

Modularização

Roberto Rocha



Dividir para conquistar!

Dividir para conquistar!



Módulos

O objetivo é pegar um grande problema e dividi-lo em pequenas partes reutilizáveis.



Teamwork
vector illustration
www.shutterstock.com

Ao resolver as partes mínimas de um problema, automaticamente obtém-se a solução do todo.
Essa estratégia de trabalho é baseada na ideia de dividir para conquistar.

Módulos

Um profissional deve saber:

A diferença entre fazer um programa certo e fazê-lo funcionar.

O mercado necessita de profissionais que saibam fazer o certo,

Que saibam de fato programar e utilizem técnicas de trabalho reconhecidas e validadas internacionalmente

Não é possível, em hipótese nenhuma, produzir sistemas de informação e outros tipos de programa com qualidade pelas mãos de programadores que fazem simplesmente os programas funcionarem.



Chega de "gambiarra".



Módulos

Problemas complexos → algoritmos complexos

possível dividir um problema grande em problemas menores



usar o processo de modularidade

Cada parte menor ou modulo tem um algoritmo mais simples, o que facilita chegar à grande solução

Módulos

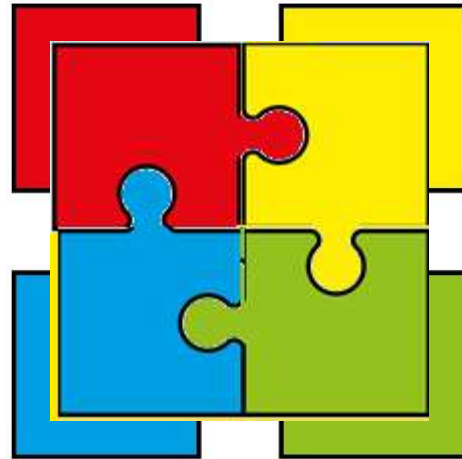
Modulo é um bloco de programa que pode efetuar operações computacionais de

entrada

processamento

saída

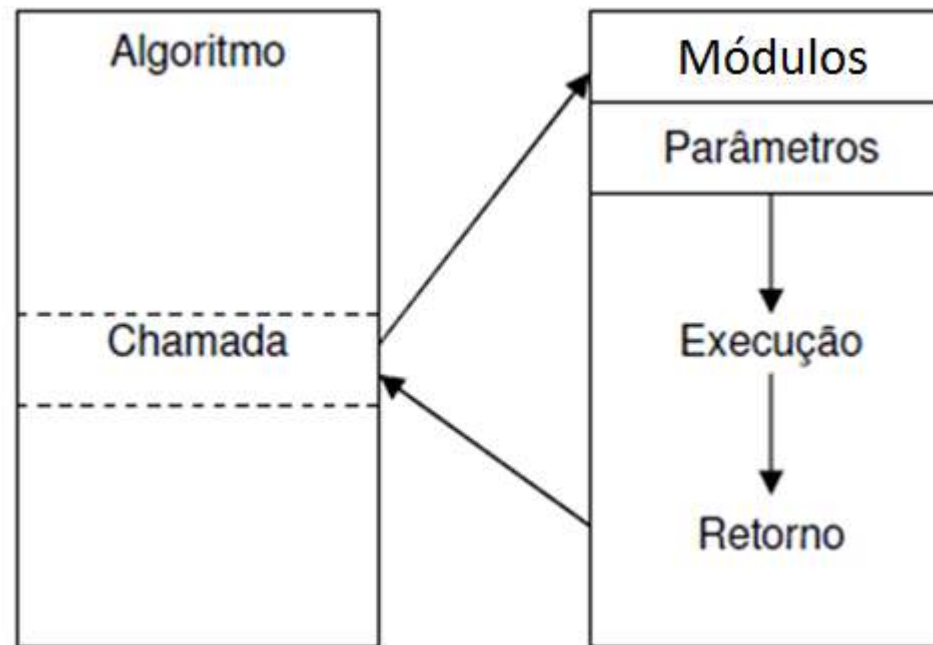
Ao dividir um problema complexo em módulos, ideia de abstração.



Abstrair significa considerar isoladamente um ou mais elementos de seu todo; separar o todo em partes.

Módulos

Os módulos, também chamados de subprogramas são chamados dentro do corpo do programa principal como se fossem *comandos*.



Após seu término, a execução continua a partir do ponto onde foi chamado.

É importante compreender que a chamada de um subprograma simplesmente gera um **desvio provisório no fluxo de execução**

Módulos - Procedimentos

Suponha o seguinte algoritmo:

```
inicio
  comando a
  comando b
  comando c
  escreva ("fim dos comandos a,b,c")
  comando a
  comando b
  comando c
  escreva ("fim dos comandos a,b,c")
  comando a
  comando b
  comando c
  escreva ("fim dos comandos a,b,c")
fim
```

Poderá ser escrito da seguinte forma :

```
procedimento comandosABC()
  inicio
    comando a
    comando b
    comando c
  fim
```

```
inicio
  comandosABC()
  escreva ("fim dos comandos a,b,c")
  comandosABC()
  escreva ("fim dos comandos a,b,c")
  comandosABC()
  escreva ("fim dos comandos a,b,c")
  comandosABC()
  escreva ("fim dos comandos a,b,c")
fim
```

Metodologia Top-Down e Bottom-Up

A metodologia top-down (de cima para baixo) e bottom-up (de baixo para cima) podem ser utilizados para facilitar a construção de programas de computador.

Top-down descreve de forma resumida as ações sem preocupar-se com detalhes

Bottom-up descreve de forma detalhada as operações mínimas de um programa

O projeto do programa em si, pode ser feito com base na metodologia top-down

O código do programa pode ser feito com o método bottom-up.

Metodologia Top-Down e Bottom-Up

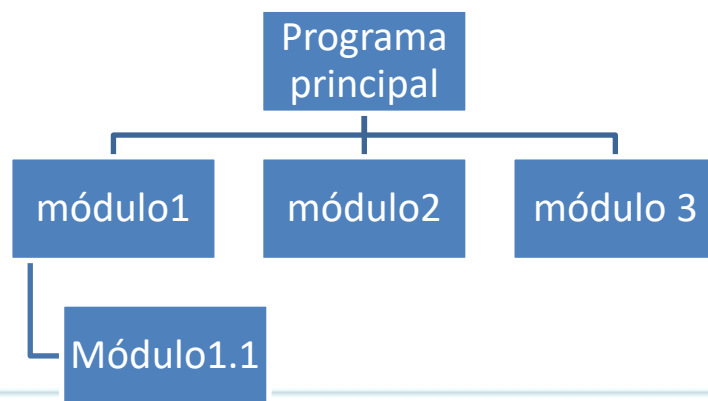
Top-down e bottom-up facilitam a aplicação das etapas da programação estruturada ou mesmo da programação orientada a objetos.

O método top-down caracteriza-se basicamente por:

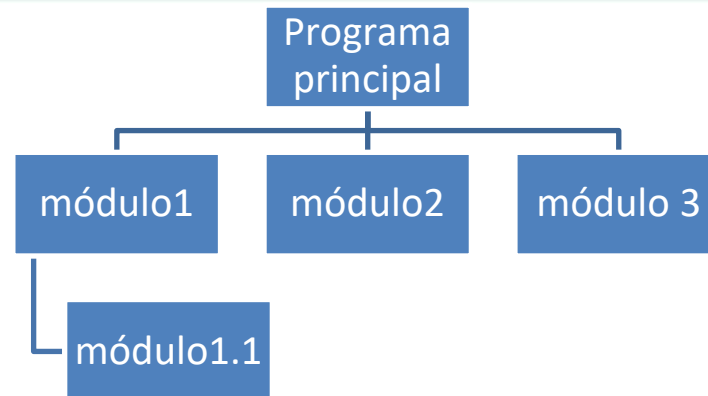
Antes de iniciar a construção de um programa de computador, o programador deve ter em mente as tarefas principais que ele deve executar.

Não é necessário saber como funcionarão, somente quantas são.

O uso do método top-down faz com que a estrutura do programa seja semelhante a um organograma.



Metodologia Top-Down e Bottom-Up



Conhecidas todas as tarefas a serem executadas, deve-se ter em mente como deve ser o programa principal, que vai controlar todas as outras tarefas distribuídas nos módulos (sub-rotinas).

Definido o programa principal, é iniciado o processo de detalhamento estrutural para cada sub-rotina.

São definidos vários algoritmos, um para cada rotina em separado, para que se tenha uma visão do que deve ser executado em cada modulo de programa.

Pode-se inclusive estabelecer o numero máximo de linhas de programa que uma rotina deve possuir.

Se o numero de linhas ultrapassa o limite preestabelecido, a rotina em desenvolvimento é dividida em outra sub-rotina (é nesse ponto que se aplica o método de refinamento sucessivo).

Procedimentos

Um modulo de procedimento (sub-rotina) é um bloco de programa com inicio e fim, identificado por um nome que referencia seu uso em qualquer parte do programa principal ou do programa chamador da sub-rotina.

procedimento <nomeprocedimento> ([var] <parâmetros>)

var

<declaração das variáveis locais ao procedimento>

inicio

<lista de comandos>

fimprocedimento

Procedimentos

Algoritmo “soma”

var

a,b,soma :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

escreva(“A soma de “,a,” e “,b, “ = “,soma)

fimalgoritmo

procedimento somaValores()

var

a,b,soma :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

escreva(“A soma de “,a,” e “,b, “ = “,soma)

fimprocedimento

Algoritmo “soma”

var

inicio

somaValores()

fimalgoritmo

Algoritmo x C

Algoritmo	C
Procedimento	
procedimento <nomeprocedimento> ([var] <parâmetros>) var <declaração das variáveis locais ao procedimento> inicio <lista de comandos> fimprocedimento	void <nomeprocedimento>(<paramêtros>) { <declaração de variáveis> <lista de comandos> }
Exemplos	
procedimento somaValores(); var a,b,soma :inteiro inicio leia(a,b) soma ← a + b escreva("A soma de “a,” e “b, “ = “,soma) fimprocedimento	void somaValores() { int a,b,soma; printf("Digite o valor para o primeiro valor:"); scanf("%d",&a); printf("Digite o valor para o segundo valor:"); scanf("%d",&b); soma =a + b; printf("A soma de %d e %d = %d\n",a,b,soma); }

Procedimentos

Programa para somar 2 valores

Algoritmo “soma”

var

a,b,soma :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

escreva(“A soma de “,a,” e “,b, “ = “,soma)

fimalgoritmo

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <locale.h>
4
5  int main()
6  {
7      system("mode con:cols=100 ");
8      setlocale(LC_ALL, "portuguese");
9      int a,b,soma;
10     printf("Digite o valor de a:");
11     scanf("%i",&a);
12     printf("Digite o valor de b:");
13     scanf("%i",&b);
14     soma = a+b;
15     printf("A soma de %i e %i = %i\n",a,b,soma);
16     return 0;
17 }
```

procedimento somaValores()

var

a,b,soma :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

escreva(“A soma de “,a,” e “,b, “ = “,soma)

fimprocedimento

Algoritmo “soma”

var

inicio

somaValores()

fimalgoritmo

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <locale.h>
4  void somaValores();
5  int main()
6  {
7      system("mode con:cols=100 ");
8      setlocale(LC_ALL, "portuguese");
9      somaValores();
10     return 0;
11 }
12 void somaValores(){
13     int a,b,soma;
14     printf("Digite o valor de a:");
15     scanf("%i",&a);
16     printf("Digite o valor de b:");
17     scanf("%i",&b);
18     soma = a+b;
19     printf("A soma de %i e %i = %i\n",a,b,soma);
20 }
```


Procedimentos

Faça um programa para calcular a média entre 2 valores

Algoritmo “soma”

var

a,b,soma :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

escreva(“A soma de “,a,” e “,b, “ = “,soma)

fimalgoritmo

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <locale.h>
4
5  int main()
6  {
7      system("mode con:cols=100 ");
8      setlocale(LC_ALL, "portuguese");
9      int a,b,soma;
10     printf("Digite o valor de a:");
11     scanf("%i",&a);
12     printf("Digite o valor de b:");
13     scanf("%i",&b);
14     soma = a+b;
15     printf("A soma de %i e %i = %i\n",a,b,soma);
16     return 0;
17 }
```

procedimento somaValores()

var

a,b,soma :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

escreva(“A soma de “,a,” e “,b, “ = “,soma)

fimprocedimento

Algoritmo “soma”

var

inicio

somaValores()

fimalgoritmo

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <locale.h>
4  void somaValores();
5  int main()
6  {
7      system("mode con:cols=100 ");
8      setlocale(LC_ALL, "portuguese");
9      somaValores();
10     return 0;
11 }
12 void somaValores() {
13     int a,b,soma;
14     printf("Digite o valor de a:");
15     scanf("%i",&a);
16     printf("Digite o valor de b:");
17     scanf("%i",&b);
18     soma = a+b;
19     printf("A soma de %i e %i = %i\n",a,b,soma);
20 }
```

Procedimentos

Faça um programa para calcular a média entre 2 valores

Algoritmo “média”

var

a,b,soma,media :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

media \leftarrow soma/2

escreva(“A media de “,a,” e “,b, “ = “,media)

fimalgoritmo

procedimento media2Valores()

var

a,b,soma,media :inteiro

inicio

leia(a,b)

soma \leftarrow a + b

media \leftarrow soma/2

escreva(“A media de “,a,” e “,b, “ = “,media)

fimprocedimento

Algoritmo “media”

var

inicio

media2Valores()

fimalgoritmo

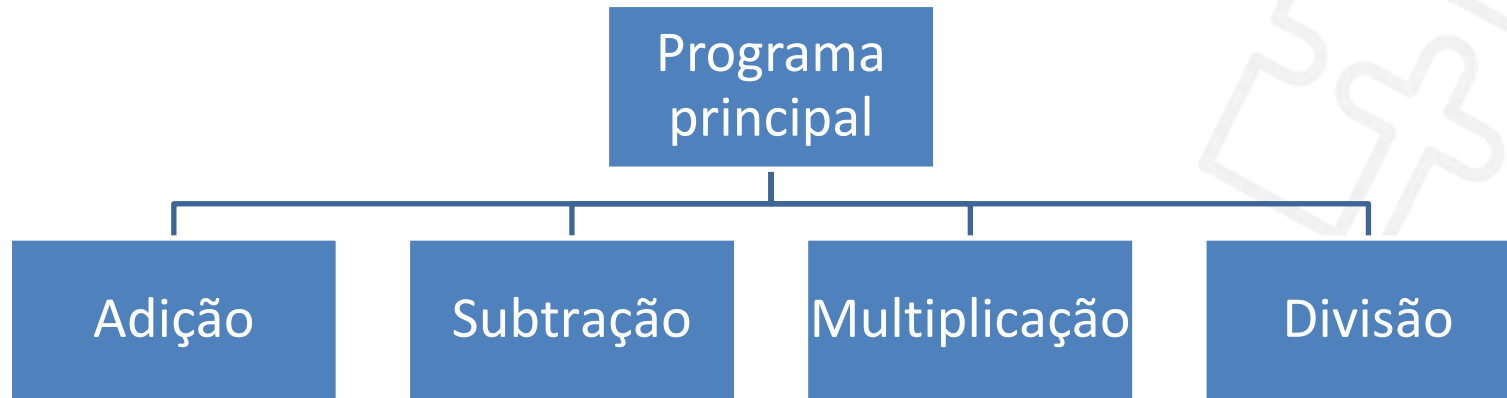
Procedimentos

Exercício de fixação:

Desenvolver um programa de computador que simule as operações básicas de uma calculadora que opere com a entrada de dois valores do tipo real após a escolha da operação a ser executada.

O programa deve apresentar uma lista de opções (menu) com as operações matemáticas de **adição**, **subtração**, **multiplicação** e **divisão**, além de uma opção de saída do programa. Escolhida a opção desejada, deve ser solicitada a entrada de dois valores numéricos para que seja possível executar o processamento escolhido. Após a execução da operação, o programa deve apresentar o resultado. Após a execução de qualquer uma das operações de calculo, o programa deve voltar para o menu de seleção.

Calculadora



Calculadora

Algoritmo "Calculadora"

var

op:inteiro

Início

faça

menu()

leia(op)

escolha (op)

1: somaValores()

2: subtraiValores()

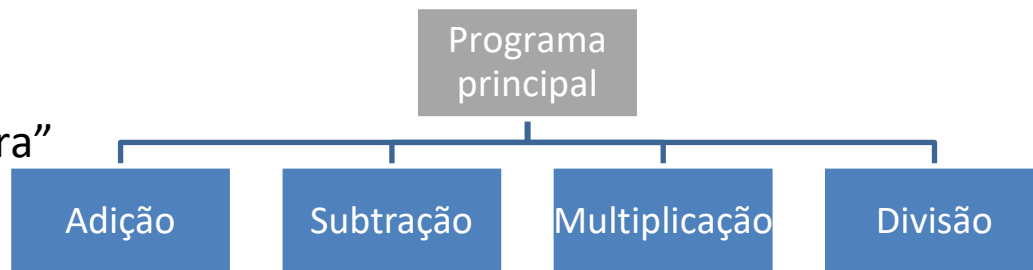
3: multiplicaValores()

4:divideValores()

fimescolha

enquanto (op<>9)

fimalgoritmo



procedimento menu()

var

início

escreva("menu:")

escreva("1 – adição")

escreva("2 – subtração")

escreva("3 – multiplicação")

escreva("4 – divisão")

escreva("9 – sair do programa")

fimprocedimento

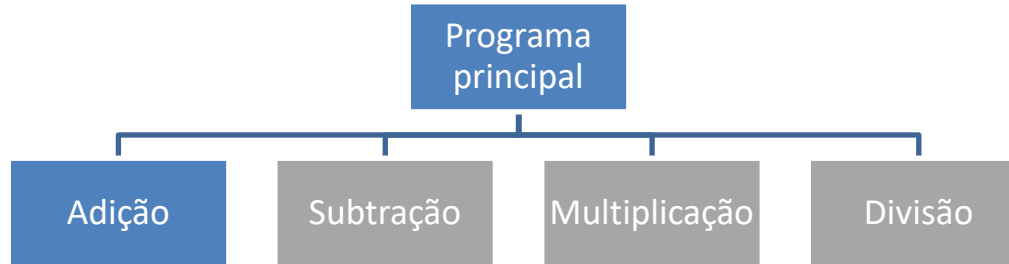
```

39 void menu()
40 {
41     system("CLS");
42     printf("menu:\n");
43     printf("1 ~ adição\n");
44     printf("2 ~ subtração\n");
45     printf("3 ~ multiplicação\n");
46     printf("4 ~ divisão\n");
47     printf("9 ~ sair do programa\n");
48 }
  
```

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <locale.h>
4  void menu();
5  void somaValores();
6  void subtraiValores();
7  void multiplicaValores();
8  void divideValores();
9  int main()
10 {
11     setlocale(LC_ALL, "portuguese");
12     int op;
13     do
14     {
15         menu();
16         printf("Escolha:");
17         scanf("%d", &op);
18         switch (op)
19         {
20             case 1:
21                 somaValores();
22                 break;
23             case 2:
24                 subtraiValores();
25                 break;
26             case 3:
27                 multiplicaValores();
28                 break;
29             case 4:
30                 divideValores();
31                 break;
32             }
33         if (op!=9) {
34             system("PAUSE");
35         }
36     } while (op!=9);
37     return 0;
38 }
  
```

Calculadora - adição

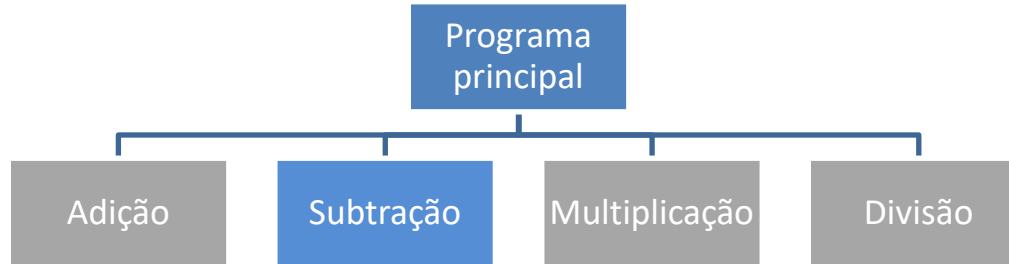


deve ser solicitada a entrada de dois valores numéricos para que seja possível executar o processamento escolhido e mostrar o resultado.

```
procedimento somaValores();  
var  
    a,b,soma :inteiro  
inicio  
    leia(a,b)  
    soma ← a + b  
    escreva("A soma de “a,” e “b, “ = “,soma)  
fimprocedimento
```

```
49 void somaValores()  
50 {  
51     int a,b,soma;  
52     printf("Digite o valor para o primeiro valor:");  
53     scanf("%d",&a);  
54     printf("Digite o valor para o segundo valor:");  
55     scanf("%d",&b);  
56     soma =a + b;  
57     printf("A soma de %d e %d = %d\n",a,b,soma);  
58 }
```

Calculadora - Subtração



deve ser solicitada a entrada de dois valores numéricos para que seja possível executar o processamento escolhido e mostrar o resultado.

```
procedimento subtraiValores();
```

```
var
```

```
  a,b,subtrai :inteiro
```

```
inicio
```

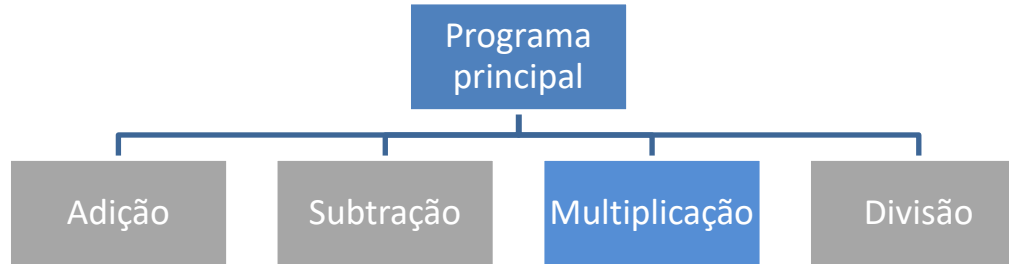
```
  leia(a,b)
```

```
  subtrai  $\leftarrow$  a - b
```

```
  escreva("A subtração de “,a,” menos “,b, “ = “,subtrai)
```

```
fimprocedimento
```

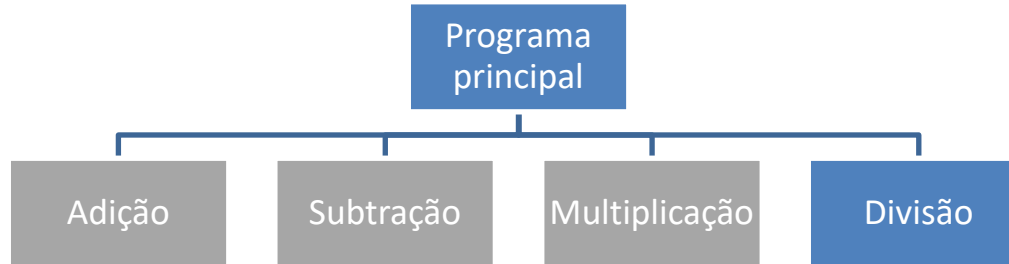
Calculadora - Multiplicação



deve ser solicitada a entrada de dois valores numéricos para que seja possível executar o processamento escolhido e mostrar o resultado.

```
procedimento multiplicaValores();  
var  
  a,b,multiplica :inteiro  
inicio  
  leia(a,b)  
  multiplica  $\leftarrow$  a * b  
  escreva("A multiplicação de “a,” e “b, “ = “,multiplicacao)  
fimprocedimento
```


Calculadora - Divisão



deve ser solicitada a entrada de dois valores numéricos para que seja possível executar o processamento escolhido e mostrar o resultado.

```
procedimento divideValores();  
  var  
    a,b :inteiro  
    divide:real  
  inicio  
    leia(a,b)  
    divide  $\leftarrow$  a / b  
    escreva("A divisão de “a,” por “b, “ = “,divide)  
  fimprocedimento
```



PUC Minas
Virtual