İTÜ

Department of Computer Engineering

BLG 458E
Functional Programming
Color Manipulation
Project Report

ID	*Name*	*Surname*
150140126	*Emre*	*Reyhanlıoğlu*

Color Manipulation

PART 1

First function of the project is **rgb2hsv** which takes a RGB triple and converts into a HSV triple. My function prototype is below;

rgb2hsv :: (Float,Float,Float) -> (Float,Float,Float)

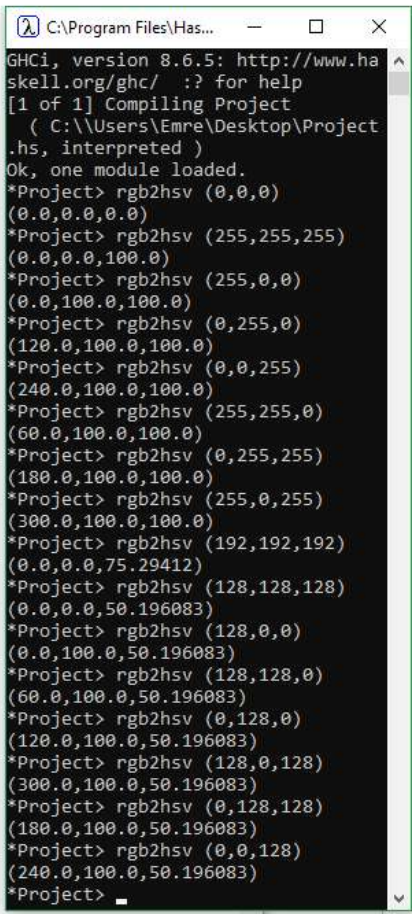
Note that; R,G and B input values are between 0 to 255.

I have written some helper functions to make the code modular.First of all, I wrote “maxThree” and “minThree“ functions to find the max/min of three values. Then I wrote calcDelta, calcHue and calcSat functions to calculate necessery parameters in the formula.

You can see the correctness of my test results about this function below;

RGB to HSV color table

Color	Color name	Hex	(R,G,B)	(H,S,V)
	Black	#000000	(0,0,0)	(0°,0%,0%)
	White	#FFFFFF	(255,255,255)	(0°,0%,100%)
	Red	#FF0000	(255,0,0)	(0°,100%,100%)
	Lime	#00FF00	(0,255,0)	(120°,100%,100%)
	Blue	#0000FF	(0,0,255)	(240°,100%,100%)
	Yellow	#FFFF00	(255,255,0)	(60°,100%,100%)
	Cyan	#00FFFF	(0,255,255)	(180°,100%,100%)
	Magenta	#FF00FF	(255,0,255)	(300°,100%,100%)
	Silver	#C0C0C0	(192,192,192)	(0°,0%,75%)
	Gray	#808080	(128,128,128)	(0°,0%,50%)
	Maroon	#800000	(128,0,0)	(0°,100%,50%)
	Olive	#808000	(128,128,0)	(60°,100%,50%)
	Green	#008000	(0,128,0)	(120°,100%,50%)
	Purple	#800080	(128,0,128)	(300°,100%,50%)
	Teal	#008080	(0,128,128)	(180°,100%,50%)
	Navy	#000080	(0,0,128)	(240°,100%,50%)



PART 2

Second function of the project is **hsv2rgb** which takes a HSV triple and converts into a RGB triple. My function prototype is below;

hsv2rgb :: (Float,Float,Float) -> (Float,Float,Float)

Note that; H input should be between 0 to 360, S and V inputs should be between 0 to 100.

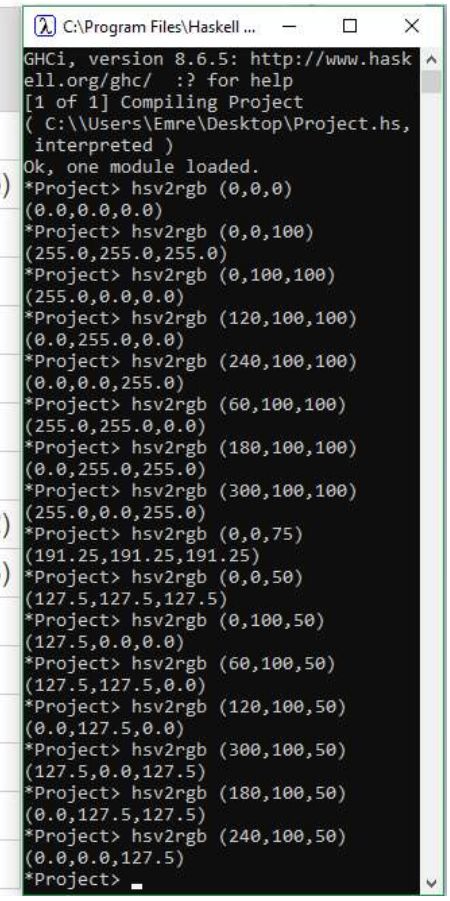
I have written one helper function named “calcRGB” to calculate the R'G'B' parameters according to formula. In the hsv2rgb function, I have completed the formula below;

$$(R,G,B) = ((R'+m)\times 255,\ (G'+m)\times 255,\ (B'+m)\times 255)$$

You can see the correctness of my test results about this function below;

HSV to RGB color table

Color	Color name	(H,S,V)	Hex	(R,G,B)
	Black	(0°,0%,0%)	#000000	(0,0,0)
	White	(0°,0%,100%)	#FFFFFF	(255,255,255)
	Red	(0°,100%,100%)	#FF0000	(255,0,0)
	Lime	(120°,100%,100%)	#00FF00	(0,255,0)
	Blue	(240°,100%,100%)	#0000FF	(0,0,255)
	Yellow	(60°,100%,100%)	#FFFF00	(255,255,0)
	Cyan	(180°,100%,100%)	#00FFFF	(0,255,255)
	Magenta	(300°,100%,100%)	#FF00FF	(255,0,255)
	Silver	(0°,0%,75%)	#C0C0C0	(192,192,192)
	Gray	(0°,0%,50%)	#808080	(128,128,128)
	Maroon	(0°,100%,50%)	#800000	(128,0,0)
	Olive	(60°,100%,50%)	#808000	(128,128,0)
	Green	(120°,100%,50%)	#008000	(0,128,0)
	Purple	(300°,100%,50%)	#800080	(128,0,128)
	Teal	(180°,100%,50%)	#008080	(0,128,128)
	Navy	(240°,100%,50%)	#000080	(0,0,128)



PART 3

In this part, **name2rgb** function takes a HTML color name as a string (i.e. #000000) and converts into a RGB triple.My function prototype is below;

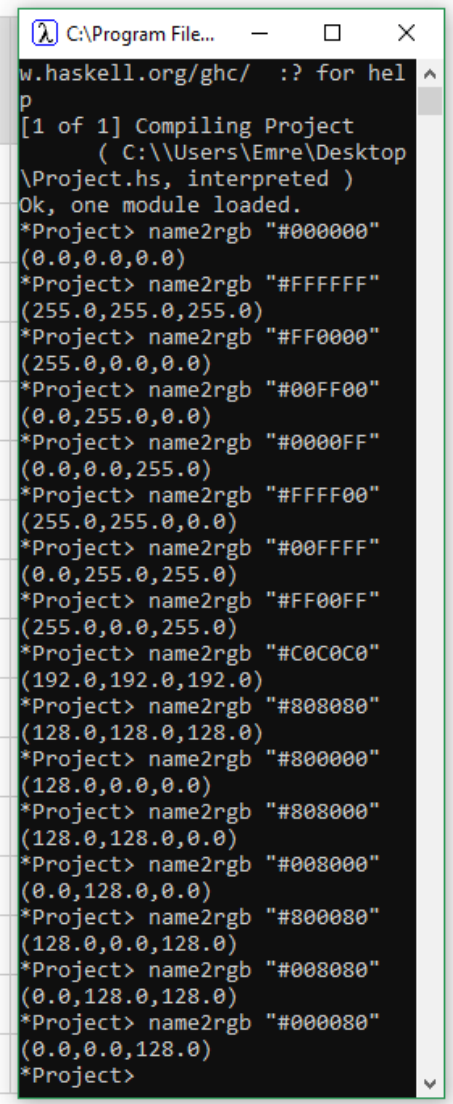
name2rgb :: String -> (Float,Float,Float)

Note that; color name should start with '#' character

I have written one helper function named “hexToInt” to convert hexadecimal numbers to integers, then I used it with drop and take functions in my “name2rgb” function.

You can see the correctness of my test results about this function below;

Color	Color name	Hex	(R,G,B)
	Black	#000000	(0,0,0)
	White	#FFFFFF	(255,255,255)
	Red	#FF0000	(255,0,0)
	Lime	#00FF00	(0,255,0)
	Blue	#0000FF	(0,0,255)
	Yellow	#FFFF00	(255,255,0)
	Cyan	#00FFFF	(0,255,255)
	Magenta	#FF00FF	(255,0,255)
	Silver	#C0C0C0	(192,192,192)
	Gray	#808080	(128,128,128)
	Maroon	#800000	(128,0,0)
	Olive	#808000	(128,128,0)
	Green	#008000	(0,128,0)
	Purple	#800080	(128,0,128)
	Teal	#008080	(0,128,128)
	Navy	#000080	(0,0,128)



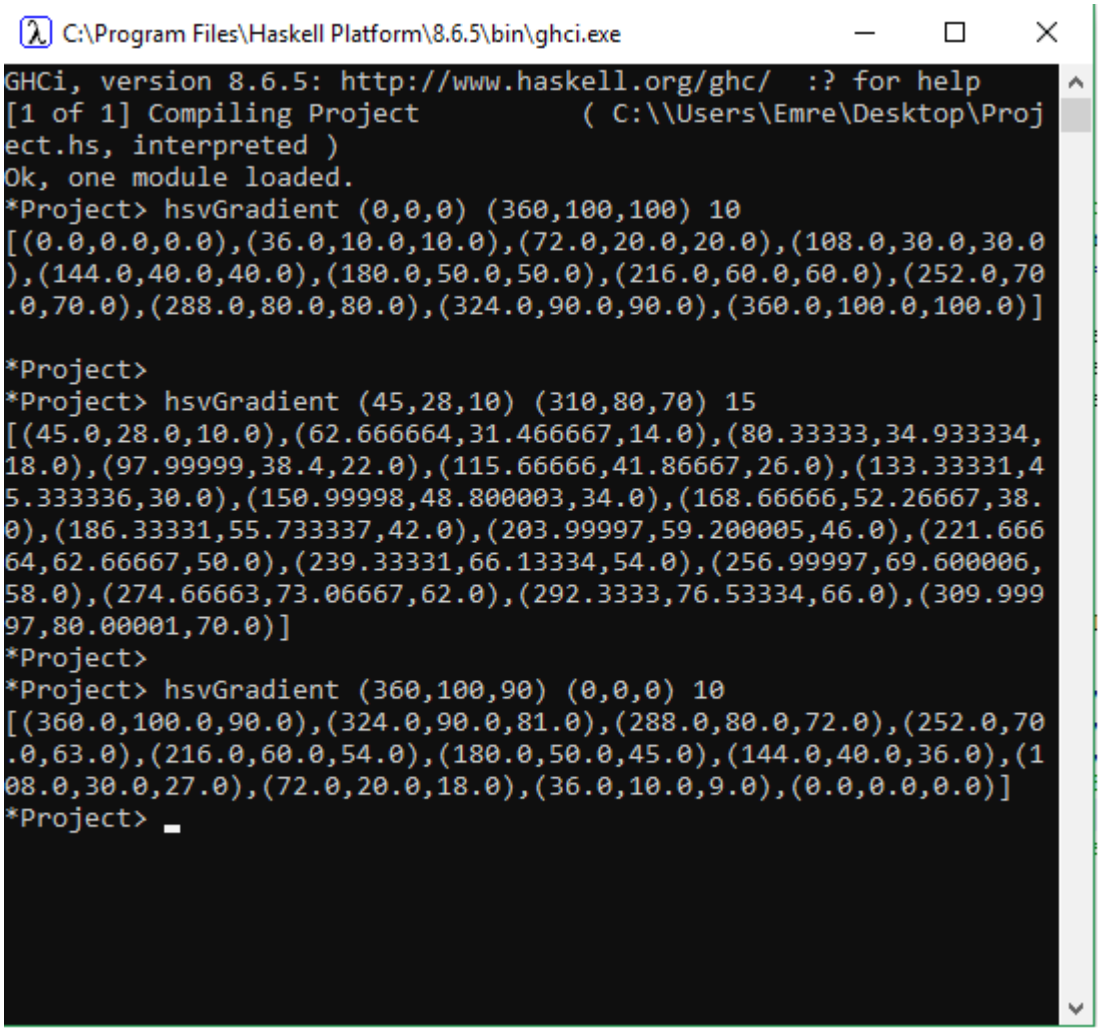
PART 4

In this part, **hsvGradient** function takes a starting HSV color, an ending HSV color, and a number of steps, and it returns a gradient that starts from the starting color and reaches the ending color in the given number of steps.My function prototype is below;

hsvGradient :: (Float,Float,Float) -> (Float,Float,Float) -> Float -> [(Float,Float,Float)]

Firstly, I have calculated intermediate values for H, S and V gradients as arrays, then I used zip3 function to combine them as triples.

You can see some of my test results about this function below;



```
C:\Program Files\Haskell Platform\8.6.5\bin\ghci.exe
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Project          ( C:\\Users\Emre\Desktop\Proj
ect.hs, interpreted )
Ok, one module loaded.
*Project> hsvGradient (0,0,0) (360,100,100) 10
[(0.0,0.0,0.0),(36.0,10.0,10.0),(72.0,20.0,20.0),(108.0,30.0,30.0
),(144.0,40.0,40.0),(180.0,50.0,50.0),(216.0,60.0,60.0),(252.0,70
.0,70.0),(288.0,80.0,80.0),(324.0,90.0,90.0),(360.0,100.0,100.0)]

*Project>
*Project> hsvGradient (45,28,10) (310,80,70) 15
[(45.0,28.0,10.0),(62.666664,31.466667,14.0),(80.33333,34.933334,
18.0),(97.99999,38.4,22.0),(115.66666,41.86667,26.0),(133.33331,4
5.333336,30.0),(150.99998,48.800003,34.0),(168.66666,52.26667,38.
0),(186.33331,55.733337,42.0),(203.99997,59.200005,46.0),(221.666
64,62.66667,50.0),(239.33331,66.13334,54.0),(256.99997,69.600006,
58.0),(274.66663,73.06667,62.0),(292.3333,76.53334,66.0),(309.999
97,80.00001,70.0)]
*Project>
*Project> hsvGradient (360,100,90) (0,0,0) 10
[(360.0,100.0,90.0),(324.0,90.0,81.0),(288.0,80.0,72.0),(252.0,70
.0,63.0),(216.0,60.0,54.0),(180.0,50.0,45.0),(144.0,40.0,36.0),(1
08.0,30.0,27.0),(72.0,20.0,18.0),(36.0,10.0,9.0),(0.0,0.0,0.0)]
*Project>
```

PART 5

In this part, **hsv2desc** function takes a HSV triple and prints the description of the color. My function prototype is below;

hsv2desc :: (Float,Float,Float) -> String

I wrote 3 helper functions(hName,sName, vName) to get descriptions according to H, S and V values. These functions take Float and return String.

```
hName :: Float -> String
hName h
  | h<15     = "red"
  | h==15    = "reddish"
  | h<=45    = "orange"
  | h<=70    = "yellow"
  | h<=79    = "lime"
  | h<=163   = "green"
  | h<=193   = "cyan"
  | h<=240   = "blue"
  | h<=260   = "indigo"
  | h<=270   = "violet"
  | h<=291   = "purple"
  | h<=327   = "magenta"
  | h<=344   = "rose"
  | h<=360   = "red"
```

```
sName :: Float -> String
sName s
  | s<4     = "grey"
  | s<=10   = "almost grey"
  | s<=30   = "very unsaturated"
  | s<=46   = "unsaturated"
  | s<=60   = "rather unsaturated"
  | s<=80   = "saturated"
  | s<=90   = "rather saturated"
  | s<=100  = "very saturated"
```

```
vName :: Float -> String
vName v
  | v<10    = "almost black"
  | v<=22   = "very dark"
  | v<=30   = "dark"
  | v<=60   = "normal"
  | v<=80   = "light"
  | v<=94   = "very light"
  | v<=100  = "almost white"
```

You can see some of my test results about this function below;

```
C:\Program Files\Haskell Platform\8.6.5\bin\ghci.exe
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Project          ( C:\\Users\Emre\Desktop\Project.hs, interpreted )
Ok, one module loaded.
*Project> hsv2desc (0,0,0)
"Hue: red, Saturation: grey, Lightness: almost black"
*Project> hsv2desc (20,5,15)
"Hue: orange, Saturation: almost grey, Lightness: very dark"
*Project> hsv2desc (50,29,30)
"Hue: yellow, Saturation: very unsaturated, Lightness: dark"
*Project> hsv2desc (75,30,50)
"Hue: lime, Saturation: very unsaturated, Lightness: normal"
*Project> hsv2desc (150,40,70)
"Hue: green, Saturation: unsaturated, Lightness: light"
*Project> hsv2desc (240,60,85)
"Hue: blue, Saturation: rather unsaturated, Lightness: very light"
*Project> hsv2desc (300,85,90)
"Hue: magenta, Saturation: rather saturated, Lightness: very light"
*Project> hsv2desc (360,99,99)
"Hue: red, Saturation: very saturated, Lightness: almost white"
*Project> hsv2desc (360,100,100)
"Hue: red, Saturation: very saturated, Lightness: almost white"
*Project>
```

PART 6

In this part, I wrote a program which takes two color names and the number of steps from the user and prints the HSV gradient. My function prototype is below;

printGradient :: IO ()

Also, I wrote a helper function to show HSV and RGB representations of given HSV triple. Then, I used this function with mapM_ function to print all of the triples in gradient array.

You can see the some example runs of the program below;

```
C:\Program Files\Haskell Platform\8.6.5\bin\ghci.exe
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Project          ( C:\\Users\Emre\Desktop\Project.hs, interpreted )
Ok, one module loaded.
*Project> printGradient
Enter two color name:
#000000
#FFFFFF
Enter number of steps:
6
Gradients from color1= #000000 to color2= #FFFFFF in steps:6;
HSV: (0.0,0.0,0.0) , RGB: (0.0,0.0,0.0)
HSV: (0.0,0.0,16.666666) , RGB: (42.499996,42.499996,42.499996)
HSV: (0.0,0.0,33.333332) , RGB: (84.99999,84.99999,84.99999)
HSV: (0.0,0.0,50.0) , RGB: (127.5,127.5,127.5)
HSV: (0.0,0.0,66.666664) , RGB: (169.99998,169.99998,169.99998)
HSV: (0.0,0.0,83.33333) , RGB: (212.49998,212.49998,212.49998)
HSV: (0.0,0.0,100.0) , RGB: (255.0,255.0,255.0)
*Project>
*Project> printGradient
Enter two color name:
#FDB086
#02468A
Enter number of steps:
7
Gradients from color1= #FDB086 to color2= #02468A in steps:7;
HSV: (21.176472,47.035572,99.21568) , RGB: (252.99998,175.99998,133.99998)
HSV: (48.15126,54.39488,92.77311) , RGB: (236.57143,211.15929,107.88869)
HSV: (75.12605,61.754185,86.330536) , RGB: (185.8704,220.14285,84.19543)
HSV: (102.10083,69.113495,79.88796) , RGB: (104.92184,203.71431,62.92023)
HSV: (129.07562,76.47279,73.44539) , RGB: (44.063103,187.28575,65.72702)
HSV: (156.05042,83.83211,67.002815) , RGB: (27.624,170.85718,113.684265)
HSV: (183.02519,91.19141,60.56024) , RGB: (13.602997,147.32822,154.42862)
HSV: (209.99998,98.55072,54.11767) , RGB: (2.000015,70.00006,138.00005)
*Project>
```

PART 7

In this final part, I wrote two functions which take RGB or HSV triple and return HTML color name. My function prototypes are below;

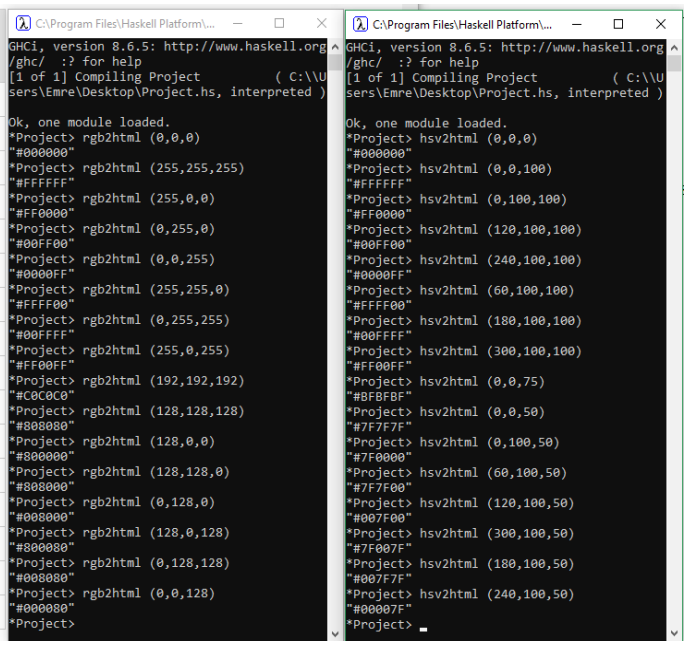
rgb2html :: :: (Float,Float,Float) -> String

hsv2html :: :: (Float,Float,Float) -> String

I wrote a helper function named 'intToHex' which converts numbers between 0 to 15 into hexadecimal numbers.

You can see some of my test results about these functions below;

Color	Color name	Hex	(R,G,B)	(H,S,V)
	Black	#000000	(0,0,0)	(0°,0%,0%)
	White	#FFFFFF	(255,255,255)	(0°,0%,100%)
	Red	#FF0000	(255,0,0)	(0°,100%,100%)
	Lime	#00FF00	(0,255,0)	(120°,100%,100%)
	Blue	#0000FF	(0,0,255)	(240°,100%,100%)
	Yellow	#FFFF00	(255,255,0)	(60°,100%,100%)
	Cyan	#00FFFF	(0,255,255)	(180°,100%,100%)
	Magenta	#FF00FF	(255,0,255)	(300°,100%,100%)
	Silver	#C0C0C0	(192,192,192)	(0°,0%,75%)
	Gray	#808080	(128,128,128)	(0°,0%,50%)
	Maroon	#800000	(128,0,0)	(0°,100%,50%)
	Olive	#808000	(128,128,0)	(60°,100%,50%)
	Green	#008000	(0,128,0)	(120°,100%,50%)
	Purple	#800080	(128,0,128)	(300°,100%,50%)
	Teal	#008080	(0,128,128)	(180°,100%,50%)
	Navy	#000080	(0,0,128)	(240°,100%,50%)



There is a little bit difference between these results of functions and the reason for that is my float representation of numbers. For example, “rgb2html (192,192,192)” returns #C0C0C0 which is correct but “hsv2html (0,0,75)” returns #BFBFBF which is very close to the exact value.

CONCLUSION

Color manipulation project was very informative and practical. I learned some important concepts better such as type compatibility, writing helper functions, using prelude functions (show, read, take, drop, map, putStrLn...) and doing IO operations. Moreover, I learned representation formats of colours and how to convert one to another. All in all, it was a good project to practice in Haskell.

REFERENCES

- https://www.rapidtables.com/convert/color/rgb-to-hsv.html
- https://www.rapidtables.com/convert/color/hsv-to-rgb.html
- https://www.w3schools.com/colors/colors_names.asp
- https://github.com/vasilisvg/human-colours