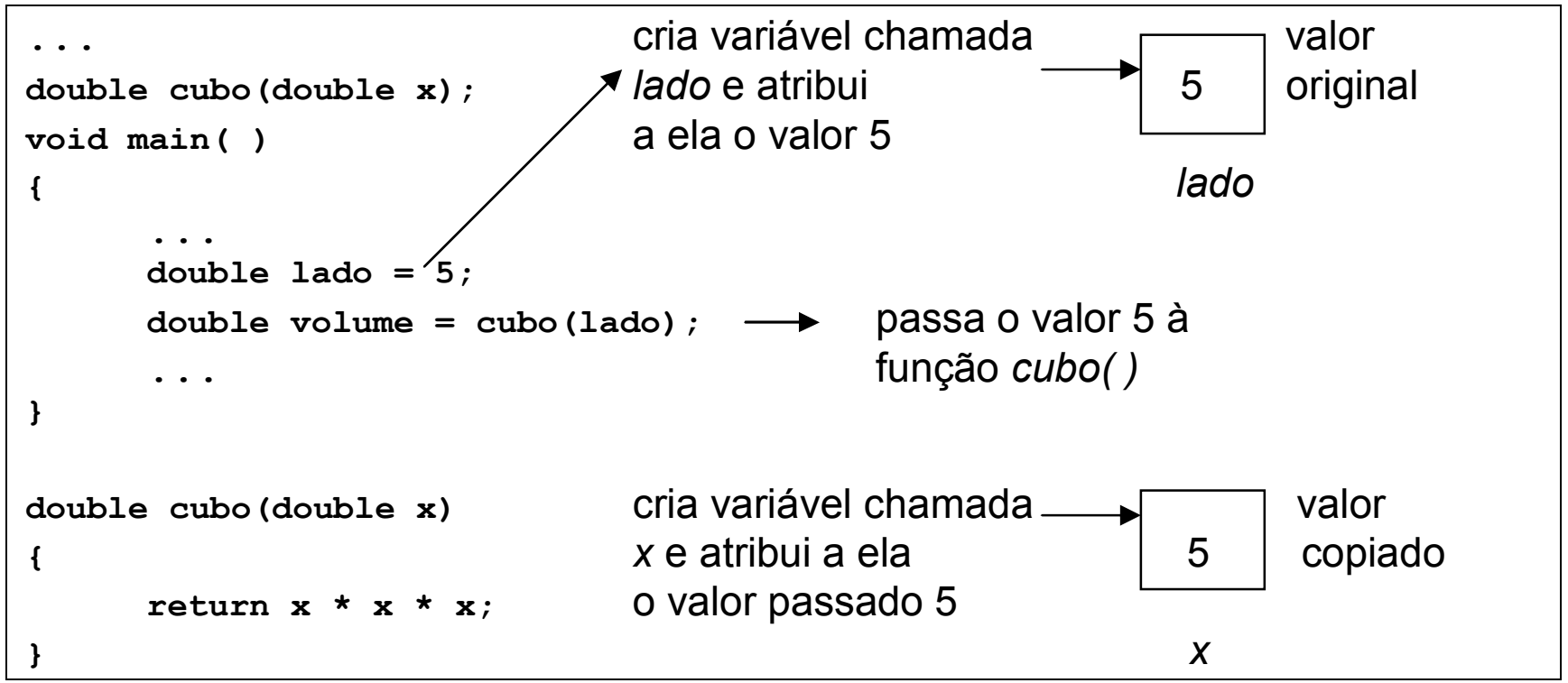
```
void saude(int n)
{
   int i;
   for (i = 0; i < n; i++)
       printf("Saude! ");
   printf("\n");
}
float cubo(float x)
{
   return x * x * x;
}
```

Saída:

```
Saúde! Saúde! Saúde! Saúde! Saúde!
Dê-me um número: 5
Um cubo de 5 cm tem um volume de 125 cm cúbicos.
Saúde! Saúde! Saúde! Saúde! Saúde! Saúde! Saúde!
   Saúde!
```

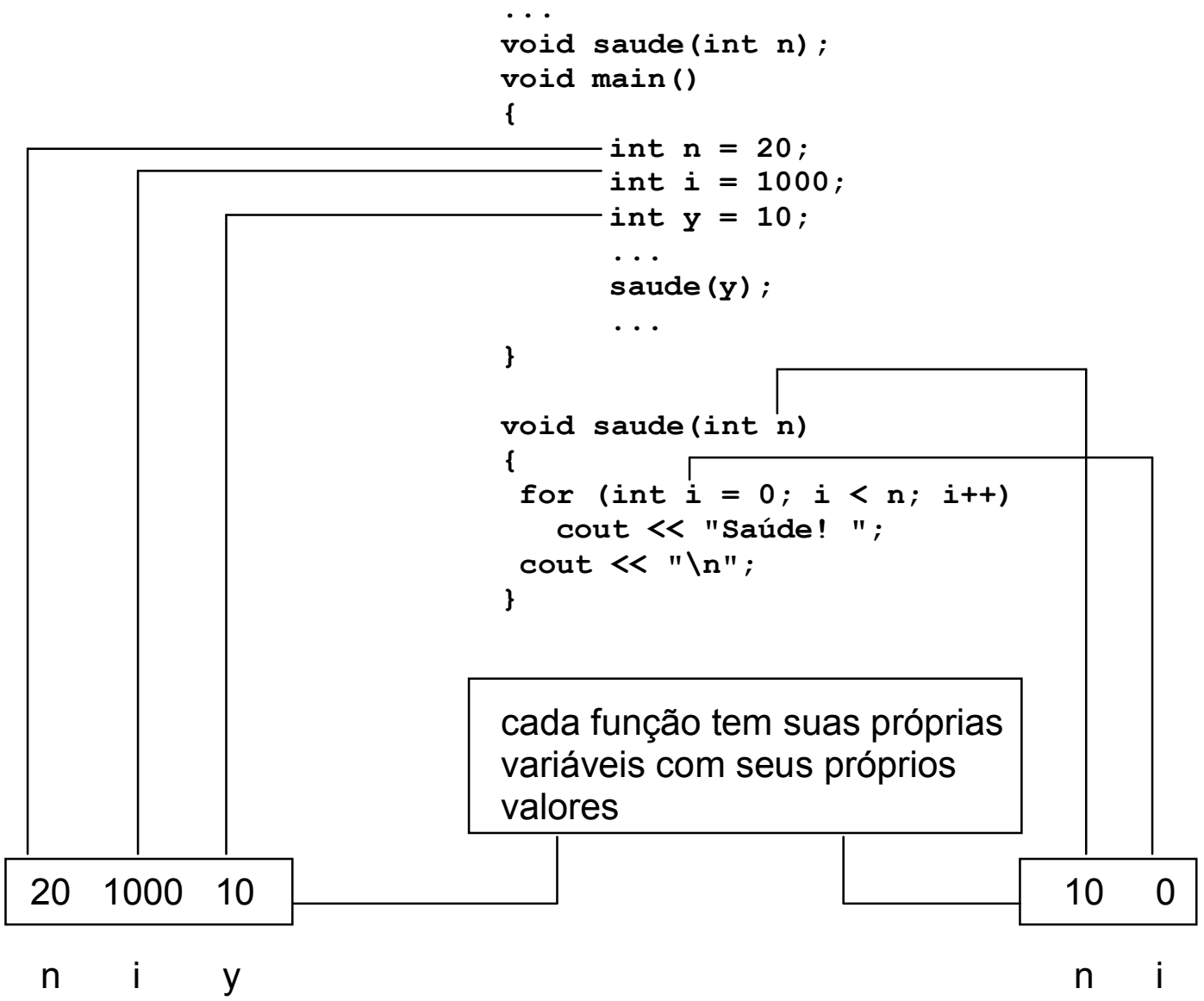
Argumentos de Função e Passagem por Valor

Argumentos de Função e Passagem por Valor



Passagem por valor

Variáveis Locais:



```
#include <stdio.h>
#include <conio.h>
void n_chars(char, int);
void main()
{
   int vezes = 0;
   char ch;
   printf("Entre com um caractere: ");
   ch = getche();
   while (ch != 's') // s para sair
   {
        printf("\nEntre com um inteiro: ");
        scanf("%d", &vezes);
        n_chars(ch, vezes); // função com dois argumentos
        printf("\nEntre com um outro caractere ou pressione a
   tecla s para sair: ");
        ch = getche();
   }
   printf("\nO valor de vezes eh %d.\n", vezes);
   printf("Tchau\n");
}
void n_chars(char c, int n) // mostra c n vezes
{
   while (n-- > 0) // continua até que n alcance 0
        printf("%c", c);
}
```

Argumentos Múltiplos

Saída:

```
Entre com um caractere: W
Entre com um inteiro: 50
WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
Entre com um outro caractere ou pressione a tecla s
   para sair: a
Entre com um inteiro: 20
aaaaaaaaaaaaaaaaaaaa
Entre com um outro caractere ou pressione a tecla s
   para sair: s
O valor de vezes é 20.
Tchau
```

```
#include <stdio.h>
float chance(unsigned numeros, unsigned pegadas);
void main()
{
  float total, escolhas;
  printf("Entre com o numero total de escolhas de
  cartoes de jogo e \n");
  printf("numero de escolhas pegadas permitida:\n");
  while (scanf("%f%f", &total, &escolhas) &&
  escolhas <= total)
  {
      printf("Voce tem uma chance em ");
      printf("%g", chance((int)total,
  (int)escolhas)); // computa a chance
      printf(" de ganhar.\n");
      printf("Proximos dois numeros (s para sair):
  ");
  }
  printf("tchau\n");
}
```

Uma Outra função de dois argumentos

```
// a seguinte função calcula a chance de pegar os
// números corretamente a partir de escolhas de
   números

float chance(unsigned numeros, unsigned pegadas)
{
   float result = 1.0; // algumas variáveis locais
   unsigned n;
   unsigned p;
   for (n = numeros, p = pegadas; p > 0; n--, p--)
       result = result * n / p ;
   return result;
}
```

Saída:

```
Entre com o número total de escolhas de cartões de
   jogo e
número de escolhas pegadas permitida:
49 6
Você tem uma chance em 1.39838e+07 de ganhar.
Próximos dois números (s para sair): 51 6
Você tem uma chance em 1.80095e+07 de ganhar.
Próximos dois números (s para sair): 38 6
Você tem uma chance em 2.76068e+06 de ganhar.
Próximos dois números (s para sair): s
tchau
```

Funções e Strings no Estilo C

```
#include <stdio.h>

int c_in_str(const char * str, char ch);

void main()
{
   char mmm[15] = "minimum"; // string em um vetor
   char *uivo = "ululado"; // uivo aponta para string
   int ms = c_in_str(mmm, 'm');
   int us = c_in_str(uivo, 'u');
   printf("%d caracteres m em %s\n", ms, mmm);
   printf("%d caracteres u em %s\n", us, uivo);
}
```

```
// esta função conta o número de caracteres ch
// na string str

int c_in_str(const char * str, char ch)
{
  int cont = 0;
  while (*str) // sai quando *str é '\0'
  {
      if (*str == ch)
            cont++;
      str++; // move ponteiro para próximo char
  }
  return cont;
}
```

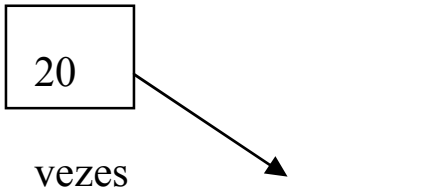
Saída:

```
3 caracteres m em minimum
2 caracteres u em ululado
```

passagem por valor

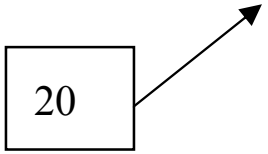
```
void espirro (int x);
void main()
{
      int vezes = 20;
      espirro (vezes);
      ...
}

void espirro (int n)
{
      ...
}
```

cria uma variável chamada vezes, atribui a ela o valor 20



duas variáveis, dois nomes

cria uma variável chamada x, atribui a ela o valor passado 20



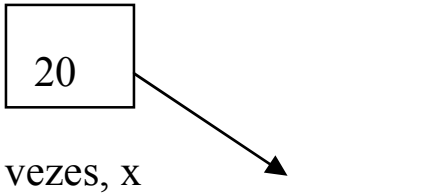
passagem por endereço

```
void ranzinza (int *);
void main()
{
      int vezes = 20;
      ranzinza (&vezes);
      ...
}

void ranzinza (int *x)
{
      ...
}
```

cria uma variável chamada vezes, atribui a ela o valor 20

Passa o endereço da variável *vezes*

faz x um alias para vezes

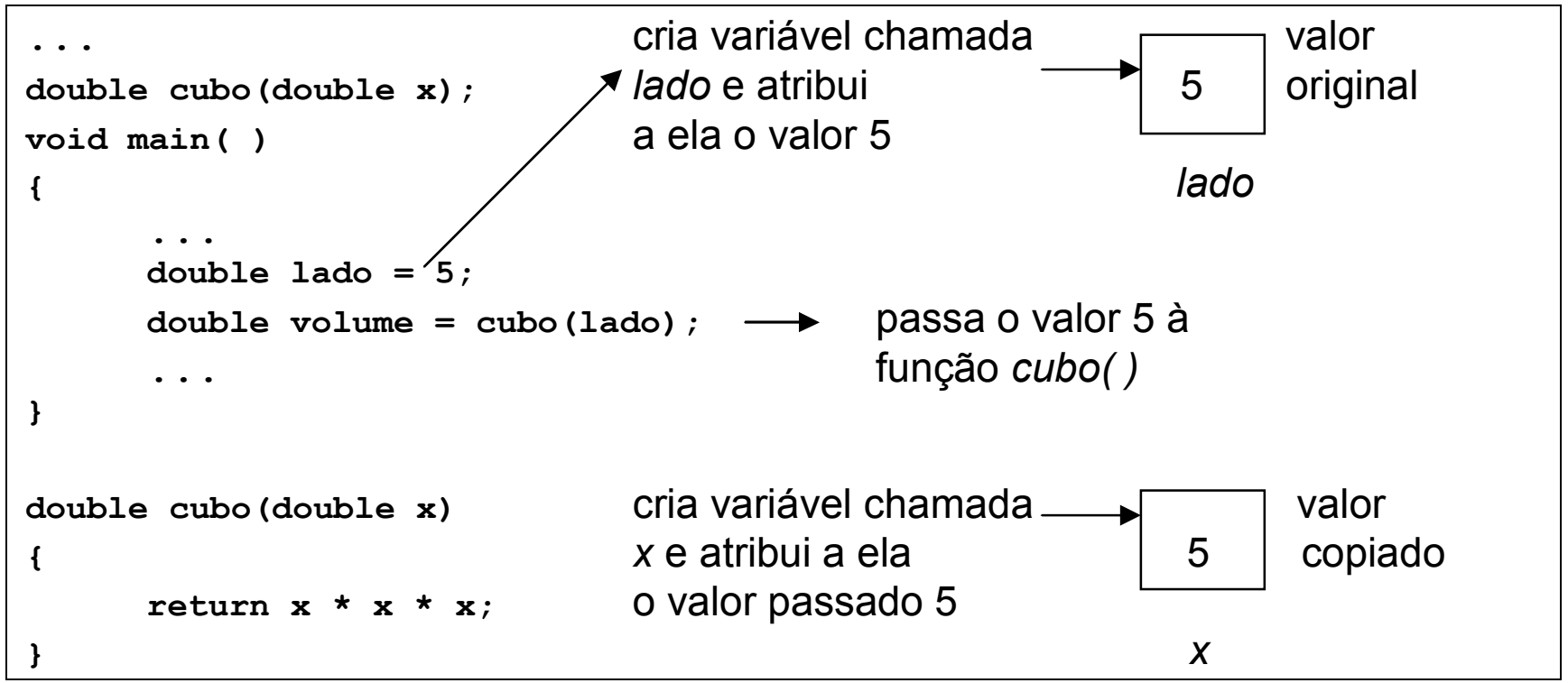


uma variável, dois nomes

```
#include <stdio.h>
altera(int *, int *);
main()
{
   int x,y;
   printf("Entre com o primeiro valor: ");
   scanf("%d", &x);
   printf("Entre com o segundo valor: ");
   scanf("%d", &y);
   altera(&x,&y);
}

altera(int *px,int *py)
{
   int soma, dif;
   soma = *px + *py;
   dif = *px - *py;
   *px = soma;
   *py = dif;
   printf("Soma = %d ou %d; Diferen%ca = %d ou %d\n", soma,
   *px, 135, dif, *py);
}
```

Argumentos de Função e Passagens por Valor e Endereço

Argumentos de Função e Passagem por Valor

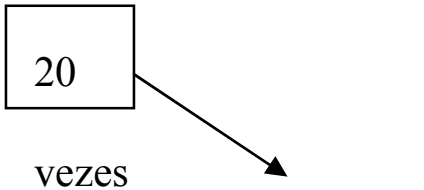


Passagem por valor

passagem por valor

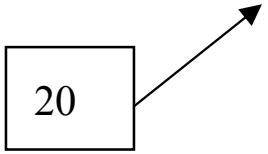
```
void espirro (int x);
void main()
{
      int vezes = 20;
      espirro (vezes);
      ...
}

void espirro (int n)
{
      ...
}
```

cria uma variável chamada vezes, atribui a ela o valor 20



duas variáveis, dois nomes

cria uma variável chamada x, atribui a ela o valor passado 20



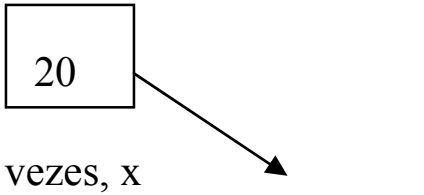
passagem por endereço

```
void ranzinza (int *);
void main()
{
      int vezes = 20;
      ranzinza (&vezes);
      ...
}

void ranzinza (int *x)
{
      ...
}
```

cria uma variável chamada vezes, atribui a ela o valor 20

Passa o endereço da variável *vezes*

faz x um alias para vezes



uma variável, dois nomes

```
#include <stdio.h>
altera(int *, int *);
main()
{
   int x,y;
   printf("Entre com o primeiro valor: ");
   scanf("%d", &x);
   printf("Entre com o segundo valor: ");
   scanf("%d", &y);
   altera(&x,&y);
}

altera(int *px,int *py)
{
   int soma, dif;
   soma = *px + *py;
   dif = *px - *py;
   *px = soma;
   *py = dif;
   printf("Soma = %d ou %d; Diferen%ca = %d ou %d\n", soma,
   *px, 135, dif, *py);
}
```

Lê um caractere por vez

```
#include <stdio.h>
main()
{
   char ch;
   int count = 0; // usa entrada básica
   scanf("%c", &ch); // pega um caractere
   while (ch != '#') // testa o caractere
   {
       if (ch != ' ')
       {
             printf("%c", ch); // ecoa o caractere
             count++; // conta o caractere
       }
       scanf ("%c", &ch); // pega o próximo caractere
   }
   printf("\n%d caracteres lidos\n", count);
}
```

Lê a string inteira

```
#include <stdio.h>
main()
{
   char ch[50];
   int i = 0;
   int count = 0;  // usa entrada básica
   gets(ch);       // pega uma string
   while (ch[i] != '#' && ch[i] != '\0')   // testa
   {
       if (ch[i] != ' ')
       {
             printf("%c", ch[i]);// ecoa o caractere
             count++;    // conta o caractere ecoado
       }
       i++;                // próximo caractere
   }
   printf("\n%d caracteres ecoados\n", count);
}
```

Funções que retornam strings

```
#include <stdio.h>
char * conststr(char c, int n); // protótipo
char str[20];
main()
{
   int vezes;
   char ch;
   char * ps = str;
   printf("Entre com um caractere: ");
   scanf("%c", &ch);
   printf("Entre com um inteiro: ");
   scanf("%d", &vezes);
   ps = conststr(ch, vezes);
   printf("%s\n", ps);
   ps = conststr('+', 20); // reusa ponteiro
   printf("%d-FIM-%s\n", ps, ps);
}
```

```
// contrói string feita de n caracteres c
char * conststr(char c, int n)
{
  char * pstr = str;
  pstr[n] = '\0'; // termina string
  while (n-- > 0)
      pstr[n] = c; // preenche resto da string
  return pstr;
}
```

Saída:

```
Entre com um caractere: R
Entre com um inteiro: 20
RRRRRRRRRRRRRRRRRRRR
4347904-FIM-++++++++++++++++++++
```

EXERCÍCIOS PROPOSTOS

1. Escreva uma função que retorne a soma de dois inteiros.
2. O que há de errado com a definição desta função:

```
salame(num)
{
    int num, cont;
    for(cont = 1; cont <= num; num++)
          printf(" Ó salame mio!\n");
}
```

3. Escreva uma função **`max(n,m)`** que retorne o maior de dois valores.

4. Use *loops* aninhados para escrever um programa que produza este padrão:

```
$$$$$$$$
$$$$$$$$
$$$$$$$$
$$$$$$$$
```

5. Que saída produzirá o seguinte *loop*?

```
for ( valor = 36; valor > 0; valor /= 2)
    printf("%3d", valor);
```

Respostas

```
// EP01
#include <stdio.h>
int soma(int, int);

main()
{
   int x,y;

   printf("Entre com o primeiro inteiro: ");
   scanf("%d", &x);
   printf("Entre com o segundo inteiro: ");
   scanf("%d", &y);
   printf("A soma de %d e %d %c %d\n", x, y, 130, soma(x,y));
}

int soma(int a, int b)
{
   return a + b;
}
```

```
// EP02
#include <stdio.h>

salame(int);

main()
{
   salame(5);
}

salame(int num)
{
   int cont;
   for(cont = 1; cont <= num; cont++)
       printf(" %c salame mio!\n", 224);
}
```

```
// EP03
#include <stdio.h>
int max(int, int);

main()
{
   int x,y;

   printf("Entre com o primeiro inteiro: ");
   scanf("%d", &x);
   printf("Entre com o segundo inteiro: ");
   scanf("%d", &y);
   printf("O maior entre %d e %d %c %d\n", x, y, 130, max(x,y));
}

int max(int a, int b)
{
   return a > b ? a : b; // retorna  a se a > b, senão retorna b
}
```

```
// EP04
#include <stdio.h>

main()
{
   int i,j;

   for (i = 1; i <= 4; i++)
   {
       for (j = 1; j <= 8; j++)
              printf("$");
       printf("\n");
   }
}
```

```
// EP05:

 36 18  9  4  2  1
```

```
#include <stdio.h>
alter(int *, int *);
main()
{
   int x,y;
   printf("Entre com o primeiro valor: ");
   scanf("%d", &x);
   printf("Entre com o segundo valor: ");
   scanf("%d", &y);
   alter(&x,&y);
   printf("\nNovos valores: x = %d e y = %d\n", x, y);
}
alter(int *px,int *py)
{
   int soma, dif;
   soma = *px + *py;
   dif = *px - *py;
   *px = soma;
   *py = dif;
   printf("Soma = %d ou %d; Diferen%ca = %d ou %d\n", soma, *px, 135, dif,
   *py);
}
```

Saída:

```
Entre com o primeiro valor: 3
Entre com o segundo valor: 7
Soma = 10 ou 10; Diferença = -4 ou -4

Novos valores: x = 10 e y = -4
```

```
#include <stdio.h>
main()
{
   char ch[50];
   char * c = ch;      // string c recebe espaço da string ch
   int i = 0;
   int count = 0;      // usa entrada básica
   gets(c);            // pega uma string
   while (*c != '#' && *c) // testa o caractere
   {
        if (*c != ' ')
        {
               printf("%c", *c);      // ecoa o caractere
               count++;               // conta o caractere ecoado
        }
        c++;                          // próximo endereço
   }
   printf("\n%d caracteres ecoados\n", count);
}
```

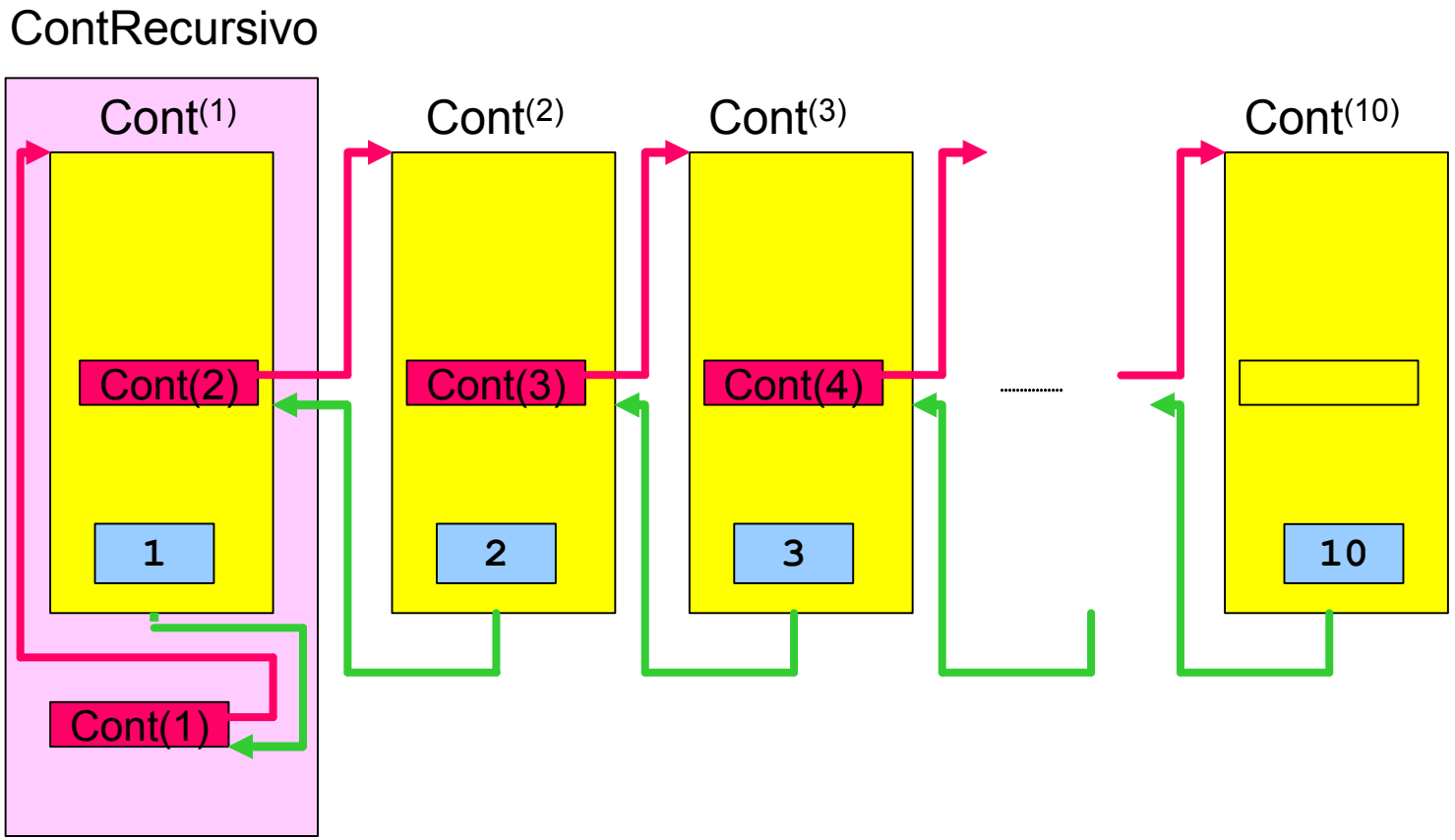
Recursão

- A recursão ocorre quando uma função chama a si própria. Quando isto acontece, várias ações ocorrem:
  - A função começa a execução do seu primeiro comando cada vez que é chamada;
  - Novas e distintas cópias dos parâmetros passados por valor e variáveis locais são criadas;
  - A posição que chama a função é colocada em estado de espera, enquanto que o nível gerado recursivamente esteja executando.

  – O próximo programa explica estes efeitos.

Contador recursivo

```
#include <stdio.h>

void cont (int);

main()
{
   cont(1);
}

void cont(int n)
{
   if (n < 10)
        cont(n+1);
   printf("%d\n", n);
}
```


ContRecursivo
Cont(1)
Cont(2)
Cont(3)
Cont(10)
Cont(2)
Cont(3)
Cont(4)
1
2
3
10
Cont(1)

Fatorial

```
#include <stdio.h>

unsigned long fat(int);

main()
{
   short N;

   printf("Calculo do fatorial\n\n");
   printf("Entre com um numero inteiro: ");
   scanf("%d", &N);

   printf("\nO fatorial de %d %c %d\n\n", N, 130, fat(N));
}

unsigned long fat(int n)
{
   int fato = 1;
   if (n > 1)
        fato = n * fat(n-1);
   return fato;
}
```

```
#include <stdio.h>

void contreg(int n);

main()
{
    contreg(4);            // chama a função recursiva
}

void contreg(int n)
{
    printf("Contagem regressiva ... %d\n", n);
    if (n > 0)
          contreg(n-1);     // função chama a si mesmo
    printf("%d: Fogo!\n", n);
}
```

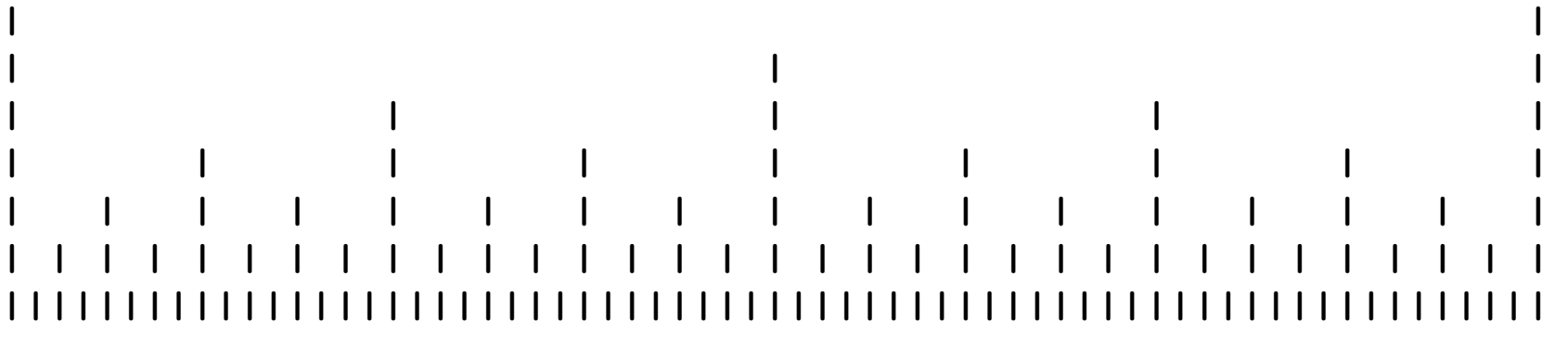
Saída:

```
Contagem regressiva ... 4
Contagem regressiva ... 3
Contagem regressiva ... 2
Contagem regressiva ... 1
Contagem regressiva ... 0
0: Fogo!
1: Fogo!
2: Fogo!
3: Fogo!
4: Fogo!
```

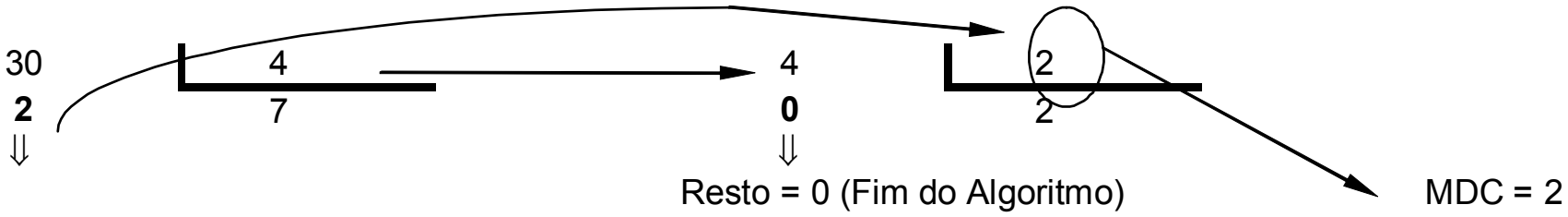
```
#include <stdio.h>
#define Comp 66
const int Divs = 6;

void subdivide(char ve[], int baixo, int alto, int nivel);

main()
{
    char regua[Comp];
    int i, j;
    int max = Comp - 2;
    int min = 0;
    for (i = 1; i < Comp - 2; i++)
          regua[i] = ' ';
    regua[Comp - 1] = '\0';
    regua[min] = regua[max] = '|';
    printf("%s\n", regua);
    for (i = 1; i <= Divs; i++)
    {
          subdivide(regua,min,max, i);
          printf("%s\n", regua);
          for (j = 1; j < Comp - 2; j++)
                      regua[j] = ' ';  // limpa a régua (branco)
    }

}
```

```
void subdivide(char ve[], int baixo, int alto, int nivel)
{
    int meio = (alto + baixo) / 2;
    if (nivel == 0)
          return;
    ve[meio] = '|';
    subdivide(ve, baixo, meio, nivel - 1);
    subdivide(ve, meio, alto, nivel - 1);
}
```

Saída:



PROGRAMA PROPOSTO. Escreva uma função recursiva que calcule o MDC (máximo divisor comum) de 2 números *a* e *b* recebidos como parâmetros. Para o cálculo do MDC, deve-se usar o Algoritmo de Euclides: Ex: a = 30 e b = 4:



Resto != 0: continua

```c
#include <stdio.h>

int MDC(int, int);

main()
{
   int a,b;
   printf("Calculo do MDC\n");
   printf("Entre com o valor de a: ");
   scanf("%d", &a);
   printf("Entre com o valor de b: ");
   scanf("%d", &b);
   printf("O MDC entre %d e %d %c %d\n", a, b, 130, MDC(a,b));
}


int MDC (int i, int j)
{
   if (i % j != 0)
        return MDC(j, i % j);
   else
        return j;
}
```