```
(*                  Computer Graphics with Applications of Dr. Makhanov.
                                      Lab 4: Problem 10-15
                         Basic Image Processing with Mathematica *)
```

```
SetOptions[{ListPlot3D, Histogram, Plot}, ImageSize → Small];

SetOptions[EvaluationNotebook[], ShowCellLabel → False];

(* Set up your directory where you uploaded the images*)
SetDirectory["E:\\Computer Graphics\\Lab4\\Lab4"]
```

```
E:\Computer Graphics\Lab4\Lab4
```

```
(* Problem (10) Turn the gray level up by 1.5 of the Blue
 component of image mona.jpg  using ImageMultiply[image,a] *)
```

```
(* Read *)
```

```
Mona := Import["mona.jpg"]
```

```
(* RemoveAplhaChannel in case your hardware generates the forth channel *)
```

```
(* Color separate *)
```

[redacted]

```
(* Turn up the grey level *)
```

```
{MonaBnew = ImageMultiply[MonaB, 1.5]}
```

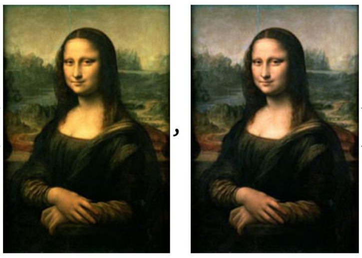
```
(* Combine into a new image *)
```

[redacted]

{}

```
(* Compare *)
```

[redacted]

{}

```
(* ImageApply applies a function f to each pixel in the image *)
```

```
(* Problem (11)  Apply a non-linear point operation given by
   f1[x_]:=1/2(1+1/Tan[a*Pi/2]*Tan[a Pi(x-1/2)])
   to mona.jpg  *)
```

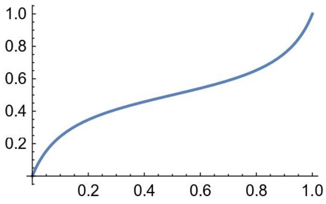
```
(* Define the transformation function *)
```

```
a := 0.8

f1[x_] := 1 / 2 (1 + 1 / Tan[a * Pi / 2] * Tan[a Pi (x - 1 / 2)])
```

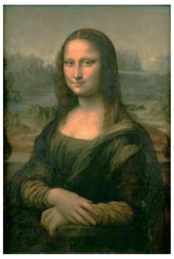
```
(* Show f1[x]  *)
```

```
Plot[f1[x], {x, 0, 1}]
```



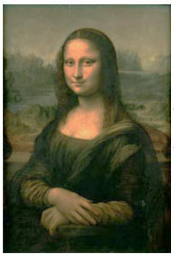

```
(* Apply f1 to the image *)
```

```
{MonaNew1 = ImageApply[f1, Mona]}
```

{}

```
(* Compare *)
```

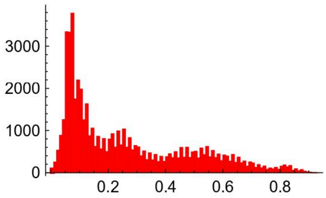
```
{Mona, MonaNew1}
```

{, }

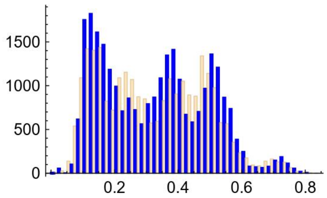
```
(* Convert to the gray level and compare histograms of Mona and MonaNew1  *)
```

[redacted]

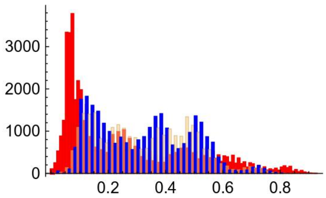
[redacted]



```
HMnew = Histogram[MonaNew1F, {0.01}, ChartStyle → {Blue, Opacity[0.5]}]
```



```
Show[HM, HMnew]
```



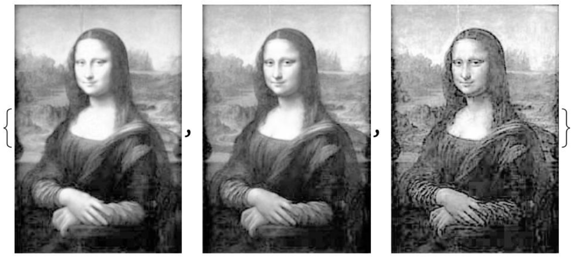
```
(* HistogramTransform performs a histogram equalization. The
 equalization can be used a to enhance features of a picture,
or use a particular display device more efficiently. The histogram of the
 equalized image is almost flat which means that every gray is equally important *)
```

```
(* Problem (12) Apply HistogramTransform to mona.jpg  *)
```

```
(* Apply HistogramTransform to mona.jpg  *)
```

```
{MonaEq = HistogramTransform[Mona]}
```

```
{MonaRE, MonaGE, MonaBE} = ColorSeparate[MonaEq]
```



```
(* Equalize the Green Matrix of Mona Lisa and apply function
f2[x_]:=x+x(1-x)*a to the Blue Matrix. a=1. Compose and save the resulting image *)
```

```
(* Separate *)
```

[redacted]

```
(* Equalize the Green Matrix  *)
```

[redacted]

{  }

```
(* Apply f2 to the Blue Matrix *)
```

[redacted]

1

[redacted]

```
{MonaBnew = ImageApply[f2, MonaB]}
```

{  }

```
(* Combine the matrices back into the color image  *)
```

[redacted]

{  }

```
(* Adding, subtracting and multiplying images *)
```

```
(* Subtracting *)
```

```
(* Problem (12). Find the difference between pill1.jpg and pill2.jpg *)
```

```
P1 = Import[ "pill1.jpg"]
```

```
P2 = Import[ "pill2.jpg"]
```

```
(* Convert to the gray level *)
```

```
{PG1 = ColorConvert[P1, "Grayscale"], PG2 = ColorConvert[P2, "Grayscale"]}
```

{, }

```
(* Let us depict the difference image. The light areas correspond
 to the motion whereas the black color indicates that no changes
 have occurred during the time period between the two images *)
```

```
DP1P2 = ImageDifference[PG1, PG2]
```

```
(* ColorNegate  *)
```

```
Result = ColorNegate[DP1P2]
```

```
(* Problem (13). Use ImagePartition and ImageDifference to find the difference
 between the pirate images in pirat.gif Use the solution of Problem 12 *)
```

```
(* Apply RemoveAplhaChannel in case your hardware generates the forth channel *)
```

```
{Pirat1 = Import["pirat.gif"]}
```

{}

```
{Pirat = RemoveAlphaChannel[Pirat1]}
```

{}

```
DimP = ImageDimensions[Pirat]
```

{300, 180}

```
xPir = DimP[[1]]
```

300

```
yPir = DimP[[2]]
```

180

```
(* Split vertically at x=300/2  *)
```

```
Pirat = ImagePartition[Pirat, {xPir / 2, yPir}]
```

{{, }}

```
Part1 = Pirat[[1]][[1]]
```

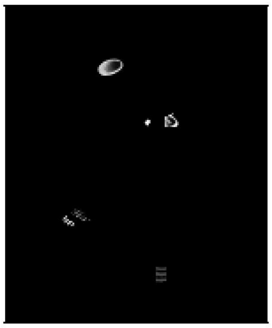
```
Part2 = [redacted]
```

[redacted]

[redacted]

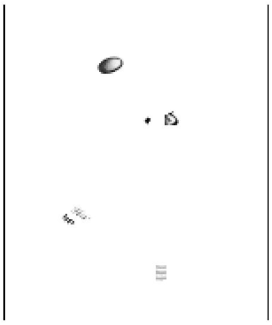
```
(* Find the difference *)
```

[redacted] ImageDifference [redacted]



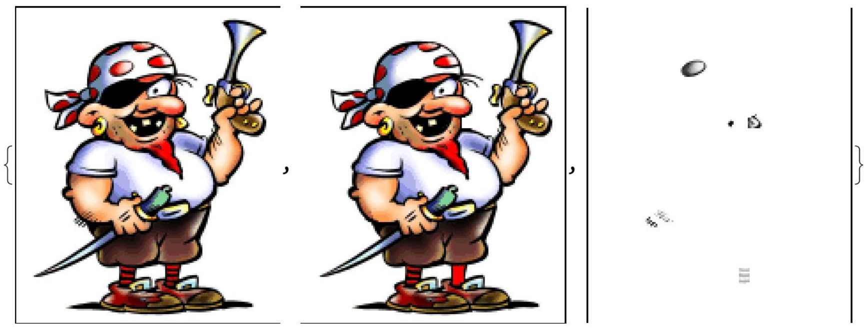
```
(* ColorNegate *)
```

[redacted]



```
(* Compare *)
```

[redacted]



```
(* Answer: The differences are (1) Hair on the left arm (2) Red dot on the bandana
  (3) Left earring (4) Two vs three teeth (5) Stripes on the left sock *)
```

```
(* Problem (14) Analyze the code to morph two images. Bush.jpg and Arnold.jpg" *)
```

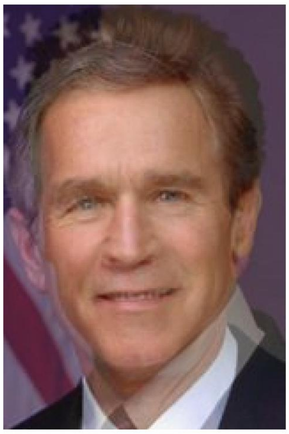
```
{Bu = Import["bush.jpg"]}
```

```
Ar := Import["arnold.jpg"]

Morph[t_] := Blend[{Bu, Ar}, t]

Mor = Table[Morph[t], {t, 0, 1, 0.01}];(* precompute 100 frames *)
```

```
{ListAnimate[Mor, AnimationRunning → False]}
```



```
(* Problem 15. Save your Lab and start a new file Lab4_bonus.nb Errors could be fatal to
   your code. You may lose the entire Lab! Plot flowers given below use ColorFunction→
  "SunsetColors" for jazmin1 and ColorFunction→"AvocadoColors" for jazmin2
    Note the beautiful mathematical formulas representing the flowers
    The parametric variables change as follows {u,-Pi,Pi} {v,0,3.4},
 use PlotPoints→50, Mesh→None  *)
```

```
SetOptions[ParametricPlot3D, ImageSize → Small];

c := 5

r1[u_, v_] := v^(1.6) (0. + (0.01 + Abs[Sin[c u / 2]])^(1 / 3));

xj1[u_, v_] := r1[u, v] × Sin[u] * Cos[v]

yj1[u_, v_] := r1[u, v] × Cos[u] * Cos[v]

zj1[u_, v_] := 3.5 Abs[Cos[v]]^(0.7) (2 + Sin[c u / 2]^2)

jazmin1[u_, v_] := {xj1[u, v], yj1[u, v], zj1[u, v]}


jazmin1[1., 1.]
{0.385262, 0.247374, 5.364}
```

[redacted] "SunsetColors"

```
c := 9

r2[u_, v_] := v^(1.6) (0. + (0.01 + Abs[Sin[c u / 2]] )^(1 / 3))

xj2[u_, v_] := r2[u, v] × Sin[u] * Cos[v]

yj2[u_, v_] := r2[u, v] × Cos[u] * Cos[v]

zj2[u_, v_] := 3.5 Abs[Cos[v]]^(0.7) (2 + Sin[c u / 2]^2)

jazmin2[u_,v_]:={xj2[u,v],yj2[u,v],zj2[u,v]}
```

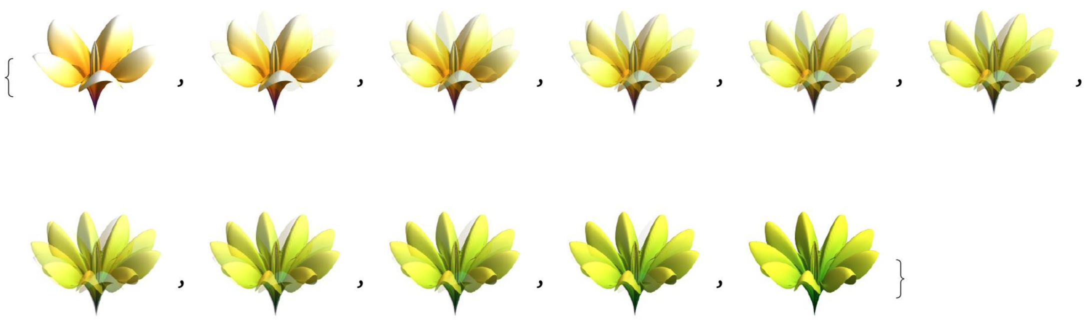
[redacted] "AvocadoColors"

```
(* Blend function *)
```

[redacted]

```
(* Generate table of 10 frames. Careful! If you set up too many frames the
   operation will hang your code. Wait until the table has been evaluated.
    It could take 1-5 min. Use ListAnimate to morph set AnimationRunning→False  *)
```

[redacted] `{t, 0, 1, 0.1}`



[redacted]