3
Revendo
Sub-Rotinas em C/C++

Prof. Dr. Eliseu LS

O que precisa saber antes ?

1) Bibliotecas: iostream, iomanip, math.h, cstdlib como declarar e para que servem.
2) Variáveis e constantes, como declarar e para que servem.
3) comandos cin, cout, system para que servem
4) Operadores relacionais, lógicos e o comandos IF e GOTO

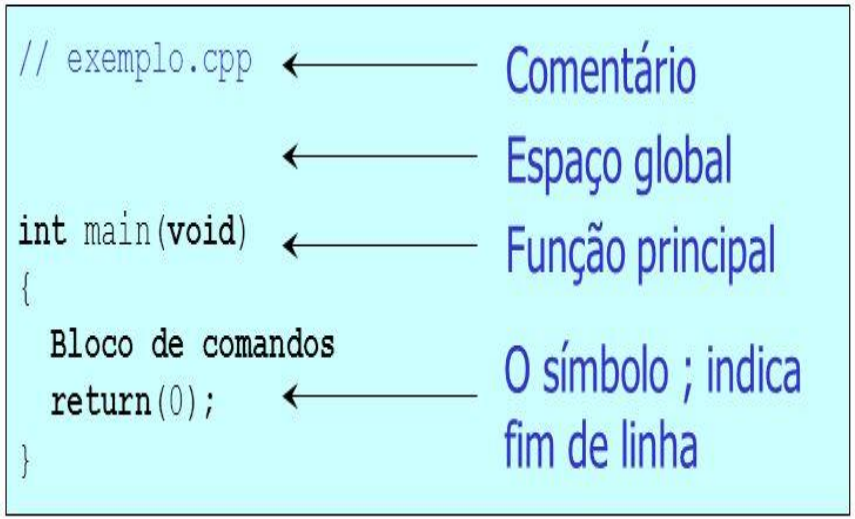
O que iremos estudar nesta aula ?

1) Declaração de variáveis globais e locais
2) Codificação de VOIDS
3) Codificação de Functions
4) Uso de parâmetros e argumentos
5) Quadro Resumo de Sub Rotinas/Declarações de Funções

Escopo e Variáveis Globais

- Um programa C++ é dividido em funções:
  - As definições/declarações no interior funções são **locais**.
  - As definições/declarações fora das funções são **globais**.

- Todo programa precisa ter uma função chamada **main**
  - Esta função é a primeira a ser chamada quando um programa é executado.
  - Sintaxe:



Todo código fonte para ser executado deve ter a função **main ( )**, entretanto você poderá criar suas próprias **sub rotinas** que podem ser functions ou voids, cada uma com um objetivo específico, para **leitura**, **cálculos** ou **exibição**.

As sub rotinas devem ser declaradas logo abaixo das diretivas do pré-processador e acima da função main ( ), exatamente no **Escopo global** onde também serão criadas as **constantes** e **variáveis globais,** que **permanecerão na memória durante toda a execução do programa.**

Porém o código das sub rotinas deve ficar abaixo da função main( ), isto é uma boa prática de programação.

Sub rotina do tipo Void e Variáveis Locais

O tipo void (vazio) pode ser considerado um pequeno programa dentro de um programa maior, é executado quando o seu nome é digitado em outra sub rotina qualquer, seu código será finalizado pelo comando **return;** **Variáveis locais** são criadas dentro sub rotinas, com escopo local, não podem ser acessadas fora das sub rotinas onde foram criadas. Ocuparão a memória somente durante o tempo de execução das sub rotinas onde foram criadas.

Simulação 1 - Tipo Void

```
double const pi = 3.14;   // constante
double    altura = 5.55; // variável global

void verVolume ( double raio ); // declaração

int main() {
double raio = 10.5;
verVolume ( raio  );     // chamada por
referência
verVolume ( 13.4 );    // chamada por valor
system("sleep 5"); }

// código do void
void verVolume ( double raio )
{ double volume = raio * raio * pi * altura;
 cout<< "\nVolume :" << volume  << endl;
return; }
```

Sub Rotina do tipo FUNCTION (não void)

Uma **função** (tipo int, double, string, char, float) têm o objetivo de produzir e armazenar o seu próprio valor através de seu código interno. Este único valor será armazenado através do comando **return** precedido de seu valor de retorno. Para executar/chamar uma function usa-se uma variável local do mesmo tipo da função dentro de uma subrotina qualquer.

Simulação 2 - Function (Não void)

```
int lern1();

int modulo (int numero);

int main ( )   {
  int num, modu;
  num     = lern1 ( );
  modu = modulo ( num );
  cout << modulo << endl; }

int lern1 ( ) { int n1;
cout << "Digite n1";
cin >> n1;  return n1;}

int modulo ( int numero )  {
  int modu;
  if ( numero < 0 ) modu = numero * -1;
  else modu = numero;
 return modu;   }
```

Simulação 3 - Sub-Rotinas

```
double const txmulta = 0.02;

// função de leitura

double lerValor ( )   {  double val;
                         cout << "Digite Prestação:";
                         cin >> val;  return val; }


// tipo void para mostrar o valor final
void mostrar (double val, double multa) {
 double valorFinal;
valorFinal= val + multa;
cout<<"\nValor a pagar:"<< valorFinal;;}

//função para calcular a multa

double calcMulta  ( double valorPrestacao )

{     return ( valorPrestacao * txmulta );  }
```

Simulação 3 - Main ( ) - Menu de Controle

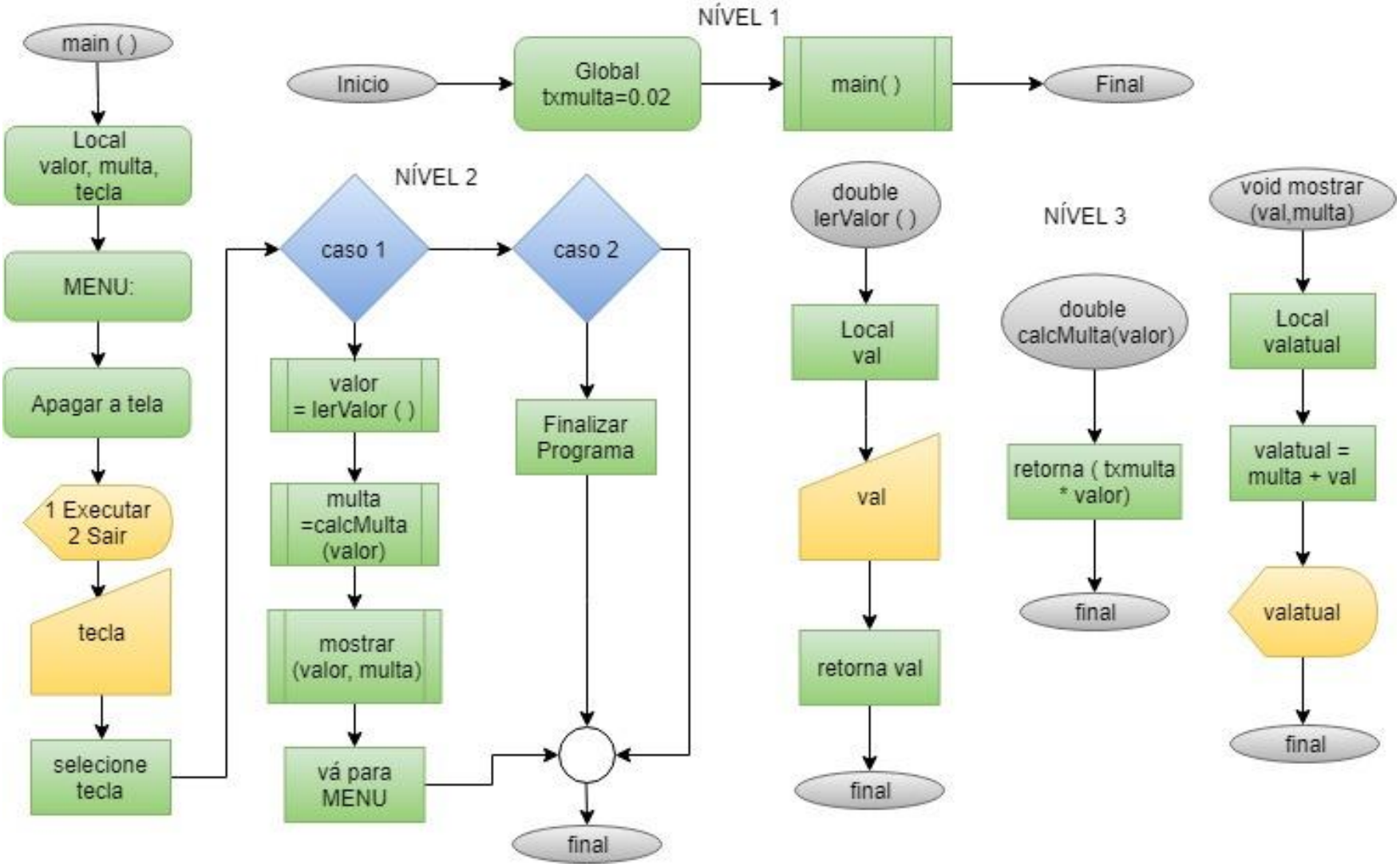
```
int main()  {
 double valorPrest, valorMulta; int tecla;

MENU:
system ("cls");
 cout << "menu\n1 Executar\n2 Sair\nitem:";
 cin >> tecla;
 switch(tecla)
  {
     case 1: valorPrest = lerValor();
             valorMulta = calcMulta(valorPrest);
             mostrar (valorPrest, valorMulta);
             break;

     case 2:  cout << "\nFim do programa!\n";
             system("pause"); exit(0);
             break;
   }
goto MENU;

 return 0;    }
```

Diagramas Simulação 3 Nível 1/2/3 (sub rotinas)



2 REGRAS PARA MEMORIZAR

1 – Parâmetros, são variáveis locais de ENTRADA de uma sub-rotina qualquer declaradas entre os parênteses.

2 – Entre os parâmetros de uma sub-rotina, não devem haver **constantes**, **variáveis globais** ou **variáveis internas de saída ou resultados** de fórmulas internas.

Recomendo: ***Assistam os vídeos Aula 3 Programação Estrutura em C++ partes 1 e 2***

Canal: Eliseu Lemes C++ (PlayList Aulas de C++)

TAREFA / AVALIAÇÃO CONTINUADA

1. Faça o programas h, i, k, l e m da página 26 do livro de exercícios, não necessita menu, somente crie as sub-rotinas e execute a partir da função int main(). Não se esqueça de declarar as sub-rotinas antes da função int main() e codificar as sub-rotinas após o int main().

2. Fazer o quadro resumo de subrotinas e código fonte de programa para ler o comprimento e calcular o diâmetro, calcular o raio e finalmente a área de um círculo; (Faça um menu com as opções necessárias. Utilize a simulação 3 como referência/Modelo para fazer este trabalho.)

NOTA: Entregar as atividades juntamente com os enunciados, códigos e prints de execução em um arquivo do tipo DOCX ou PDF, via Teams ou Google Home Class.