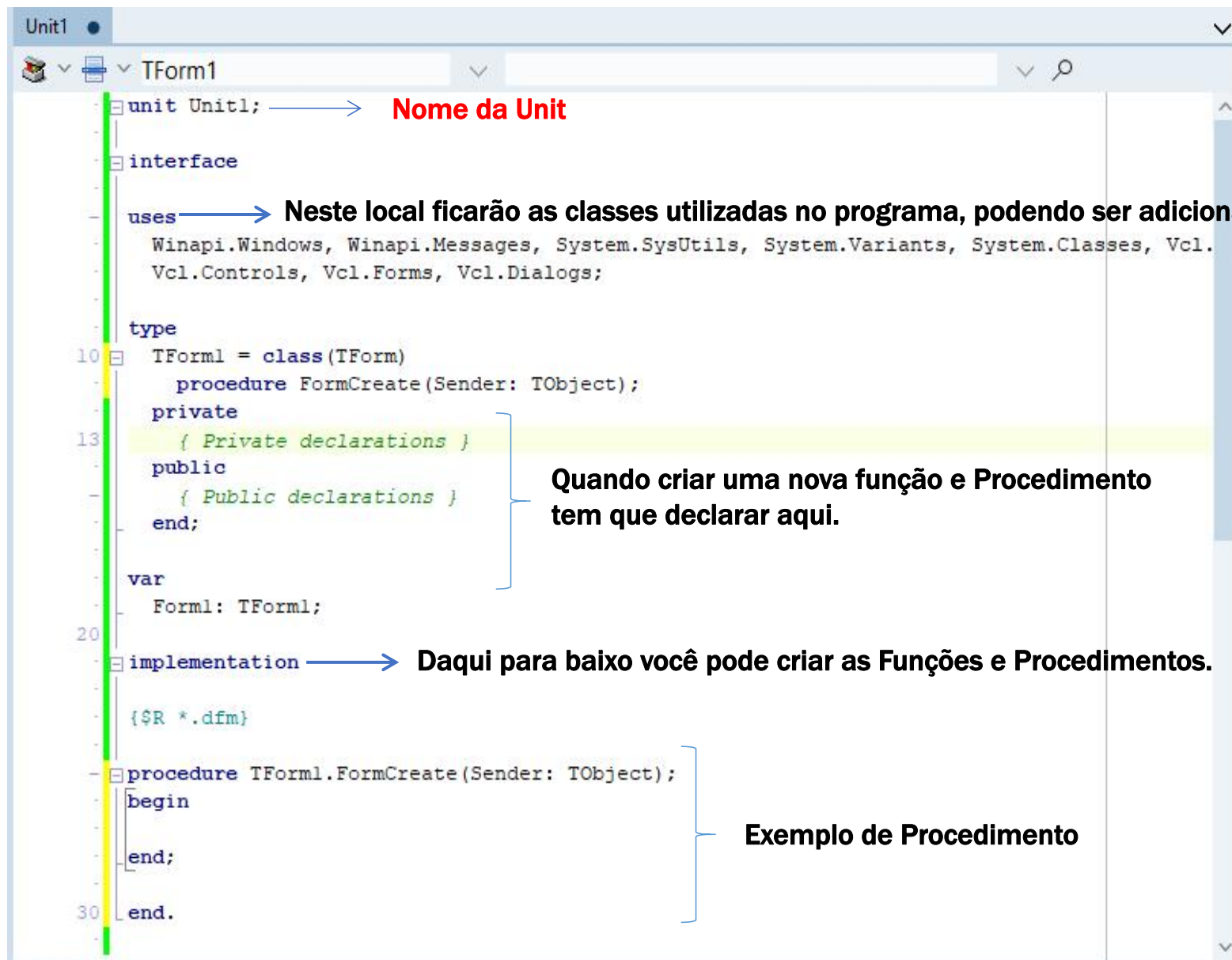


Módulo 1

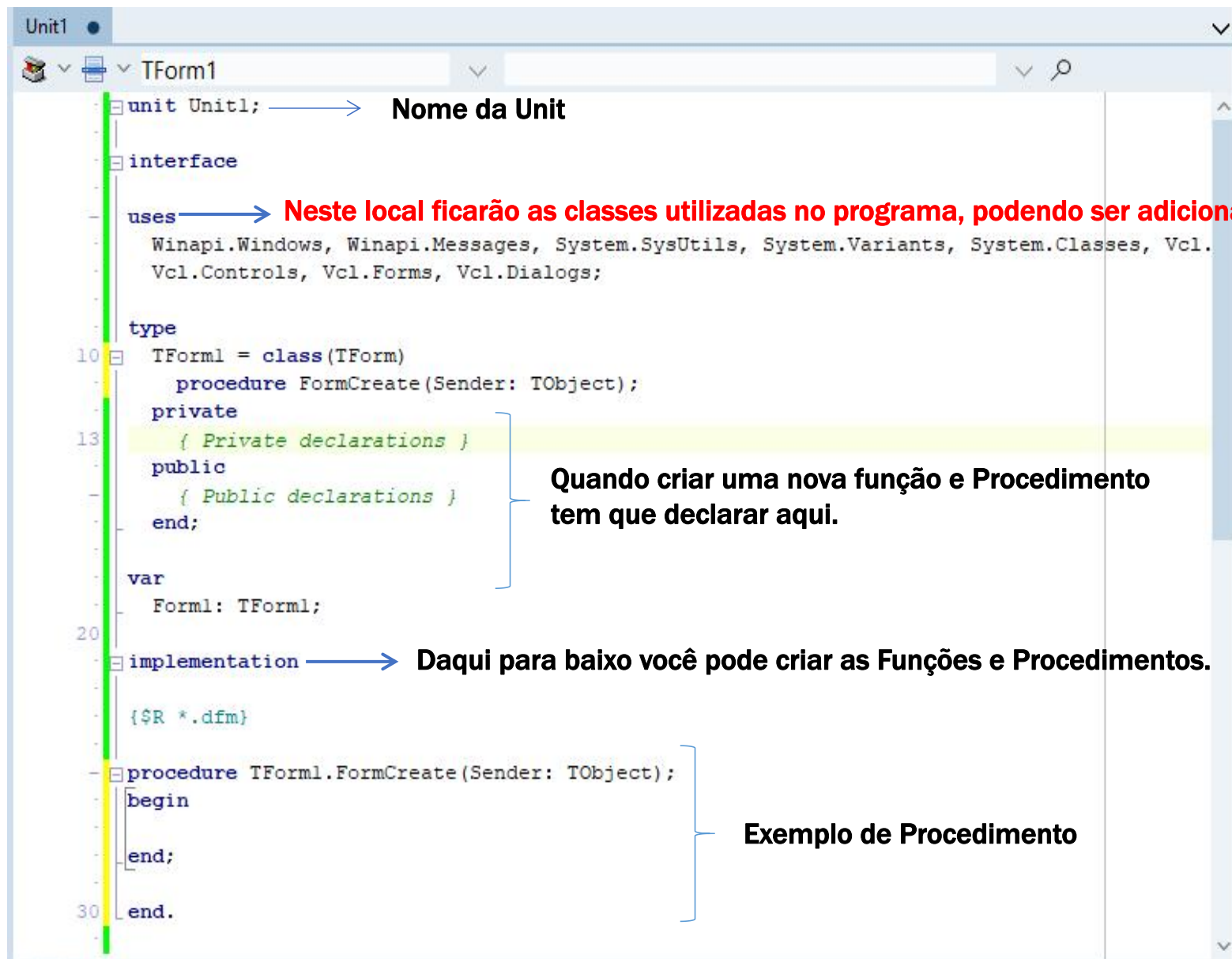
AULA 2



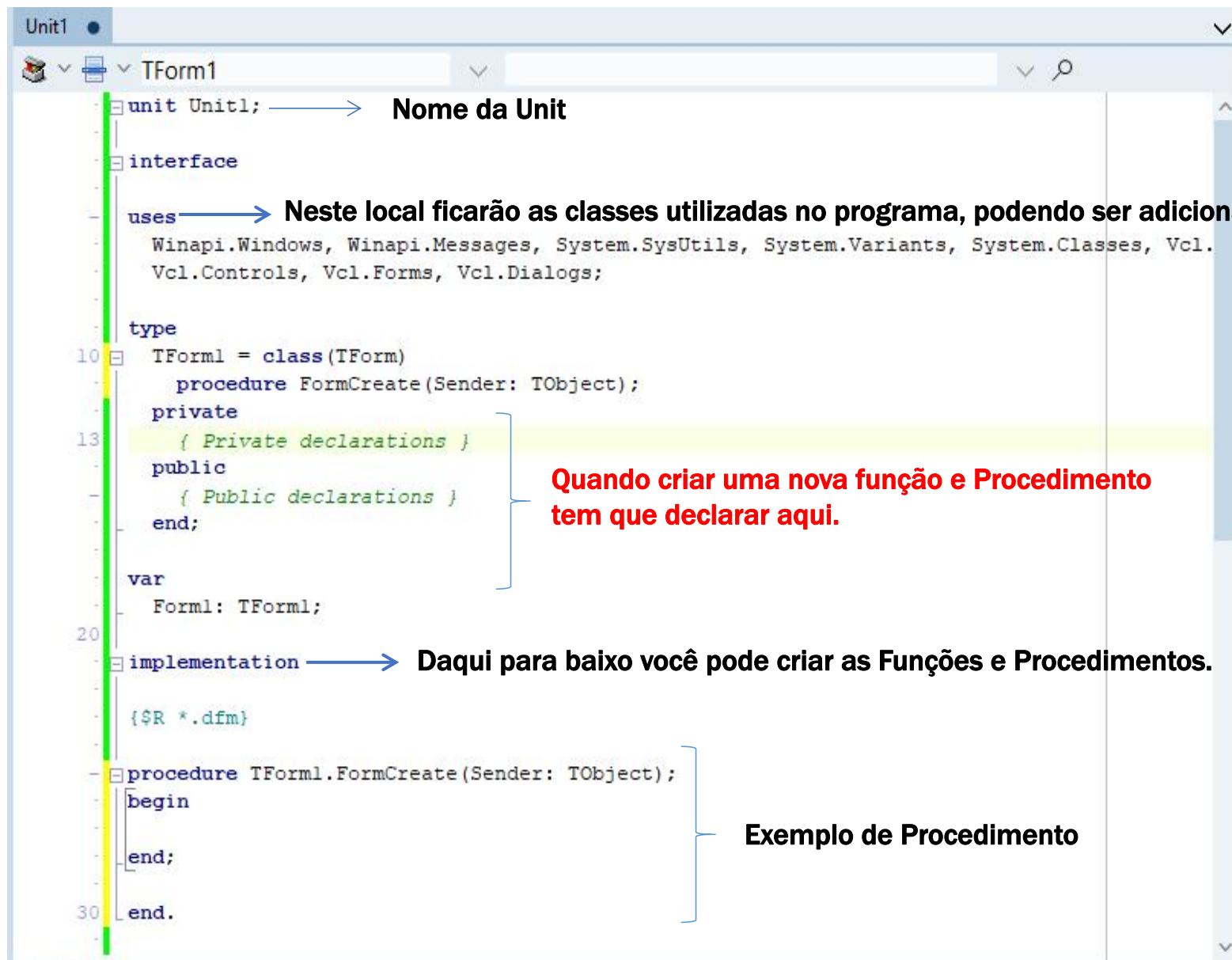
ENTENDENDO A UNIT



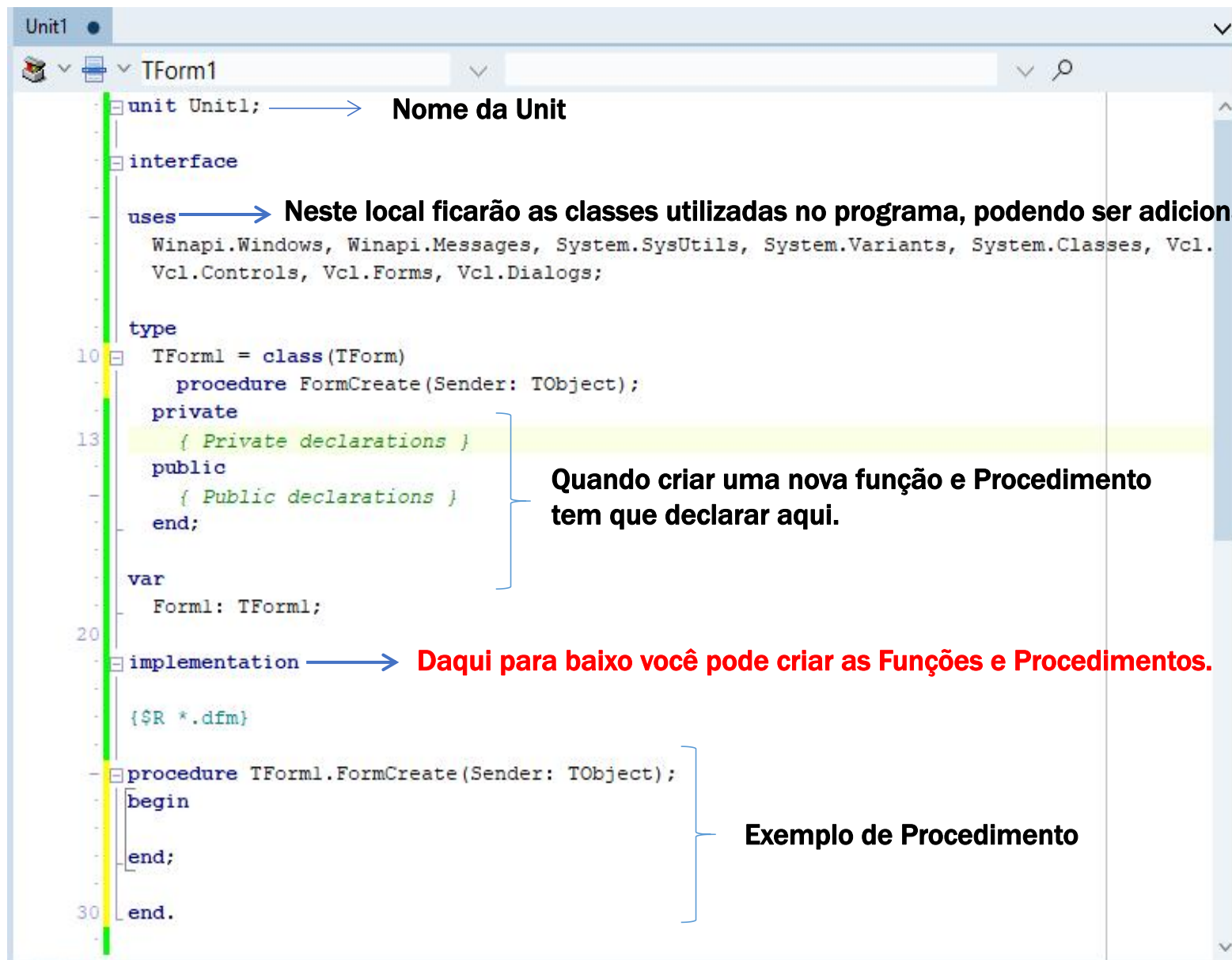
ENTENDENDO A UNIT



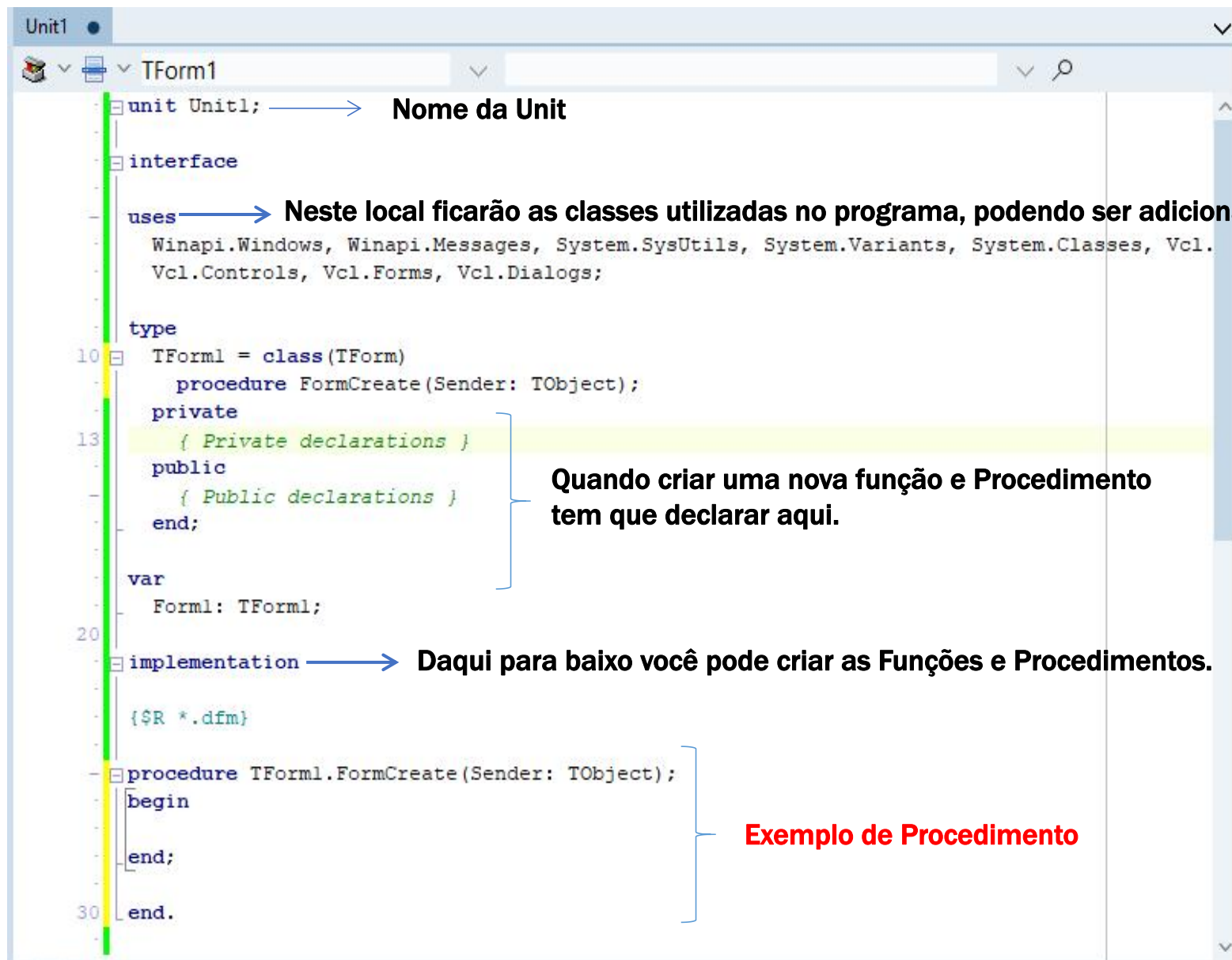
ENTENDENDO A UNIT



ENTENDENDO A UNIT



ENTENDENDO A UNIT



IDENTANDO O CÓDIGO

- A **identação** ou **tabulação do código fonte** consiste na organização ou formatação do seu código para que ele fique legível e de fácil entendimento para você e para outros programadores que forem trabalhar com ele.
- É importante dizer que na maioria das linguagens não é obrigatório a indentação, mas em todas elas, a formatação do código é extremamente recomendada, fazendo parte das **boas práticas de programação**.

Código **SEM IDENTAÇÃO**

```
procedure TForm1.teste;  
var x, y : double;  
begin  
  if x > 0 then  
    y := y+1;  
  end;  
end.
```

Código **IDENTADO**

```
procedure TForm1.teste;  
var  
    x, y: double;  
begin  
    if x > 0 then  
        y := y + 1;  
    end;  
end.
```


ESCREVENDO O CÓDIGO

- **Fim de comando**

- Sempre no final de cada comando é necessário utilizar o caractere ; (ponto e virgula). Exemplo:
- Variavel := 1 +2;

- **Atribuindo valores a variáveis**

- Assim como foi demonstrado no exemplo anterior, para se atribuir o valor a uma variável, basta usar os caracteres := (dois pontos e igual). Exemplo:
- Variavel := Valor;

ESCREVENDO O CÓDIGO

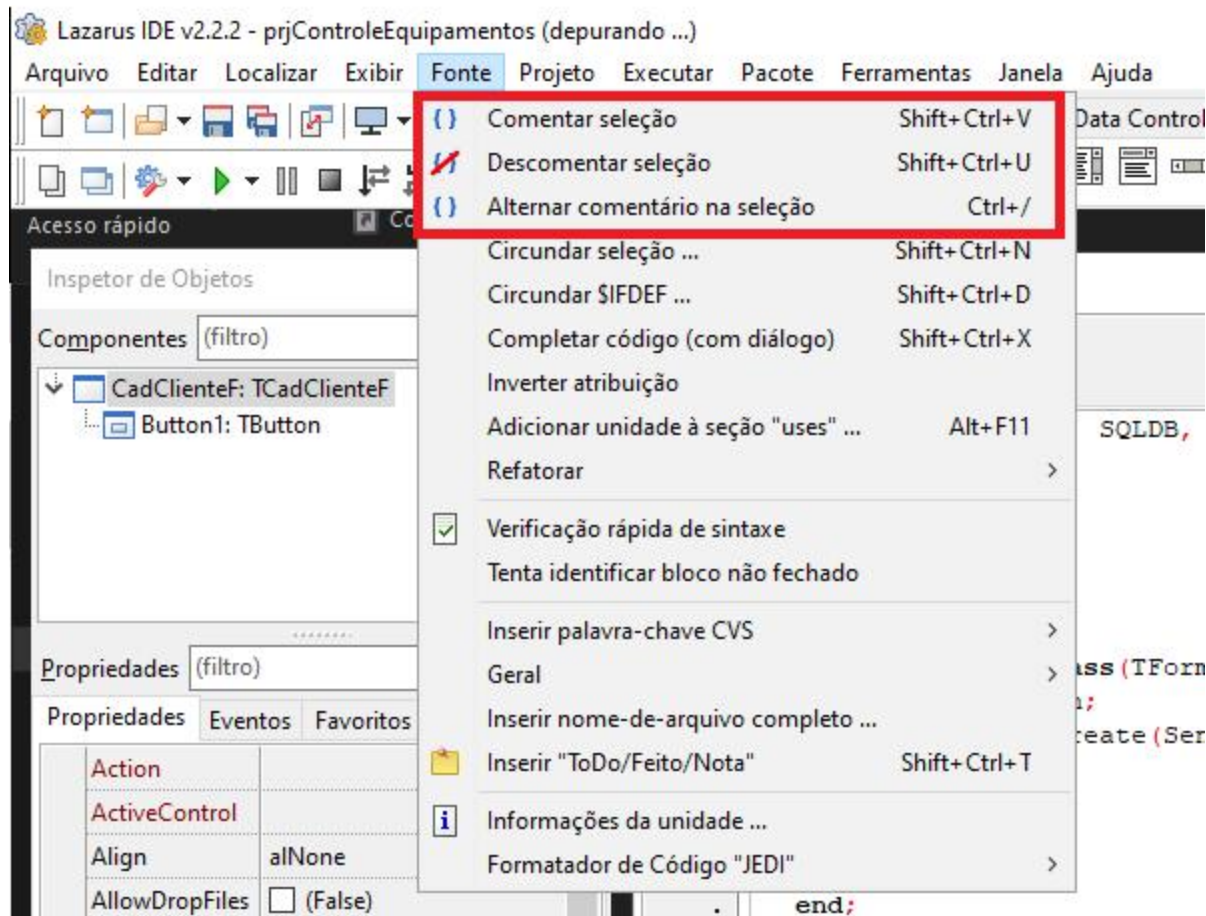
Comentários

Uma boa prática de programação é você fazer comentários explicativos em seu algoritmo, isso pode ajudar a você a lembrar o que foi escrito ou até mesmo ajudar para que outros programadores possam dar manutenção em um código existente. Exemplos:

```
-  
· //Comentário simples (linha)  
·  
· {Comentário em Blocos  
· Onde você pode comentar trechos de códigos,  
40 não esqueça de fechar}
```

ESCREVENDO O CÓDIGO

Comentários



ESCREVENDO O CÓDIGO

Variáveis / Constantes

Variáveis e constantes são regiões de memória onde damos um nome e armazenamos valores de determinados tipos.

- As **variáveis** podem ter seus valor alterado durante a execução do programa.
- As **constantes** permanecem sempre com o mesmo valor com qual foram declaradas.

ESCREVENDO O CÓDIGO

Tipos de Variáveis

- **Integer:** Número inteiro (de -2147483648 a 2147483647).
- **Double:** Ponto flutuante com 64 bits (utilizado para números grandes e com casas decimais).
- **Boolean:** Valor Booleano (true ou false).
- **String:** Cadeia de caracteres (utilizado para textos).
- **Date:** Referência para data ou hora.

ESCREVENDO O CÓDIGO

Funções com **STRINGS**

Função	Aplicação	Sintaxe
Length	Recuperar o tamanho de uma string ou de um Array	<code>VarInteira := Length(varTexto);</code>
Copy	Copia parte de uma string	<code>StrFinal := copy(varTexto, posiçãoInicial, quantosCaracteres);</code>
Pos	Recupera a posição de um determinado caractere (ou conjunto) dentro de uma string	<code>VarInteira := Pos('A',varTexto);</code>
Trim	Elimina espaços a direita e a esquerda de uma string (utilize TrimLeft para eliminar somente esquerda e TrimRight somente direita)	<code>StrFinal := Trim(varTexto);</code>
StringReplace	Substitui parte de uma string por outra	<code>StrFinal := StringReplace(varTexto, caracterASubstituir, ovoCaracter, outrasOpcoes);</code>

ESCREVENDO O CÓDIGO

Funções de **NÚMEROS**

Função	Aplicação	Sintaxe
Inc	Incrementa o valor de uma variavel do tipo inteiro	Inc(varInteiro);
Random	Retorna um numero inteiro aleatório.	VarInteira := Random(range); obs: utilizar sempre a função Randomize antes de usar a random.

ESCREVENDO O CÓDIGO

Funções de **DATA E HORA**

Função	Aplicação	Sintaxe
Date	Retorna a data atual	VarData := Date;
Now	Retornar a data e hora atual	VarDataTempo := now;
IncDay	Incrementa dias a uma data	VarData := incDay(varData, 1) obs: caso não informe a quantidade dia, a função ira incrementar somente 1
DaysBetween	Retorna numero de dias entre uma data e outra	VarInteira := DaysBetween(varData, varProximaData);

ESCREVENDO O CÓDIGO

Funções de **CONVERSÃO DE TIPO**

Função	Aplicação	Sintaxe
IntToStr	Converte um integer para uma string	<code>VarStr := IntToStr(varInteira);</code>
FloatToStr	Converte um double para uma string	<code>VarStr := FloatToStr(varDouble);</code>
StrToInt	Converte uma string para um Integer	<code>VarInteira := StrToInt(varStr);</code>
StrToFloat	Converte uma string para um double	<code>VarDouble := StrToFloat(varStr);</code>
DateToStr	Converte um Date para uma String	<code>VarStr := DateToStr(varData);</code>
DateTimeToStr	Converte um DateTime (data hora) para uma string	<code>VarStr := DateTimeToStr(varDataEHora);</code>
TimeToStr	Converte um Time para uma string	<code>VarStr := TimeToStr(varHora);</code>
StrToDate	Converte uma String em um Date	<code>VarData := StrToDate(varStr);</code>
FloatToStrF	Converte um Double para uma String formatada	<code>VarStr := FloatToStrF(varDouble,formato, precisao, casas decimais);</code>

ESCREVENDO O CÓDIGO

Estrutura de Decisão

- No momento da escrita do software se faz necessário que o aplicativo seja capaz de tomar diferentes caminhos de acordo com as condições que apareçam no decorrer de sua execução. Para que isso seja possível existem as estrutura de decisões, tendo como principal representante o IF/ELSE.

ESCREVENDO O CÓDIGO

Estrutura de Decisão - IF/ELSE

if simples sem ELSE

```
if <condição> then  
begin  
end;
```

if com ELSE

```
if <condição> then  
begin  
end  
else  
begin  
end;
```

ESCREVENDO O CÓDIGO

Variáveis / Constantes

Constantes

```
public
{ Public declarations }
procedure Teste;
end;

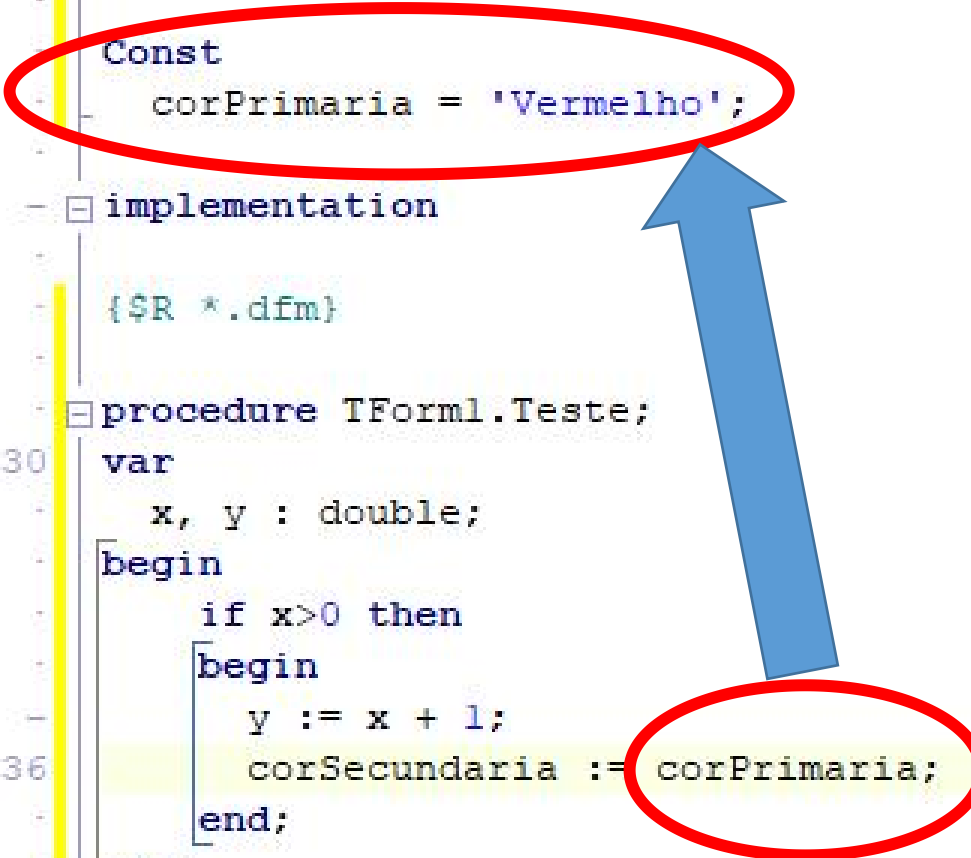
var
  Form1: TForm1;
  corSecundaria : String;

Const
  corPrimaria = 'Vermelho';

implementation

{$R *.dfm}

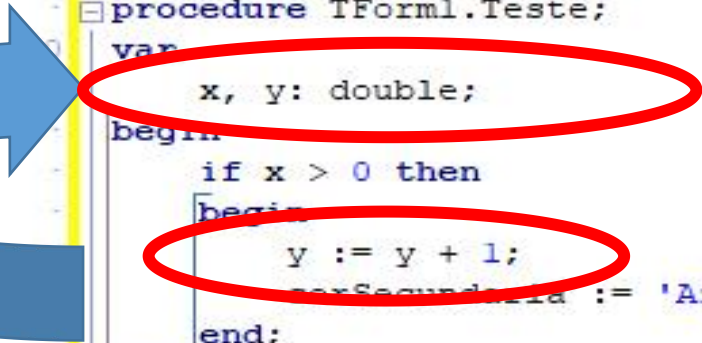
procedure TForm1.Teste;
var
  x, y : double;
begin
  if x>0 then
  begin
    y := x + 1;
    corSecundaria := corPrimaria;
  end;
end;
```



ESCREVENDO O CÓDIGO

Variáveis / Constantes

Variável LOCAL



```
type
TForm1 = class(TForm)
private
{ Private declarations }
public
{ Public declarations }
procedure Teste;
end;

var
Form1: TForm1;
corSecundaria: String;

Const
corPrincipal = 'Vermelho';

implementation

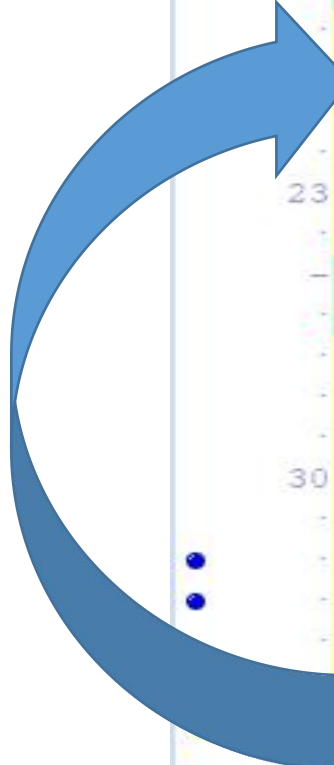
{$R *.dfm}

procedure TForm1.Teste;
var
x, y: double;
begin
if x > 0 then
begin
y := y + 1;
corSecundaria := 'Azul';
end;
end;
```

ESCREVENDO O CÓDIGO

Variáveis / Constantes

Variável GLOBAL



```
type
10  TForm1 = class(TForm)
    private
        { Private declarations }
    public
        { Public declarations }
        procedure Teste;
    end;

var
    Form1: TForm1;
    corSecundaria: String;

Const
23  corPrincipal = 'Vermelho';

implementation

{$R *.dfm}

30  procedure TForm1.Teste;
    var
        x, y: double;
    begin
        if x > 0 then
        begin
            y := y + 1;
            corSecundaria := 'Azul';
        end;
    end;
```

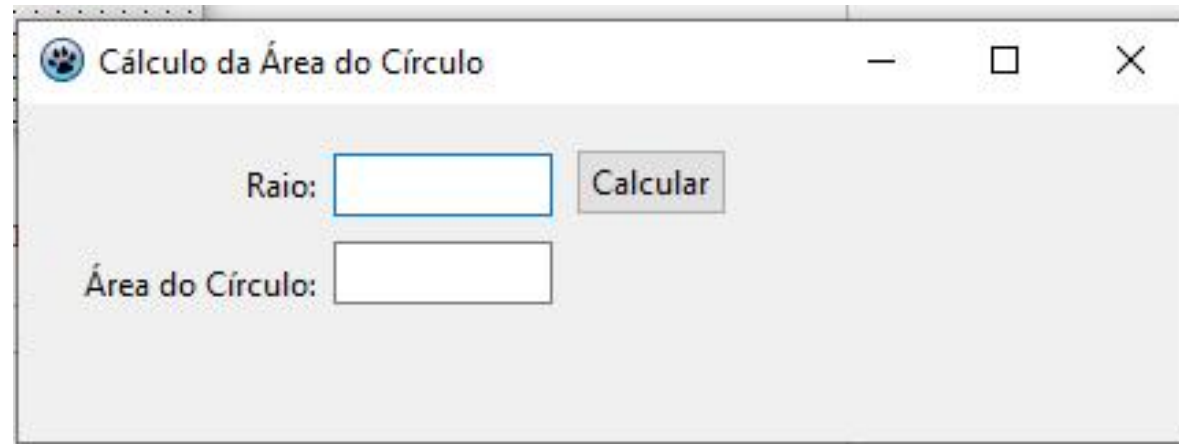
ESCREVENDO O CÓDIGO

Funções e Procedimentos

- As **funções** (**functions**), são blocos de códigos também conhecidas como sub-rotinas, são utilizadas com o intuito de não precisar copiar o código todas as vezes que precisar executar aquela operação, além de deixar a leitura do código mais intuitiva.
- Os **procedimentos** (**procedures**) também são blocos de códigos, porém, eles se diferem das funções apenas por não retornarem resultado, um exemplo seria o um procedimento que envia e-mail onde ele não precisa retornar valores apenas executar um processo.

ESCREVENDO O CÓDIGO

Declarando Funções – Cálculo da Área do Círculo



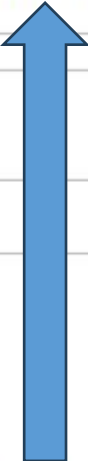
A screenshot of a Windows application window titled "Cálculo da Área do Círculo". The window has a standard Windows title bar with a minimize button, a maximize button, and a close button. The main content area is light gray and contains two input fields and a button. The first input field is labeled "Raio:" and is empty. To its right is a button labeled "Calcular". Below the "Raio:" field is another input field labeled "Área do Círculo:", which is also empty.

ESCREVENDO O CÓDIGO

Declarando Funções

*Declaração
Pública ou
Privada*

```
untAreadoCirculo
10 type
. { TForm1 }
. TForm1 = class(TForm)
.   btnCalculaArea: TButton;
.   edtRaio: TEdit;
.   edtArea: TEdit;
15   Label1: TLabel;
.   Label2: TLabel;
.   procedure btnCalculaAreaClick(Sender: TObject);
. private
. public
20   function calculaAreaCirculo (Raio: Double): Double;
. end;
.
.
25 var
.   Form1: TForm1;
.
. const
.   Pi = 3.1515;
.
30 implementation
.
.   {$R *.1fm}
.
. { TForm1 }
35
. function TForm1.calculaAreaCirculo (Raio: Double): Double;
. begin
.   Result := Pi * (Raio * Raio);
39 end;
40
. procedure TForm1.btnCalculaAreaClick(Sender: TObject);
. begin
.   edtArea.Text := FloatToStr (calculaAreaCirculo(StrToFloat(edtRaio.Text)));
. end;
45
. end.
47
```



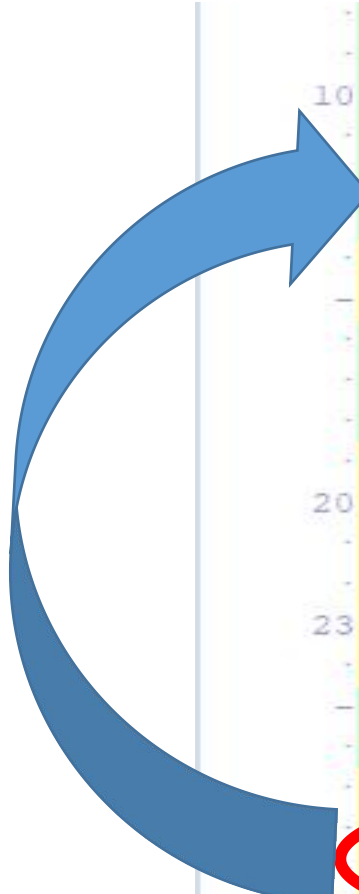
Utilizando/chamando a função



ESCREVENDO O CÓDIGO

Declarando Procedimentos

Declaração Pública ou Privada



```
type
  TForm1 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
    procedure Teste;
  end;

var
  Form1: TForm1;
  corSecundaria: String;

Const
  corPrincipal = 'Vermelho';


implementation

  {$R *.res}

  procedure TForm1.Teste;
  var
    x, y: double;
  begin
    if x > 0 then
    begin
      y := y + 1;
      corSecundaria := 'Azul';
    end;
  end;
```

ESCREVENDO O CÓDIGO

Utilizando/chamando um procedimento



```
30 procedure TForm1.Button1Click(Sender: TObject);  
31 begin  
32     MostraMensagem;  
33 end;  
34  
35 procedure TForm1.MostraMensagem;  
36 begin  
37     ShowMessage('Estou chamando de uma Procedure');  
38 end;  
40
```

The diagram illustrates a call from the `Button1Click` procedure to the `MostraMensagem` procedure. A blue curved arrow originates from the `MostraMensagem;` line within the `Button1Click` procedure and points to the `procedure TForm1.MostraMensagem;` definition below it.

ESCREVENDO O CÓDIGO

Utilizando/chamando um procedimento

```
30 |  
31 | procedure TForm1.Button1Click(Sender: TObject);  
32 | begin  
33 |     MostraMensagem;  
34 | end;  
35 |  
36 |  
37 |  
38 |  
39 |  
40 | procedure TForm1.MostraMensagem;  
41 | begin  
42 |     ShowMessage('Estou chamando de uma Procedure');  
43 | end;
```

Nome do Formulário Nome da Procedure

ESCREVENDO O CÓDIGO

Operadores Lógicos

- São usados principalmente em estruturas de decisão.
- Você pode utilizar os operadores para definir em que momento será realizado um bloco de códigos.
- Os operadores lógicos nos ajudam a realizar operações sobre um ou dois valores booleano (verdadeiro ou falso).

ESCREVENDO O CÓDIGO

Operadores Lógicos

- Igual (=)
- Diferente (<>)
- E (and)
- Ou (or)
- Negação (not)

ESCREVENDO O CÓDIGO

Operadores Lógicos

- Igual (=)



A

VALOR1 = VALOR2

- Diferente (<>)

- E (and)

- Ou (or)

- Negação (not)

ESCREVENDO O CÓDIGO

Operadores Lógicos

- Igual (=)



A

VALOR1 = VALOR2

```
procedure TForm1.Button1Click(Sender: TObject);  
var  
    idadeMaioridade, idadePessoa : Integer;  
begin  
    idadeMaioridade := 18;  
    if idadePessoa = idadeMaioridade then  
        ShowMessage('Esta pessoa tem 18 anos.')  
    else  
        ShowMessage('Esta pessoa tem idade diferente de 18 anos.');
```

ESCREVENDO O CÓDIGO

Operadores Lógicos

- Igual (=)
- Diferente (<>)
- E (and)
- Ou (or)
- Negação (not)



A
VALOR1 <> VALOR2

ESCREVENDO O CÓDIGO

Operadores Lógicos

- Diferente (<>)



A

VALOR1 <> VALOR2

```
procedure TForm1.Button1Click(Sender: TObject);  
var  
    Fruta, FrutadaEstacao : String;  
begin  
    FrutadaEstacao := 'MANGA';  
    if Fruta <> FrutadaEstacao then  
        ShowMessage('Esta fruta não é desta Estação.')  
    else  
        ShowMessage('Esta fruta é da estação.');
```


ESCREVENDO O CÓDIGO

Operadores Lógicos

- Igual (=)
- Diferente (<>)
- E (and)
- Ou (or)
- Negação (not)



A	B	A E B
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

ESCREVENDO O CÓDIGO

Operadores Lógicos

- E (and)

```
procedure TForm1.Button1Click(Sender: TObject);  
var  
    LimiteMedia, MaximoFaltas, MediaAluno, nrFaltas : Double;  
begin  
    LimiteMedia := 70;  
    MaximoFaltas := 25;  
    if (MediaAluno >= LimiteMedia) AND (nrFaltas <= MaximoFaltas) then  
        ShowMessage('Aluno Aprovado.')    else  
        ShowMessage('Aluno Reprovado.');end;
```

ESCREVENDO O CÓDIGO

Operadores Lógicos

- Igual (=)
- Diferente (<>)
- E (and)
- Ou (or)
- Negação (not)



A	B	A OU B
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

ESCREVENDO O CÓDIGO

Operadores Lógicos

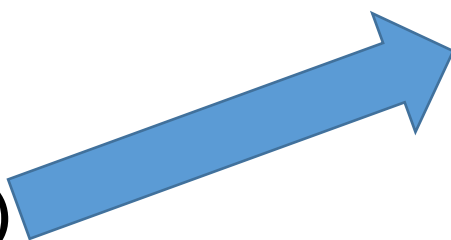
- Ou (or)

```
procedure TForm1.Button1Click(Sender: TObject);
var
    LimiteMedia, MaximoFaltas, MediaAluno, nrFaltas : Double;
begin
    LimiteMedia := 70;
    MaximoFaltas := 25;
    if (MediaAluno < LimiteMedia) OR (nrFaltas > MaximoFaltas) then
        ShowMessage('Aluno Reprovado.')
    end;
end;
```

ESCREVENDO O CÓDIGO

Operadores Lógicos

- Igual (=)
- Diferente (<>)
- E (and)
- Ou (or)
- Negação (not)



A	NÃO A
TRUE	FALSE
FALSE	TRUE

ESCREVENDO O CÓDIGO

Operadores Lógicos

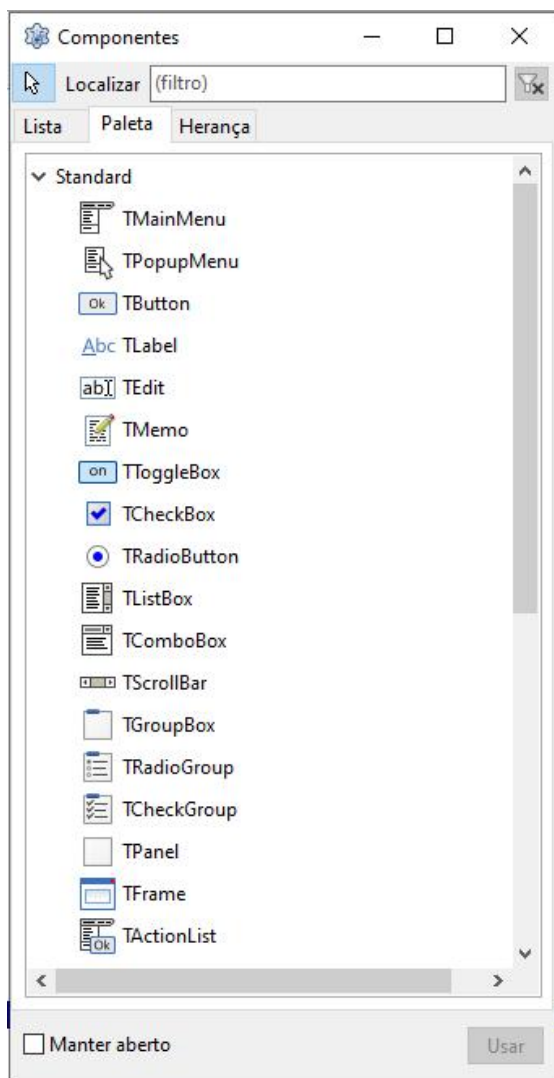
- Negação (not)

```
procedure TForm1.Button1Click(Sender: TObject);
var
    LimiteMedia, MaximoFaltas, MediaAluno, nrFaltas : Double;
    bAprovado : Boolean;
begin
    LimiteMedia := 70;
    MaximoFaltas := 25;
    if (MediaAluno >= LimiteMedia) AND (nrFaltas <= MaximoFaltas) then
        bAprovado := True
    else
        bAprovado := False;

    if not (bAprovado) then
        showmessage('Reprovado')
    else
        showmessage('Aprovado');
```

PALLETE

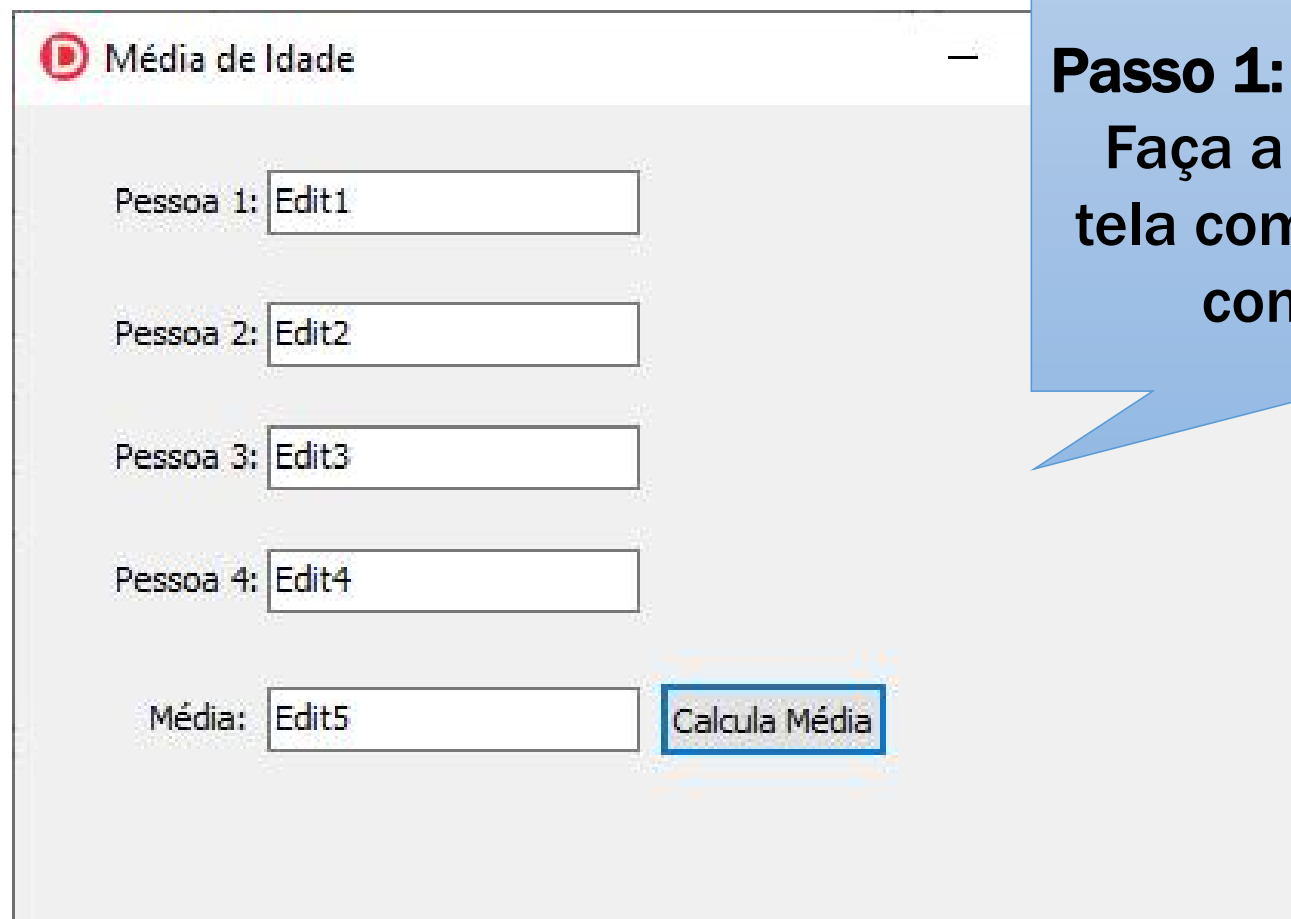
Standard



PALLETE - STANDARD

Criando uma tela

- Faça um programa para Calcular a Média de 4 pessoas.



The screenshot shows a Java Swing window titled "Média de Idade". Inside the window, there are four text input fields labeled "Pessoa 1:", "Pessoa 2:", "Pessoa 3:", and "Pessoa 4:". Each field contains the text "Edit1", "Edit2", "Edit3", and "Edit4" respectively. Below these fields is another input field labeled "Média:" containing "Edit5". To the right of the "Média:" field is a button labeled "Calcula Média". The button is highlighted with a blue border and a yellow glow effect.

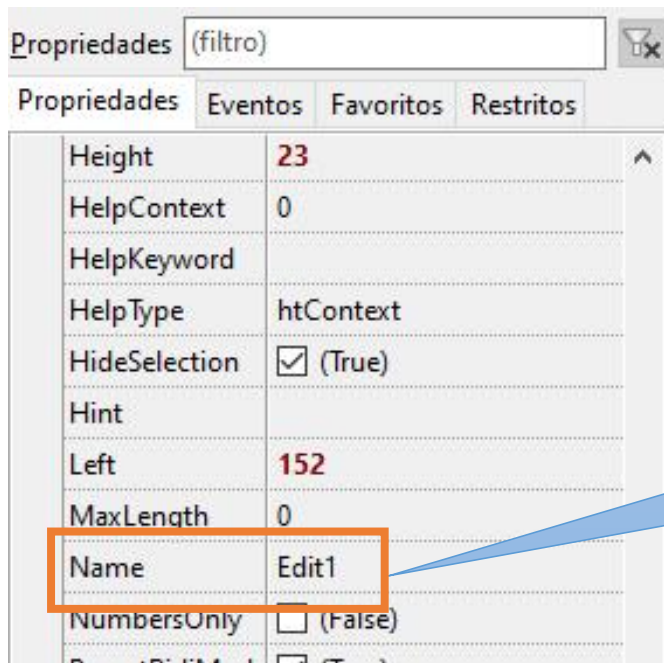
Passo 1:

Faça a montagem da tela com os respectivos componentes.

PALLETE - Standard

Criando uma tela

- Faça um programa para Calcular a Média de 4 pessoas.



Passo 2:

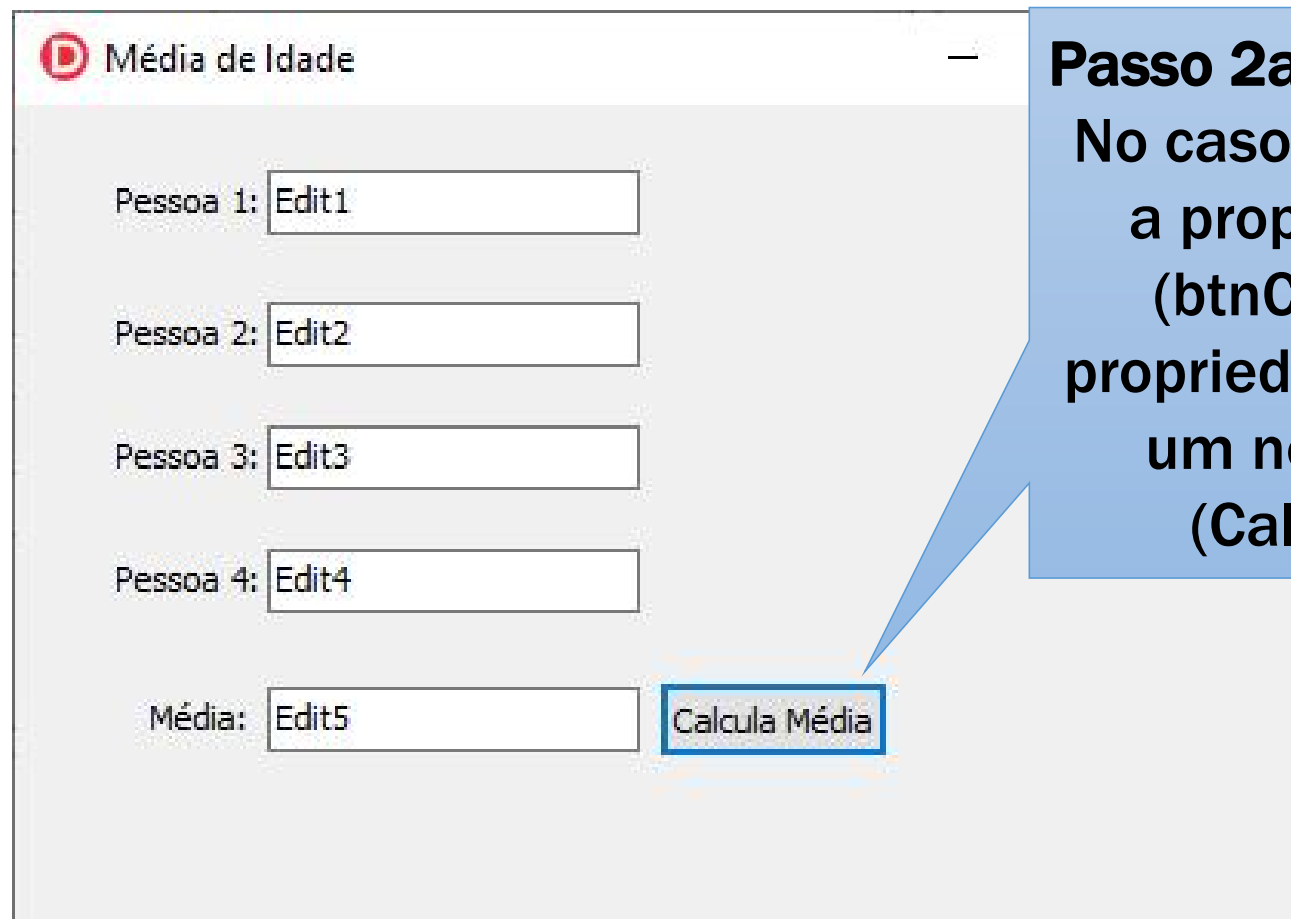
Ajuste o nome de cada componente, modificando a propriedade Name (padronise) e retirando o texto dos edits da propriedade Text.



PALLETE - STANDARD

Criando uma tela

- Faça um programa para Calcular a Média de 4 pessoas.

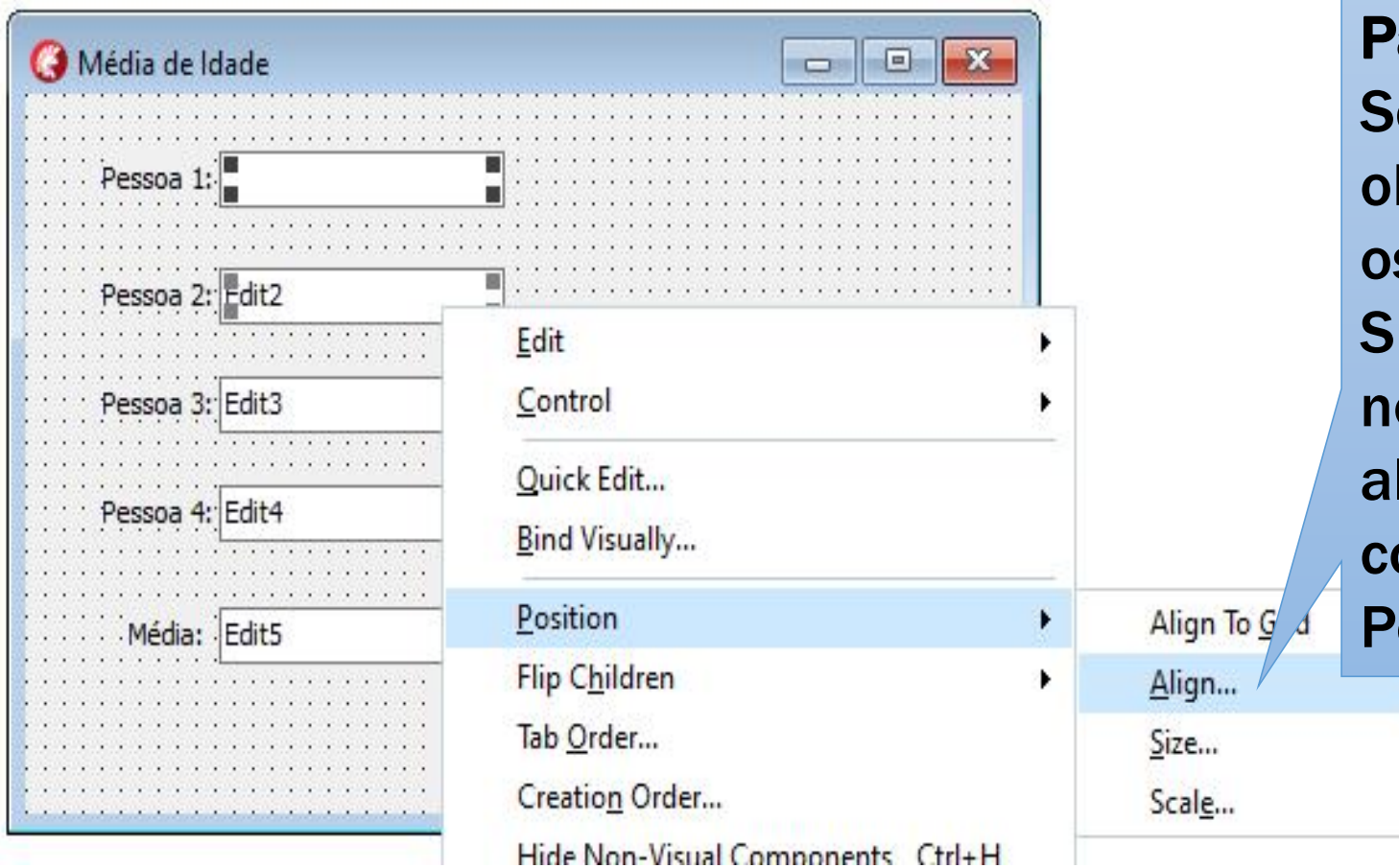


The screenshot shows a Windows application window titled "Média de Idade". Inside the window, there are four rows of input fields labeled "Pessoa 1:", "Pessoa 2:", "Pessoa 3:", and "Pessoa 4:". Each input field contains the text "Edit1", "Edit2", "Edit3", and "Edit4" respectively. Below these, there is a row labeled "Média:" followed by an input field containing "Edit5". To the right of the "Média:" input field is a button labeled "Calcula Média". A blue callout box points to this button, containing the text "Passo 2a: No caso do botão, ajuste a propriedade Name (btnCalcMedia) e a propriedade Caption com um nome amigável (Calcula Média).".

PALLETE - STANDARD

Criando uma tela

- Faça um programa para Calcular a Média de 4 pessoas.



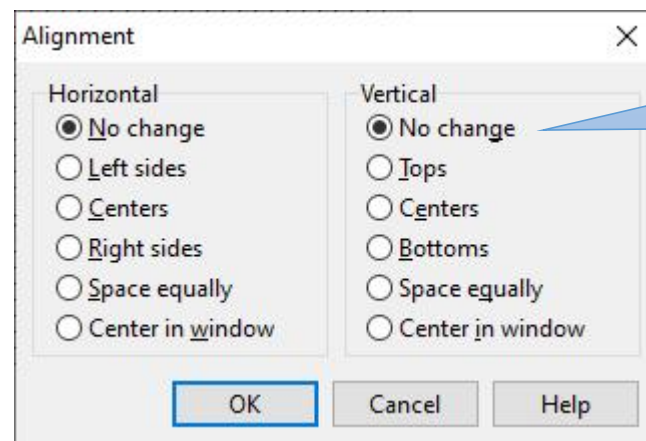
Passo 3a:

Se necessário, alinhe seus objetos, para isto selecione os objetos utilizando a tecla Shift do Teclado e clique nos objetos que você deseja alinhar. Após isto, clique com o botão direito e vá em Position / Align:

PALLETE - STANDARD

Criando uma tela

- Faça um programa para Calcular a Média de 4 pessoas.



Passo 3b:

Aqui você pode escolher o que você deseja alinhar e em que posição.

PALLETE - STANDARD

Criando uma tela

- Faça um programa para Calcular a Média de 4 pessoas.



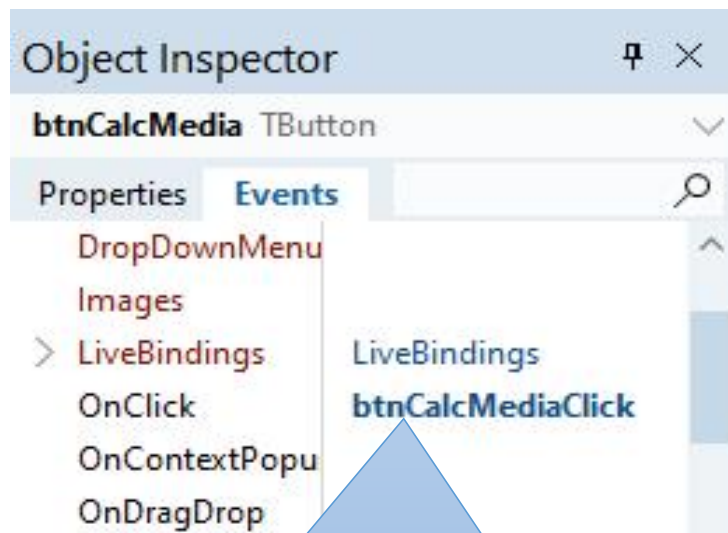
The image shows a Java Swing window titled "Média de Idade". Inside the window, there are four text input fields labeled "Pessoa 1:", "Pessoa 2:", "Pessoa 3:", and "Pessoa 4:". Below these fields is another input field labeled "Média:". To the right of the "Média:" field is a button labeled "Calcula Média". The button is highlighted with a blue border and a blue callout bubble pointing to it.

Passo 5:
Programando a ação, no
Evento Onclick do botão
Calcula Média.

PALLETE - STANDARD

Criando uma tela

- Faça um programa para Calcular a Média de 4 pessoas.



Para programar dê dois cliques no evento Onclick ou dois cliques no objeto ele irá gerar a Procedure.

```
procedure TForm1.btnCalcMediaClick(Sender: TObject);  
begin  
end;
```

Passo 6:
Aqui colocaremos nosso código...

PALLETE - STANDARD

Programando na tela

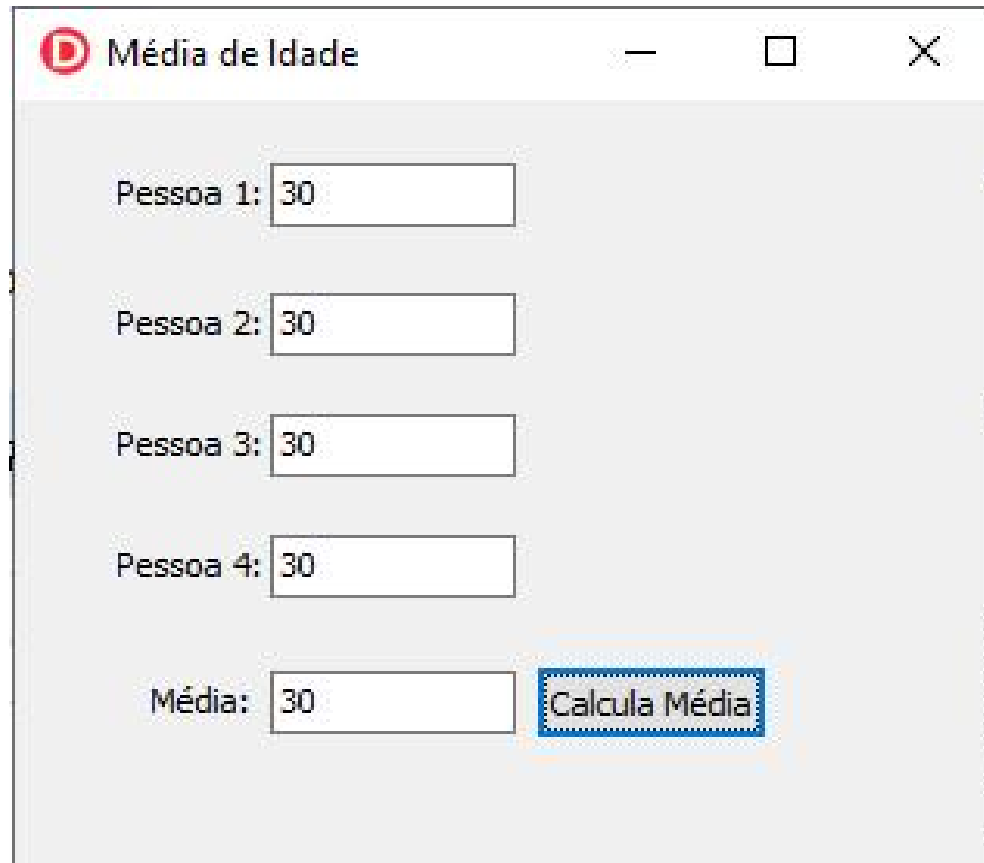
- Faça um programa para Calcular a Média de 4 pessoas.

```
procedure TForm1.btnCalcMediaClick(Sender: TObject);
var
50   idadePes1, idadePes2, idadePes3, idadePes4, Media: Double;
begin
    //Passando os valores dos edits para as variáveis
    //Utilizando o StrToFloat para converter Texto em Double
    idadePes1 := StrToFloat(edtPessoa1.Text);
    idadePes2 := StrToFloat(edtPessoa2.Text);
    idadePes3 := StrToFloat(edtPessoa3.Text);
    idadePes4 := StrToFloat(edtPessoa4.Text);
    //A variável Média Recebe a soma das idades dividido pela quantidade
    Media := (idadePes1 + idadePes2 + idadePes3 + idadePes4) / 4;
60   //Passando o Resultado da Média para o Edit e assim mostrar o resultado.
    edtMedia.Text := FloatToStr(Media);
end;
```


PALLETE - STANDARD

Resultado

- Faça um programa para Calcular a Média de 4 pessoas.



The image shows a Java Swing window titled "Média de Idade" with standard window controls (minimize, maximize, close). Inside the window, there are four text input fields, each preceded by a label "Pessoa 1:", "Pessoa 2:", "Pessoa 3:", and "Pessoa 4:". Each of these fields contains the number "30". Below these is a fifth input field labeled "Média:" which also contains "30". To the right of the "Média:" field is a button labeled "Calcula Média". The button has a blue dashed border and is highlighted by a light orange rectangular selection box.

ESCREVENDO O CÓDIGO

Exercícios de Fixação

- 1) Faça um programa que receba 4 notas dos alunos, onde cada nota será relativa a média de cada Bimestre, faça a soma e tire a média delas. Após isto verifique se o aluno foi aprovado ou reprovado. Utilize uma constante para armazenar a média de aprovação e o número máximo de faltas (caso ele esteja com mais faltas que o número máximo estará automaticamente reprovado independente da nota). Insira edits para receber os valores e Mostre para o usuário o resultado e depois altere o valor das variáveis para chegar em diferentes resultados.

ESCREVENDO O CÓDIGO

Exercícios de Fixação

- 2) Calcule quanto um funcionário recebe mês a mês em um ano (Bruto e Líquido), após isto, mostre o total em um ano (Ignorando 13º salário e férias). Na tela coloque um componente edit para cada mês, onde será digitado o Salario Bruto, o percentual de desconto e o salário líquido, no final coloque um edit para mostrar a soma do salário Bruto e do Líquido.

ESCREVENDO O CÓDIGO

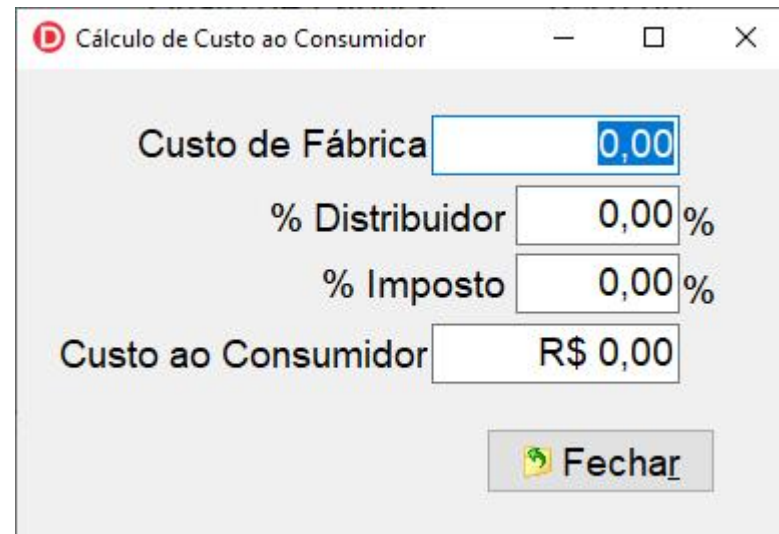
Exercícios de Fixação

- 3) Desenvolva um programa que faça conversão de grau Celsius para Fahrenheit.
- 4) Faça um programa que leia a idade de uma pessoa expressa em dias e mostre-a expressa em anos, meses e dias.
- 5) Faça um algoritmo que leia o tempo de duração de um evento em uma fábrica expressa em segundos e mostre-o expresso em horas, minutos e segundos.

ESCREVENDO O CÓDIGO

Exercícios de Fixação

- 6) O custo ao consumidor de um carro novo é a soma do custo de fábrica com a percentagem do distribuidor e dos impostos (aplicados ao custo de fábrica). Supondo que a percentagem do distribuidor seja de 28% e os impostos de 45%, escrever um algoritmo que leia o custo de fábrica de um carro e escreva o custo ao consumidor.




Cálculo de Custo ao Consumidor

Custo de Fábrica

% Distribuidor %

% Imposto %

Custo ao Consumidor

 Fechar

REFERÊNCIAS

- <https://bit.ly/3s5DKck>
- <https://bit.ly/3H17tqZ>
- <https://bit.ly/3lNyLcP> (Adaptado)
- <https://www.lazarus-ide.org/>