



Computação Gráfica

Aula 3

Processamento de Imagens

Conceitos e Tecnologia

Profa. Fátima Nunes

Questões

- O que já sabemos:
 - Diferença e semelhança entre PI, CG e RV
 - O que é uma imagem digital
 - Resolução espacial
 - Resolução de contraste
 - Definição de pixel

Questões

- O que aprenderemos hoje:
 - Relações de vizinhança entre pixels
 - Distância entre pixels
 - Histograma: conceitos, implementação e operações
 - Como manipular uma imagem em uma determinada tecnologia

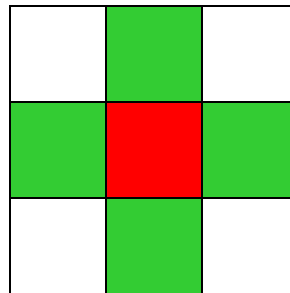
Sobre o pixel são definidas
algumas relações básicas:
vizinhança, conectividade,
distância ...

Questões

- O que é vizinhança de um pixel?

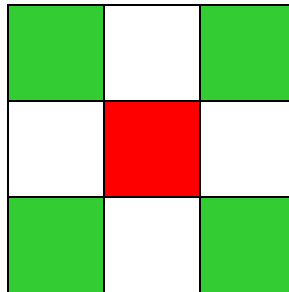
- Vizinhança

- seja p , um pixel nas coordenadas (x,y) . A vizinhança de 4 (ou $N_4(p)$) de um pixel é composta por seus vizinhos na horizontal e na vertical, cujas coordenadas são: $(x+1,y)$, $(x-1,y)$, $(x,y+1)$, $(x,y-1)$.



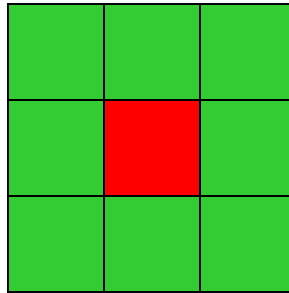
- Vizinhança

- A vizinhança diagonal (ou $N_D(p)$) de um pixel é constituída por seus vizinhos com coordenadas: $(x+1,y+1)$, $(x+1,y-1)$, $(x-1,y+1)$, $(x-1,y-1)$.



- Vizinhança

- A vizinhança de 8 (ou $N_8(p)$) é o conjunto de todos os pixels vizinhos, ou seja: $N_4(p) \cup N_D(p)$.



Questões

- O que é adjacência de um pixel?

- Adjacência

- É característica de um par de pixels vizinhos que compartilham uma borda ou um vértice, sendo que:
 - par de pixels que compartilham uma borda ▶
“adjacente por borda” ou “4-adjacente”;
 - par de pixels que compartilham um vértice ▶
“adjacente por vértice” ou “8-adjacente”.

Questões

- O que é conectividade entre pixels?

- Conectividade

- Conceito importante para estabelecer bordas de objetos e componentes de regiões em uma imagem.
- Dois pixels são conectados se:
 - a) são adjacentes
 - b) obedecem a um critério de similaridade dentro de uma escala de cinza, isto é, seus valores estão dentro de um conjunto pré-estabelecido de valores de cinza.

- Conectividade

- Seja $V=\{G_1, G_2, \dots, G_k\}$ o conjunto de “k” valores de níveis de cinza usado para definir a conectividade. São definidos três tipos de conectividade:

- Conectividade-4
 - Conectividade-8
 - Conectividade-m

- **Conectividade**

- **Conectividade-4:** dois pixels p e q com valores em V e $q \supset N_4(p)$
- **Conectividade-8:** dois *pixels* p e q com valores em V e $q \supset N_8(p)$
- **Conectividade-m (mista):** dois pixels p e q com valores em V e:
 - i) $q \supset N_4(p)$ ou
 - ii) $q \supset N_D(p)$ e $N_4(p) \cap N_4(q) = \emptyset$.

Questões

- Para que serve a mensuração da distância entre pixels?
- Como se faz?

- Distância entre pixels

- Geralmente é um valor mensurável e:

- $d(x,y) = 0$, se $x = y$;
- $d(x,y) = d(y,x)$;
- $d(x,y) + d(y,z) \geq d(x,z)$.

- Diversas fórmulas empregadas para a definição de distância.

- São definidas e adaptadas fórmulas para aplicações específicas.

- Distância entre pixels

- Algumas das métricas mais conhecidas, aplicadas para dois pixels $p=(x_1, y_1)$ e $q=(x_2, y_2)$:

- Distância Euclidiana:

$$d(p, q) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

- Distância “City Block”:

$$d(p, q) = |x_1 - x_2| + |y_1 - y_2|$$

- Distância “Chessboard”:

$$d(p, q) = \max \{|x_1 - x_2|, |y_1 - y_2|\}$$

Definições

- Qual o valor dessas distâncias para os pixels destacados em vermelho?

1	2	3
4	5	6
7	8	9

- $$d(p, q) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$
- $$d(p, q) = |x_1 - x_2| + |y_1 - y_2|$$
- $$d(p, q) = \max \{|x_1 - x_2|, |y_1 - y_2|\}$$

Questões

- O que é histograma?

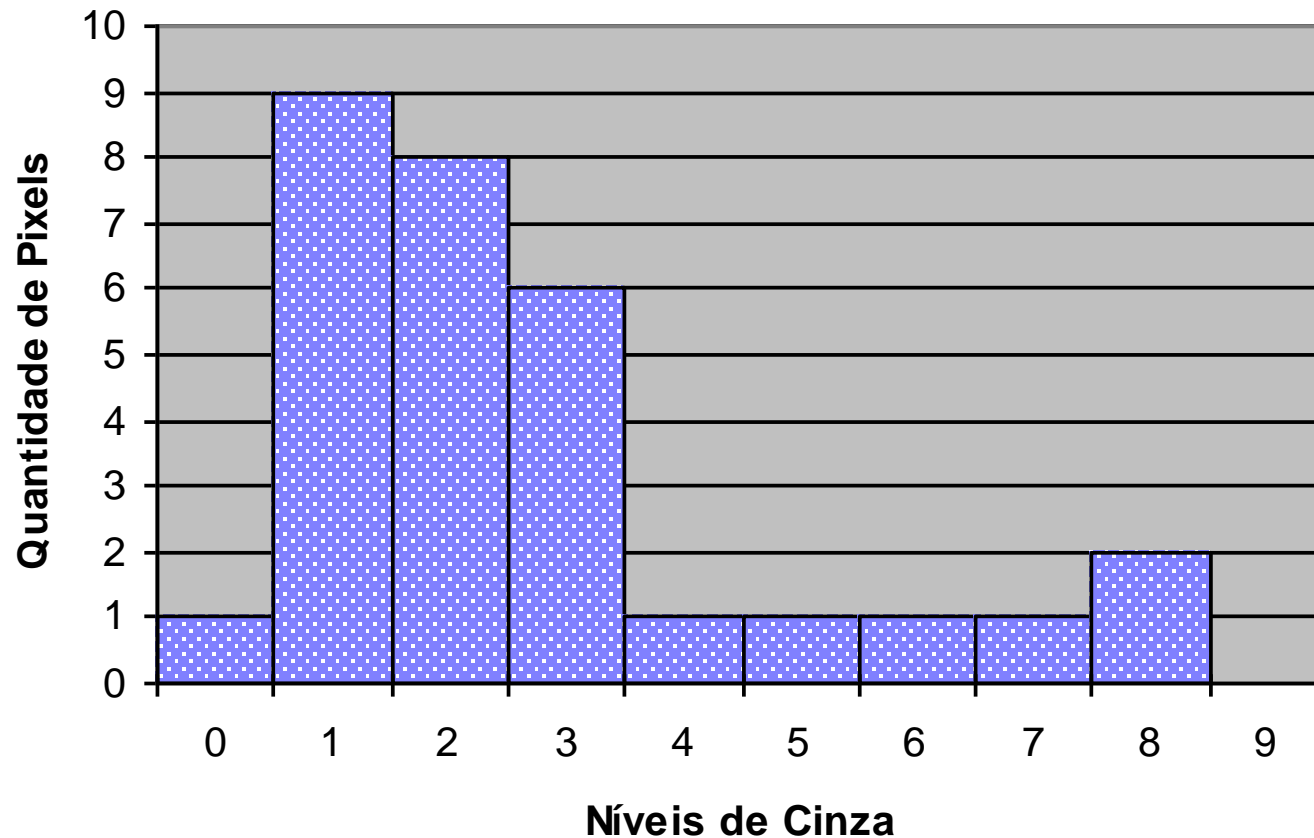
Questões

- O que é histograma de uma imagem?

- **Histograma de uma imagem**

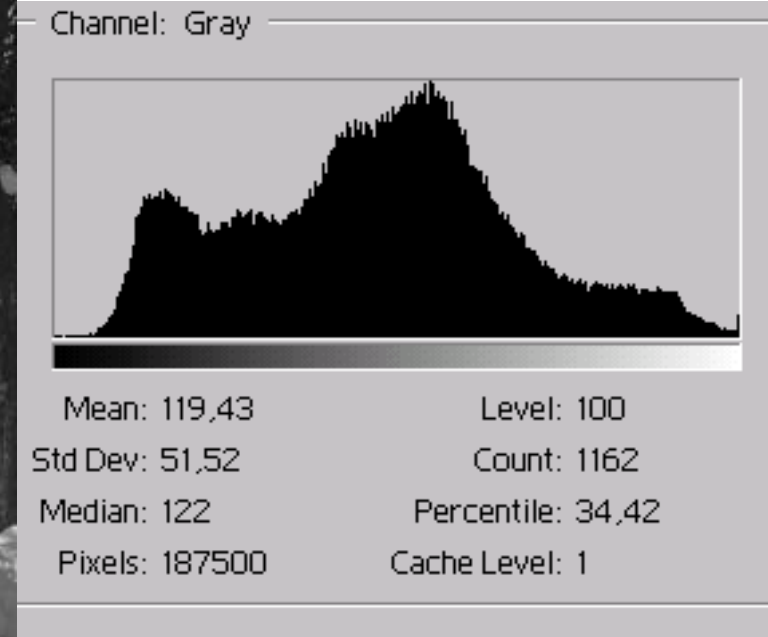
- Função que fornece a frequência de cada nível de cor na imagem.
- Valor do histograma em um nível de cinza
 - ▶ $H(k)$ ▶ quantidade de pixels da imagem com aquele nível de cinza.
- Útil para alterações globais na imagem.
- Impossível aplicá-lo em processamentos que necessitem de conhecimento sobre a localização de pixels.

Exemplo de Histograma de uma imagem



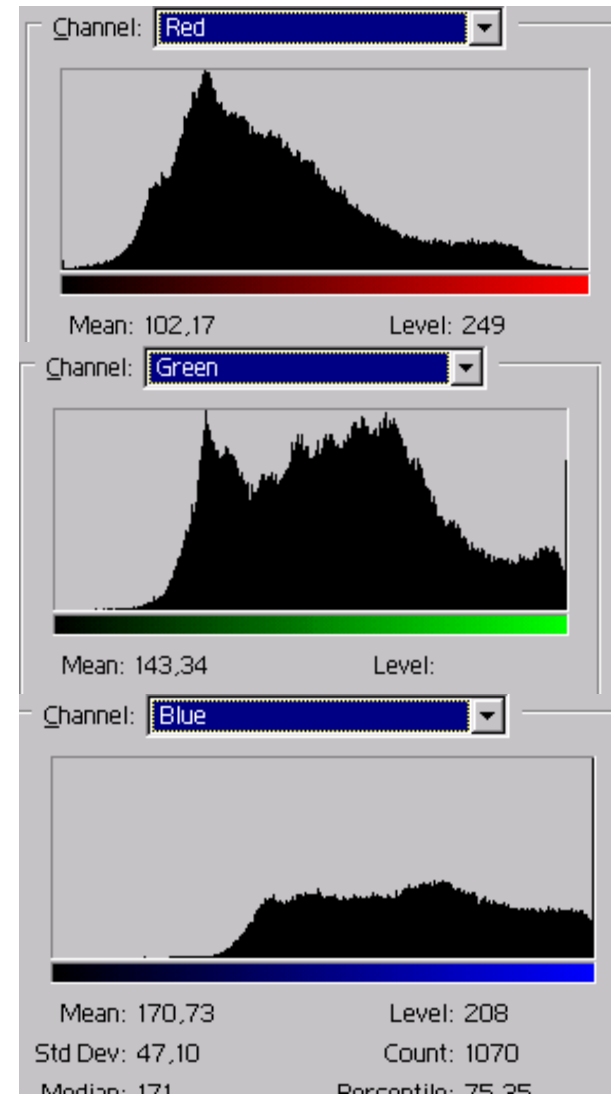
Processamento de Baixo Nível

Exemplo de Histograma de uma imagem



Processamento de Baixo Nível

Exemplo de Histograma de uma imagem colorida



Histograma de uma imagem

Algoritmo

Histograma de uma imagem

Partes principais:

- Criar um vetor de inteiros (aqui chamado H).
Tamanho: quantidade de cores da imagem (no nosso exemplo: 256)
 - `Int H[256]`
- Percorrer a matriz de pixels (dois *loops*)
 - Ler a cor do pixel
 - Somar 1 na posição do vetor que tem aquela cor: $H[Cor] = H[Cor] + 1$

Histograma de uma imagem

Várias técnicas de processamento de baixo nível são aplicadas com base no histograma da imagem e/ou acarretam alterações no histograma.

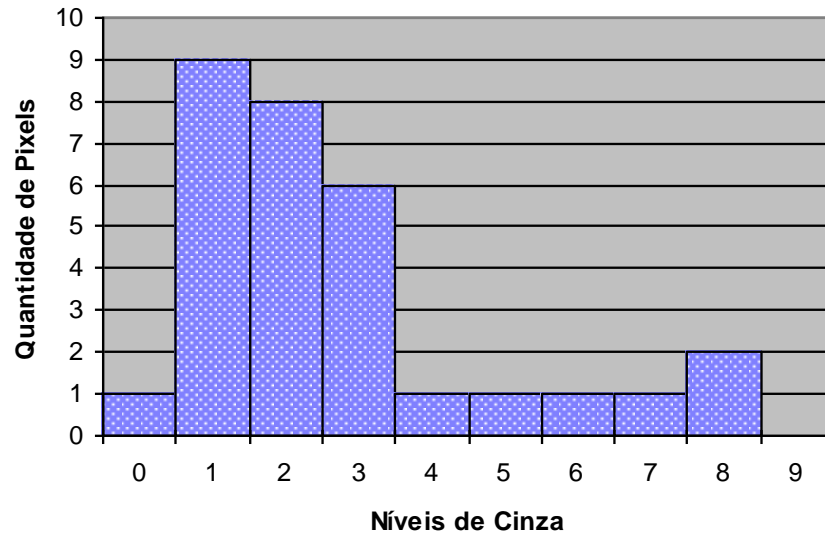
- **Alterações globais no brilho**

- Para tornar uma imagem mais clara ou mais escura ► soma ou subtração de uma constante em todos os pixels da imagem.
- Acarreta **alteração no histograma**

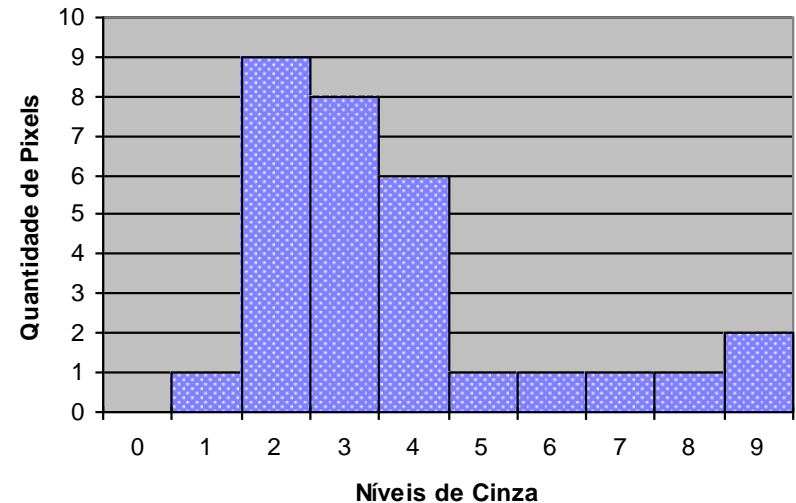
Processamento de Baixo Nível

- Alterações globais no brilho

Histograma antes da alteração no brilho

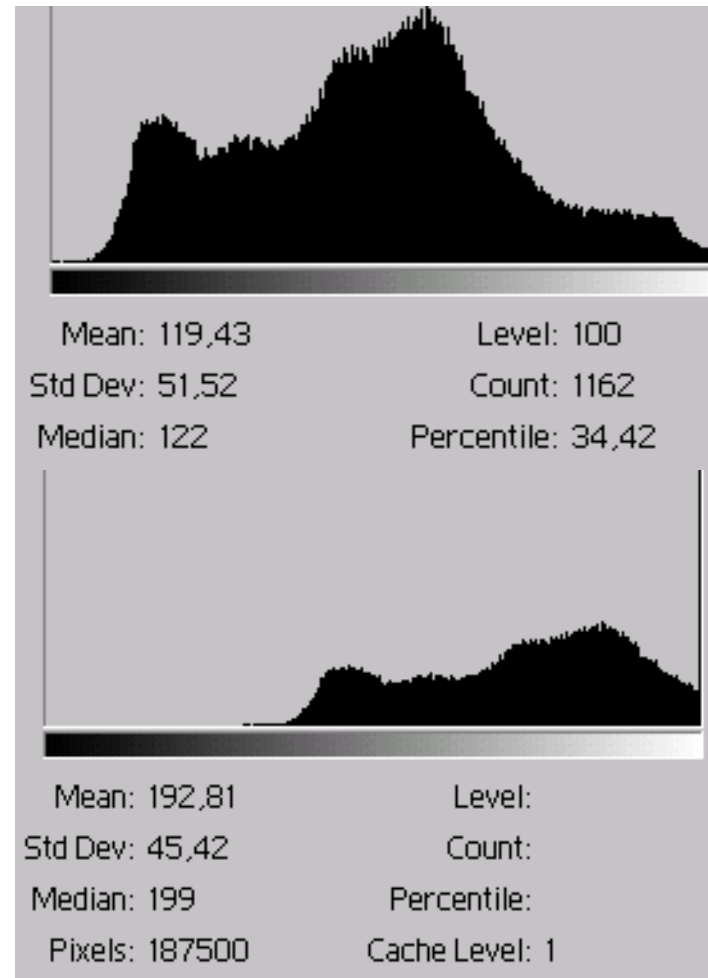


Histograma com alteração na intensidade no brilho



Processamento de Baixo Nível

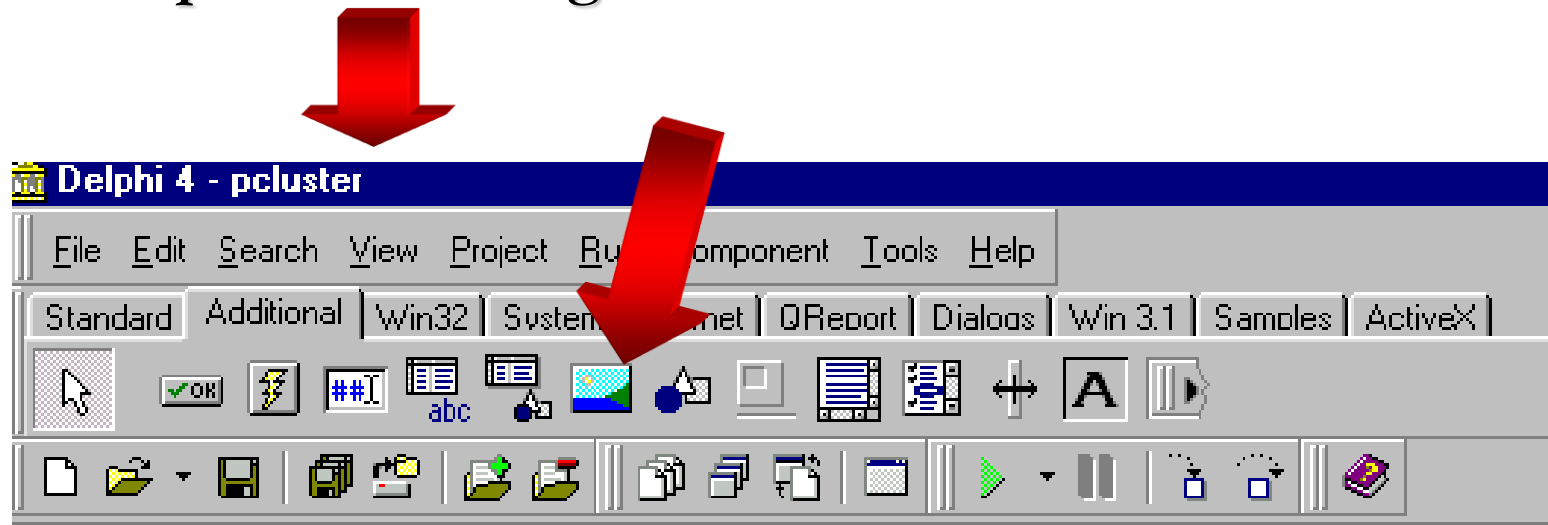
Alterações globais no brilho



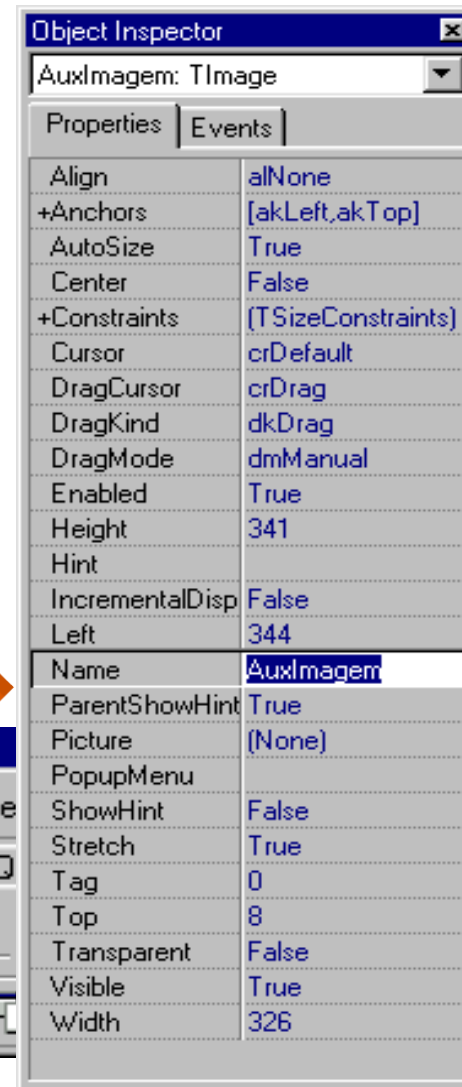
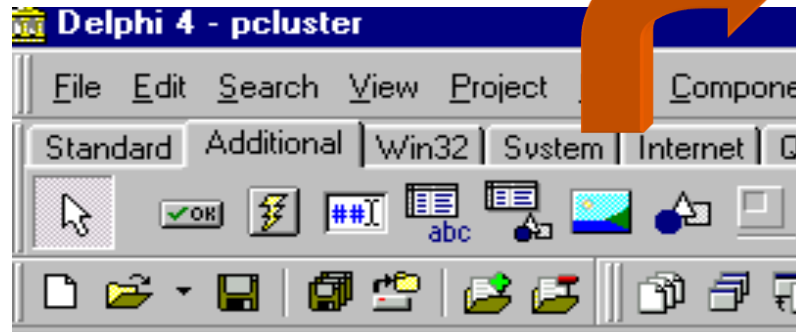
Aprendendo a manipular uma imagem

- ⇒ *Como manipular uma imagem?*
- ⇒ *Exemplo em linguagem Delphi*

⇒ **Componente TImage**

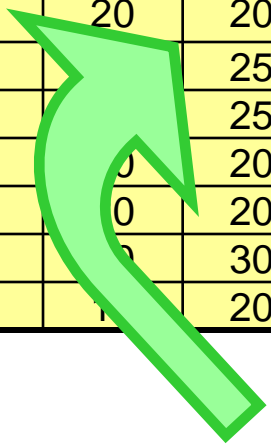


⇒ **Componente
TImage**



Implementação

⇒ Componente TImage



0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
10	10	0	0	0	0	0	0	0	0
10	20	20	10	0	0	0	0	0	0
10		25	25	25	20	0	0	0	0
20		25	25	25	30	30	30	0	0
20		20	20	30	40	40	50	40	0
15		20	30	40	40	50	80	80	0
30		30	30	30	50	50	50	50	0
10		20	20	20	40	40	40	50	10

```
Imagem.canvas.pixels[coluna,linha] := numero;
```

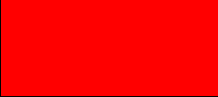







⇒ *Como são formadas as cores?*

⇒ *Combinação de três canais: R, G, B*

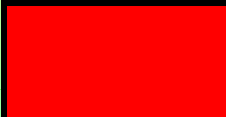




⇒ *R = RED (vermelho)*

⇒ *G = GREEN (verde)*

⇒ *B = BLUE (azul)*


Cor	R	G	B
	255	0	0
	0	255	0
	0	0	255
	250	125	50
	250	20	200
	100	100	100
	0	0	0
	255	255	255

⇒ *Como são formadas as cores?*

Cor	R	G	B
	255	0	0
	0	255	0
	0	0	255
	250	125	50
	250	20	200

Em linguagem Delphi:

```
Imagem.canvas.pixels[col,lin] :=  
RGB(250,20,200);
```



⇒ *Para percorrer a imagem: trecho de repetição.*

⇒ Exemplo:

```
for linha := 1 to QuantLinhas do
  for coluna := 1 to QuantColunas do
    begin
      valorpixel := GetRValue(Imagem.canvas.pixels[coluna,linha]);
      valorpixel := valorpixel + 10;
      Imagem.canvas.pixels[coluna,linha] :=
        RGB(valorpixel, valorpixel, valorpixel);
    end;
```

Implementação

```
for linha := 1 to QuantLinhas do
  for coluna := 1 to QuantColunas do
    begin
      valorpixel := GetRValue(Imagem.canvas.pixels[coluna,linha]);
      valorpixel := valorpixel + 10;
      Imagem.canvas.pixels[coluna,linha] :=
        RGB(valorpixel, valorpixel, valorpixel);
    end;
```

0	0	0	0	0	0
0	0	0	0	0	0
10	10	0	0	0	0
10	20	20	10	0	0
10	20	25	25	25	20
20	20	25	25	25	30
20	20	20	20	30	40
15	20	20	30	40	40
30	30	30	30	30	50
10	10	20	20	20	40

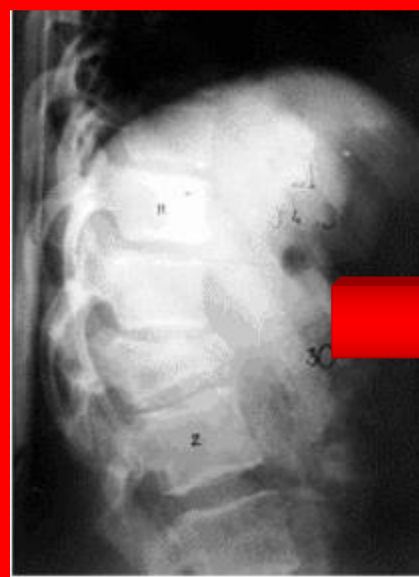


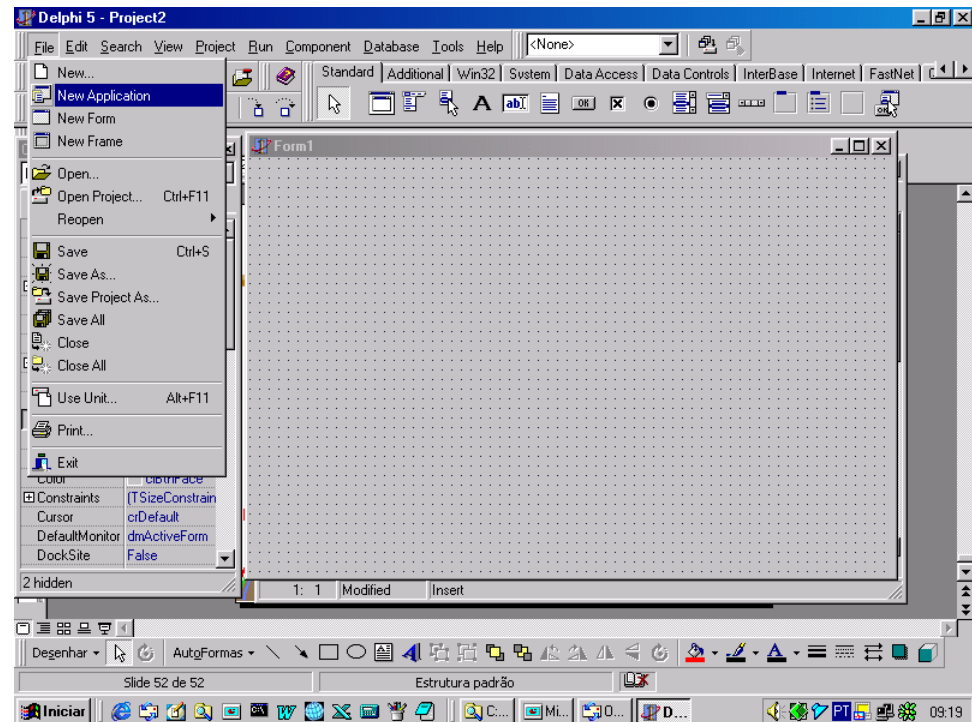
Imagem original

10	10	10	10	10
10	10	10	10	10
20	20	10	10	10
20	30	30	20	10
20	30	35	35	35
30	30	35	35	35
30	30	30	30	40
25	30	30	40	50
40	40	40	40	40
20	20	30	30	30

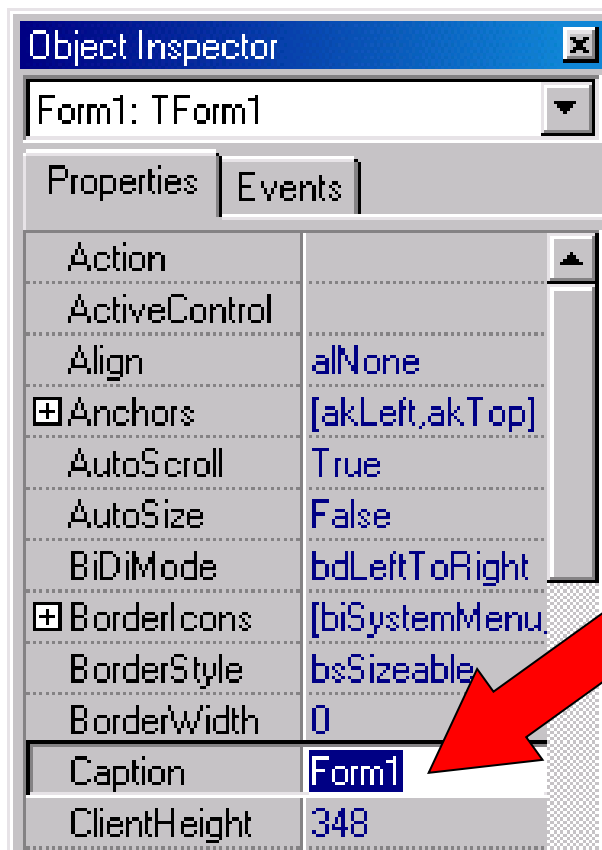


Imagem mais clara

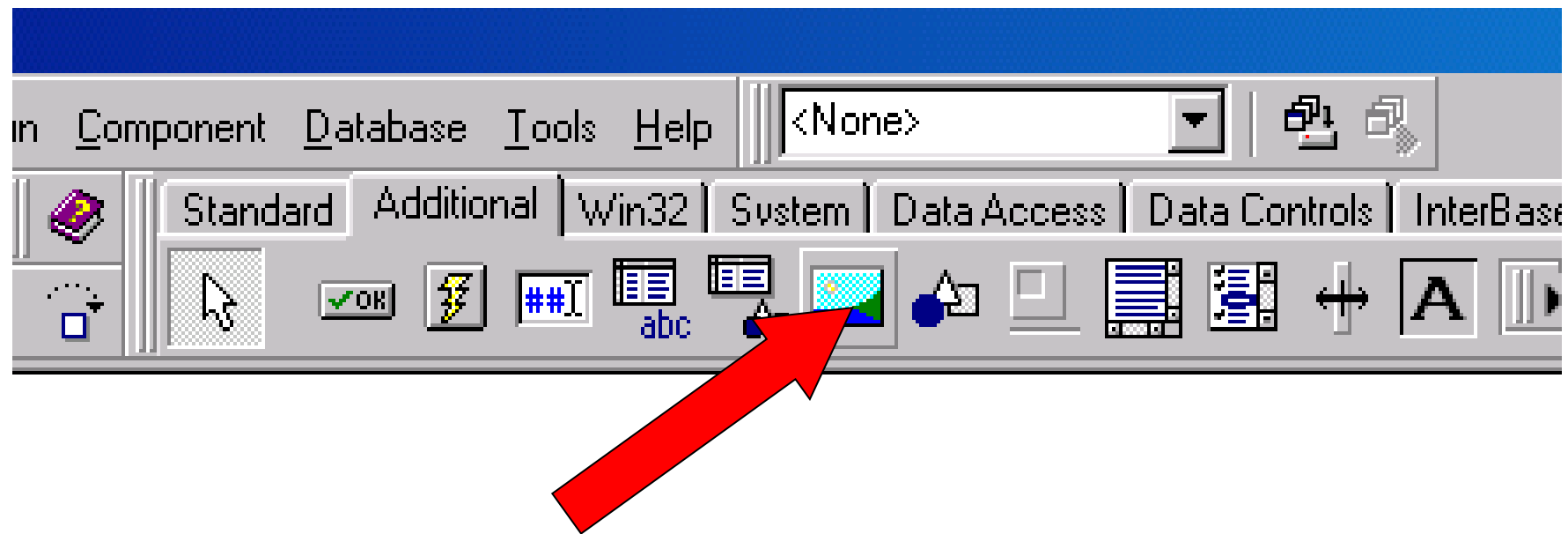
- Em Delphi:
 - Abrir uma nova aplicação: File/New Application
 - Salvar a Unit como *principal.pas*: File/Save
 - Salvar o projeto com o nome *prog1.dpr*: File/Save



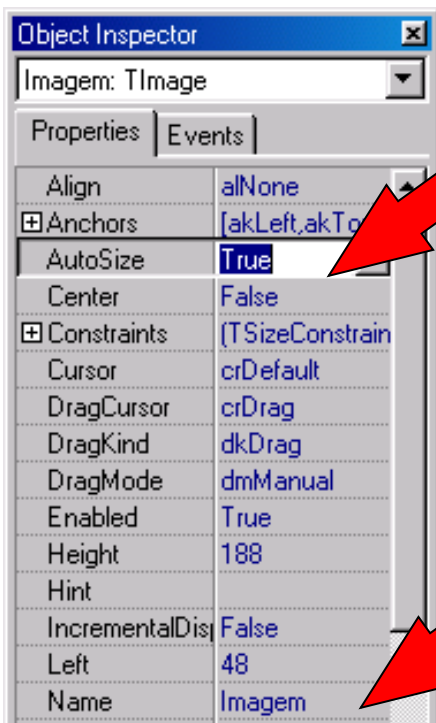
- Em Delphi:
 - Mudar o nome do Formulário para *Principal*



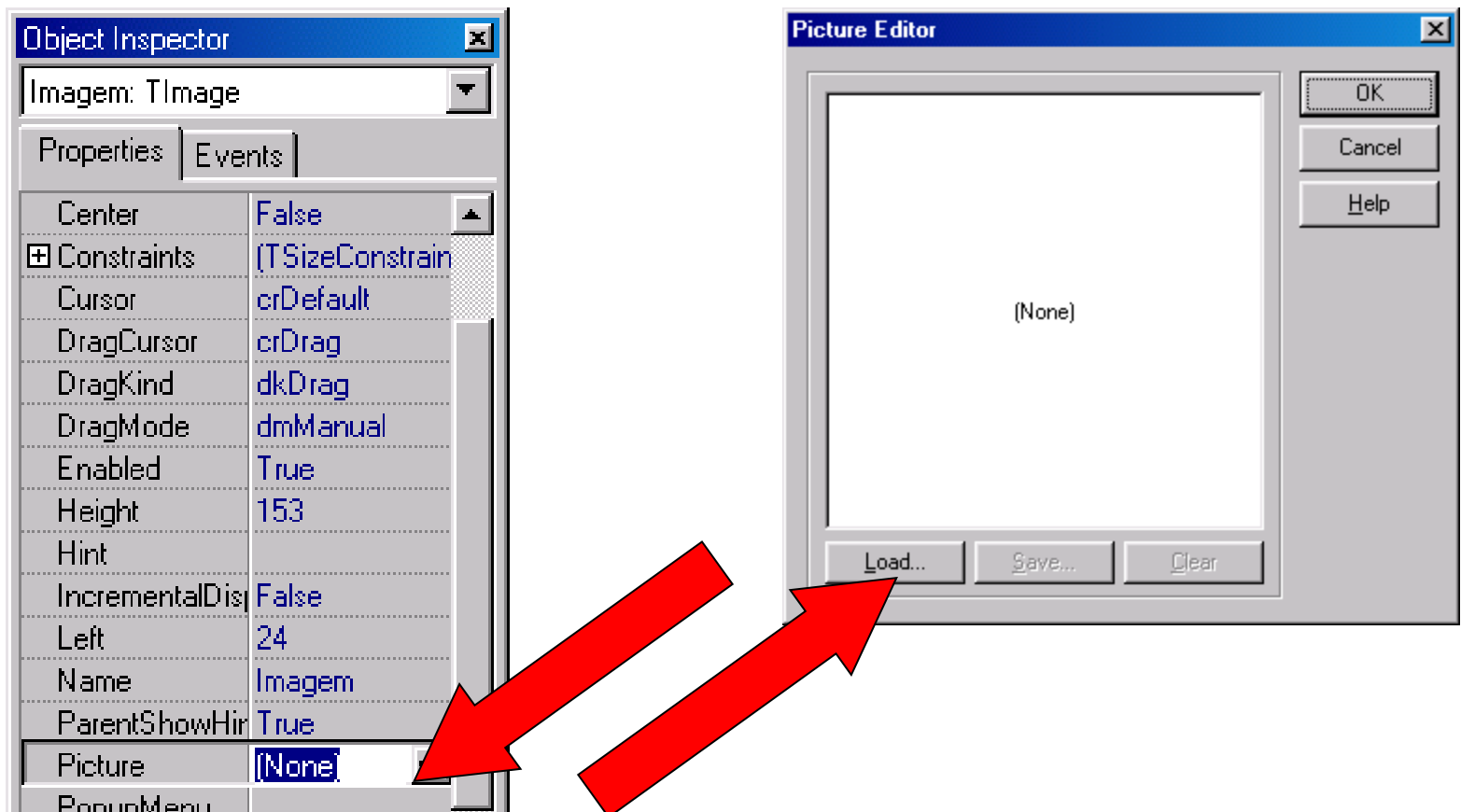
- Em Delphi:
 - Inserir um componente Timage: Barra de Ferramentas Additional



- Em Delphi:
 - Mudar o nome do componente para Imagem
 - Mudar o atributo AutoSize para True

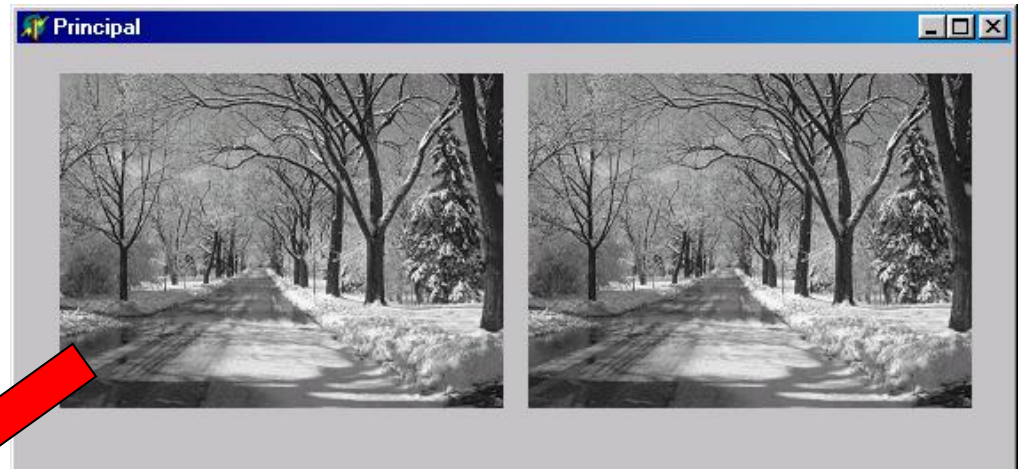
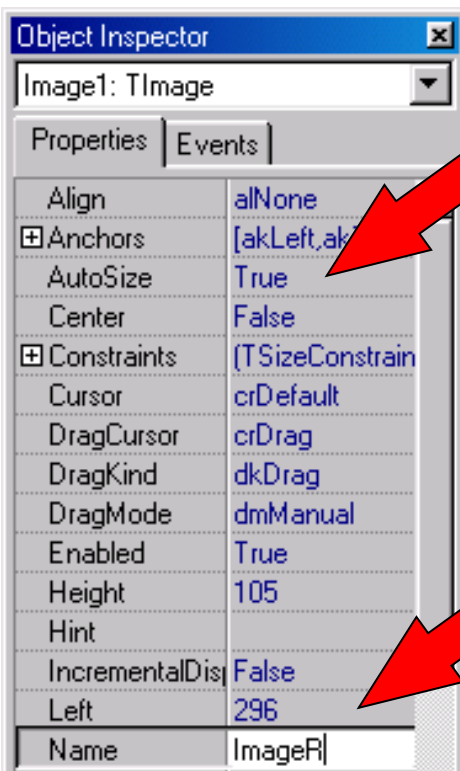


- Em Delphi:
- Carregar a imagem foto1.bmp



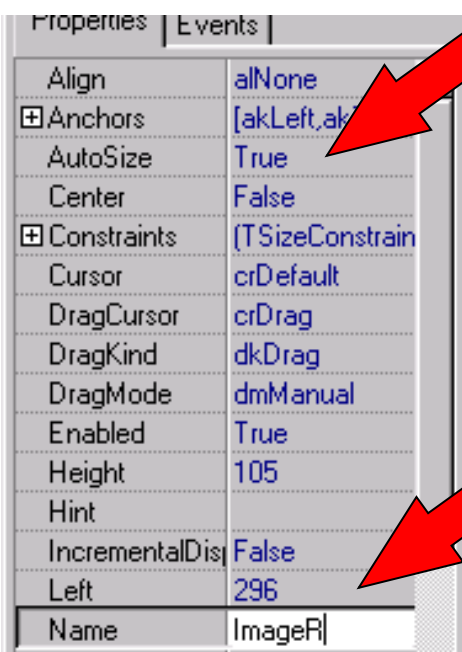
•Em Delphi:

- Inserir outro componente Timage. Mudar o nome para ImagemR e mudar o atributo Autosize para True.
- Carregar a mesma imagem no componente.



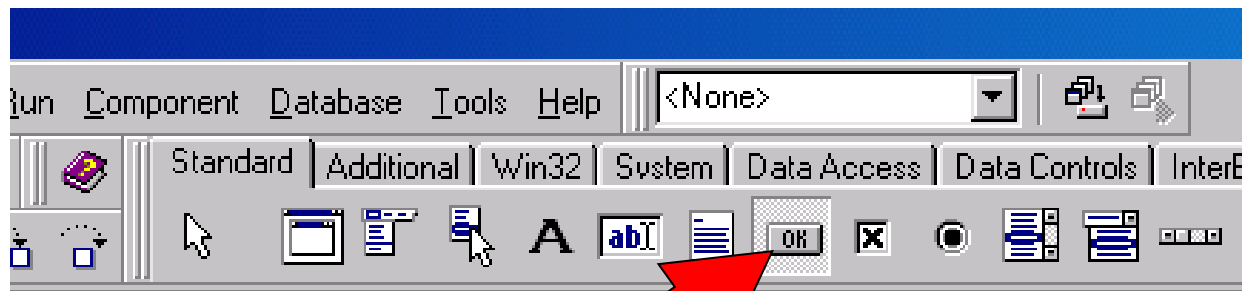
- Em Delphi:

- Inserir outro componente Timage. Mudar o nome para ImagemR e mudar o atributo Autosize para True.
- Carregar a mesma imagem no componente.

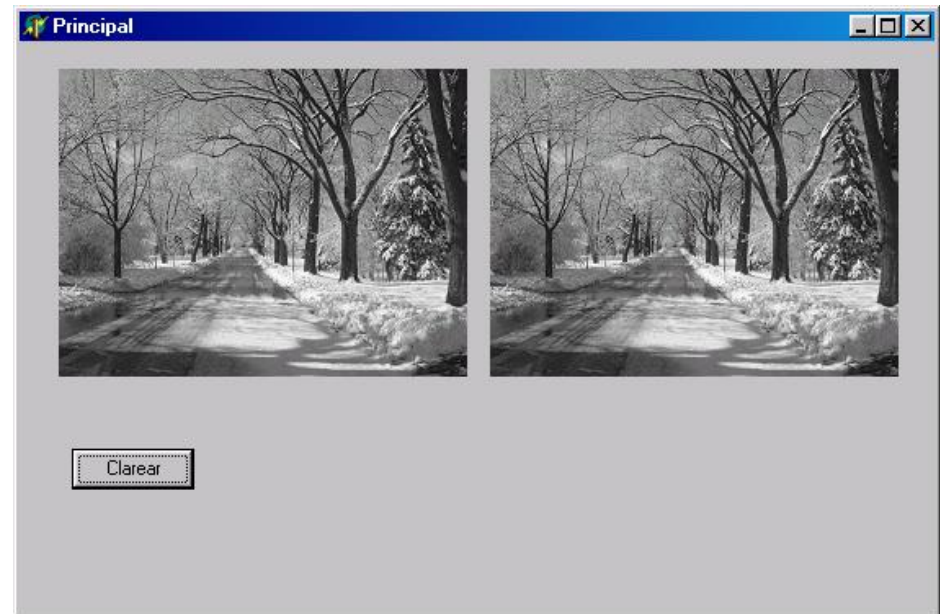
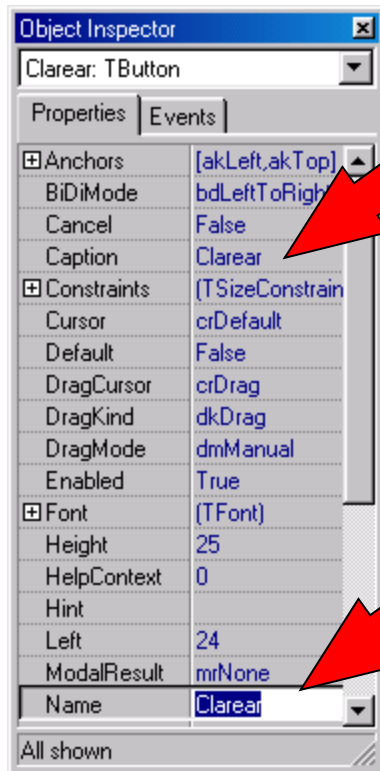


- Em Delphi:

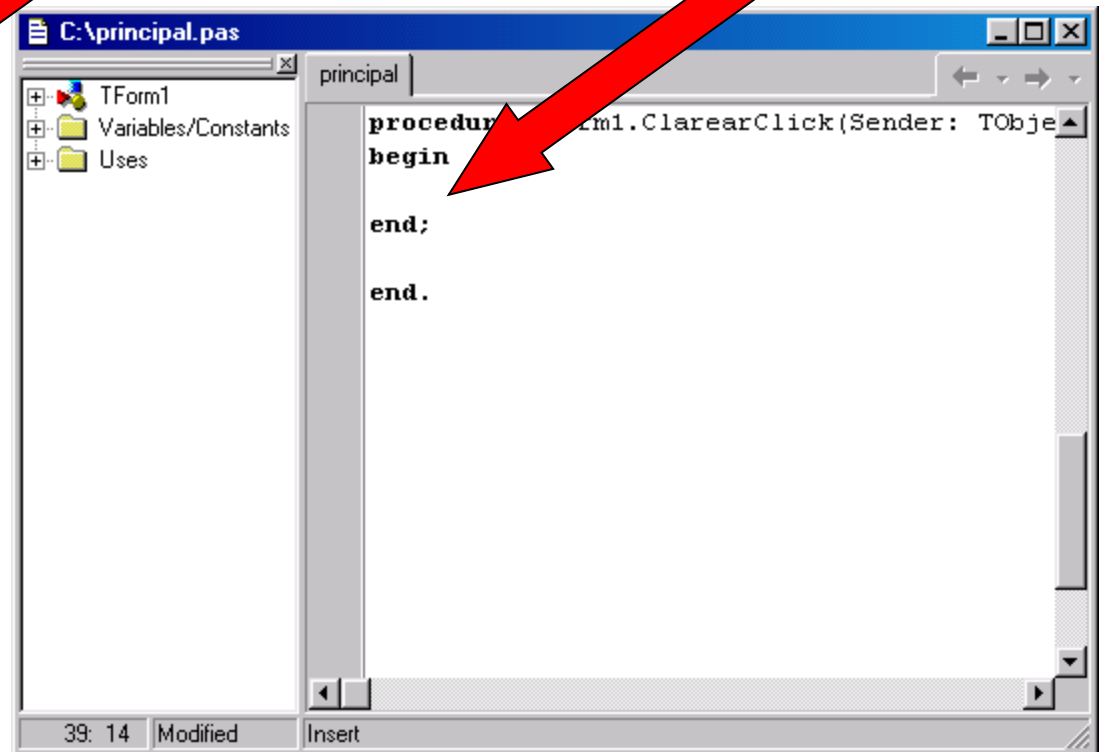
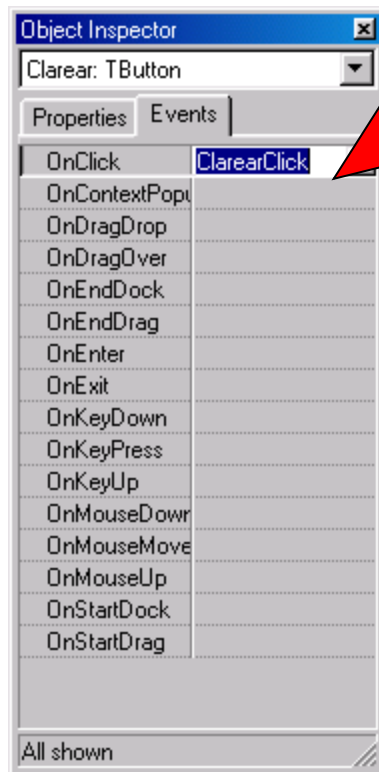
- Inserir um Botão: Barra de Ferramentas Standard



- Em Delphi:
 - Mudar o Nome e o Caption para *Clarear*



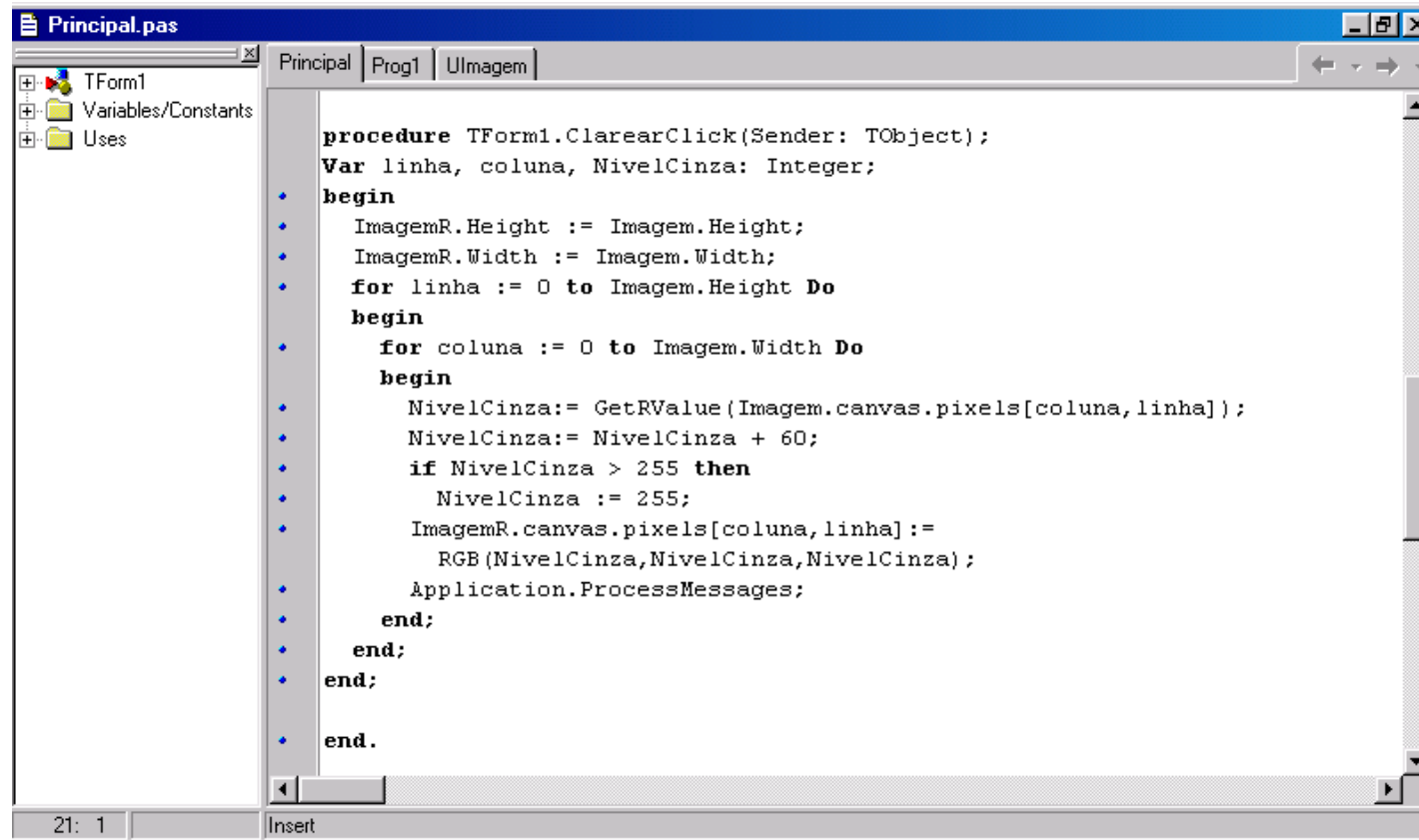
- Em Delphi:
 - Clique duplo no Evento OnClick do Botão *Clarear*



- Em Delphi:
- Inserir o seguinte código no local indicado

```
procedure TForm1.ClarearClick(Sender: TObject);
Var linha, coluna, NivelCinza: Integer;
begin
  ImagemR.Height := Imagem.Height;
  ImagemR.Width := Imagem.Width;
  for linha := 0 to Imagem.Height Do
  begin
    for coluna := 0 to Imagem.Width Do
    begin
      NivelCinza:= GetRValue(Imagem.canvas.pixels[coluna,linha]);
      NivelCinza:= NivelCinza + 60;
      if NivelCinza > 255 then
        NivelCinza := 255;
      ImagemR.canvas.pixels[coluna,linha]:=
        RGB(NivelCinza,NivelCinza,NivelCinza);
      Application.ProcessMessages;
    end;
  end;
end;
```

- Em Delphi:
- Tela final abaixo. Executar o programa: F9



Exercício (para entregar)

- Fazer um programa que tenha 3 funções:
 1. Carregar uma imagem
 2. Mostrar a imagem e seu histograma (pode usar funções prontas para plotar o gráfico).
 3. Transformar a imagem carregada em mais clara ou mais escura, a partir de um valor recebido como parâmetro (não pode usar funções prontas, tem que implementar a manipulação da matriz de pixels). Deve ser exibida a imagem resultante e seu histograma (usando o que foi implementado no item 2)



Computação Gráfica

Aula 3

Processamento de Imagens

Conceitos e Tecnologia

Profa. Fátima Nunes