

ISTANBUL TECHNICAL UNIVERSITY

Color Manipulation

Functional Programming Project

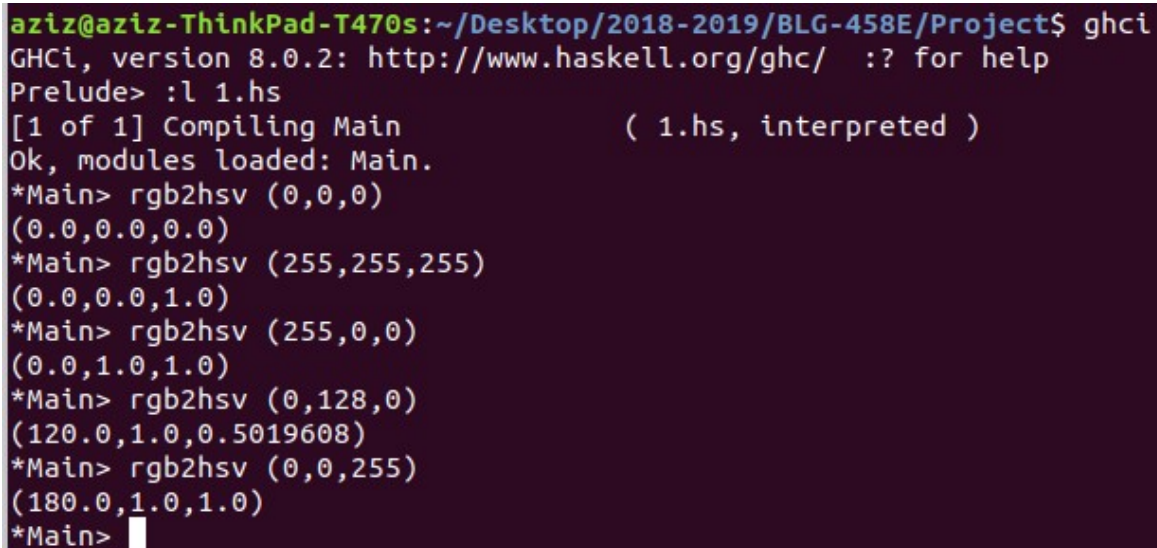
Aziz ÇETİN
150160515

1. Write a function "rgb2hsv" that will convert an RGB triple into an HSV triple.

In the first question, "rgb2hsv" function takes **(Float, Float, Float)** parameter and return **(Float, Float, Float)** value which is HSV triple by converting the RGB triple.

I test it by using black, white, red, green and blue. Calling the function like:

```
rgb2hsv (0, 0, 0)
rgb2hsv (255,255,255)
rgb2hsv (255,0,0)
rgb2hsv (0,128,0)
rgb2hsv (0,0,255)
```



```
aziz@aziz-ThinkPad-T470s:~/Desktop/2018-2019/BLG-458E/Project$ ghci
GHCi, version 8.0.2: http://www.haskell.org/ghc/  :? for help
Prelude> :l 1.hs
[1 of 1] Compiling Main                  ( 1.hs, interpreted )
Ok, modules loaded: Main.
*Main> rgb2hsv (0,0,0)
(0.0,0.0,0.0)
*Main> rgb2hsv (255,255,255)
(0.0,0.0,1.0)
*Main> rgb2hsv (255,0,0)
(0.0,1.0,1.0)
*Main> rgb2hsv (0,128,0)
(120.0,1.0,0.5019608)
*Main> rgb2hsv (0,0,255)
(180.0,1.0,1.0)
*Main> █
```

2. Write a function "hsv2rgb" that will convert an HSV triple into an RGB triple.

In the first question, "rgb2hsv" function takes **(Float, Float, Float)** parameter and return **(Float, Float, Float)** value which is RGB triple by converting the HSV triple.

I test it by using black, white, red, yellow and blue. Calling the function like:

```
hsv2rgb (0, 0, 0)
hsv2rgb (0, 0, 1)
hsv2rgb (0, 1, 1)
hsv2rgb (60, 1, 1)
hsv2rgb (240, 1, 1)
```

```

aziz@aziz-ThinkPad-T470s:~/Desktop/2018-2019/BLG-458E/Project$ ghci
GHCi, version 8.0.2: http://www.haskell.org/ghc/  :? for help
Prelude> :l 2.hs
[1 of 1] Compiling Main                                ( 2.hs, interpreted )
Ok, modules loaded: Main.
*Main> hsv2rgb (0,0,0)
(0.0,0.0,0.0)
*Main> hsv2rgb (0,0,1)
(255.0,255.0,255.0)
*Main> hsv2rgb (0,1,1)
(255.0,0.0,0.0)
*Main> hsv2rgb (60,1,1)
(255.0,255.0,0.0)
*Main> hsv2rgb (240,1,1)
(0.0,0.0,255.0)
*Main> 

```

3. Write a function “name2rgb” that will convert an HTML color name into an RGB triple.

This function takes the name of the HTML color and convert it into the RGB. The function takes “String” parameter and return (Integer, Integer, Integer) value.

```

name2rgb "Black"
name2rgb "White"
name2rgb "Red"
name2rgb "Green"
name2rgb "Blue"

```

```

aziz@aziz-ThinkPad-T470s:~/Desktop/2018-2019/BLG-458E/Project$ ghci
GHCi, version 8.0.2: http://www.haskell.org/ghc/  :? for help
Prelude> :l 3.hs
[1 of 1] Compiling Main                                ( 3.hs, interpreted )
Ok, modules loaded: Main.
*Main> name2rgb "Black"
(0,0,0)
*Main> name2rgb "White"
(255,255,255)
*Main> name2rgb "Red"
(255,0,0)
*Main> name2rgb "Green"
(0,128,0)
*Main> name2rgb "Blue"
(0,0,255)
*Main> 

```

4. Write a function “hsvGradient” that will take a starting HSV color, an ending HSV color, and a number of steps, and will produce a gradient that starts from the starting color and reaches the ending color in the given number of steps.

In this function “hsvGradient” takes $((t_0, t_0, t_0) \rightarrow (t_0, t_0, t_0) \rightarrow t_0 \rightarrow [(t_0, t_0, t_0)])$ parameter and return as you see $[(t_0, t_0, t_0)]$ value.

```
hsvGradient (0,0,0) (0,0,1) 5
hsvGradient (120,0,0) (240,0,1) 3
hsvGradient (50,1,0) (100,1,1) 2
hsvGradient (0,1,0) (300,0,0) 4
```

```
aziz@gaziz-ThinkPad-T470s:~/Desktop/2018-2019/BLG-458E/Project$ ghci
GHCi, version 8.0.2: http://www.haskell.org/ghc/  :? for help
Prelude> :l 4.hs
[1 of 1] Compiling Main                ( 4.hs, interpreted )
Ok, modules loaded: Main.
*Main> hsvGradient (0,0,0) (0,0,1) 5
[(0.0,0.0,0.0),(0.0,0.0,0.2),(0.0,0.0,0.4),(0.0,0.0,0.6),(0.0,0.0,0.8),(0.0,0.0,1.0)]
*Main> hsvGradient (120,0,0) (240,0,1) 3
[(120.0,0.0,0.0),(160.0,0.0,0.3333333333333333),(200.0,0.0,0.6666666666666667),(240.0,0.0,1.0)]
*Main> hsvGradient (50,1,0) (100,1,1) 2
[(50.0,1.0,0.0),(75.0,1.0,0.5),(100.0,1.0,1.0)]
*Main> hsvGradient (0,1,0) (300,0,0) 4
[(0.0,1.0,0.0),(75.0,0.75,0.0),(150.0,0.5,0.0),(225.0,0.25,0.0),(300.0,0.0,0.0)]
*Main> 
```

5. Write a function “hsv2desc” that will take an HSV triple and print a description of the color.

In this function “hsv2desc” takes $(\text{Float}, \text{Float}, \text{Float})$ parameter and return as you see $(\text{String}, \text{String}, \text{String})$ value.

```
hsv2desc (0,1,0.5)
hsv2desc (0,0,0.5)
```

```

*Main> hsv2desc (180,1,1)
("cyan","very saturated","normal")
*Main> hsv2desc (300,1,1)
("magenta","very saturated","normal")
*Main> hsv2desc (60,1,0.5)
("yellow","very saturated","dark")
*Main> hsv2desc (180,1,0.5)
("cyan","very saturated","dark")
*Main> 

```

6. Write a program that will take two color names and the number of steps from the user and prints the HSV gradient.

In this question I took two color name and the step numbers with I/O. After that to take the gradient firstly I took the name of colors to the rgb and than convert this rgb to hsv after all convert to this the hsv gradient.

To call the function we call the **getInteger** and press enter. Enter the colors name and step numbers after all the gradient is ready.

```

*Main> :l 6.hs
[1 of 1] Compiling Main                ( 6.hs, interpreted )
Ok, modules loaded: Main.
*Main> getInteger
First Color:
"Red"
Second Color:
"Green"
Step Number:
4
[(0.0,1.0,1.0),(30.0,1.0,0.8754902),(60.0,1.0,0.7509804),(90.0,1.0,0.62647057),(
120.0,1.0,0.5019608)]
*Main> getInteger
First Color:
"White"
Second Color:
"Blue"
Step Number:
2
[(0.0,0.0,1.0),(90.0,0.5,1.0),(180.0,1.0,1.0)]
*Main> 

```

7. (Bonus) Write a function that, given an RGB or HSV triple, returns the “closest” HTML color name.

For this question I try to calculate closest Html Color. I take the name of the color and I took the index, min, counter and color of RGB. Using this parameters I find the index of the color in the list of colors. I gave the min 10000 for starting. I gave it too high because handle the errors. I subtract the (r,g,b) to (a,d,e) after that if the result is smaller than min result is result else result is min. This recursion give us the index of the color in the list.

```
closestHtmlColor "RGB" (1,1,1)
closestHtmlColor "HSV" (100,1,1)
```

```
*Main> :l 7.hs
[1 of 1] Compiling Main                ( 7.hs, interpreted )
Ok, modules loaded: Main.
*Main> closestHtmlColor "RGB" (1,1,1)
"Black"
*Main> closestHtmlColor "RGB" (0,0,0)
"Black"
*Main> closestHtmlColor "RGB" (120,0,0)
"Green"
*Main> closestHtmlColor "RGB" (240,1,1)
"DarkOliveGreen"
*Main> closestHtmlColor "RGB" (255,1,1)
"Olive"
*Main> 
```

```
Ok, modules loaded: Main.
*Main> closestHtmlColor "HSV" (100,1,1)
"Chocolate"
*Main> closestHtmlColor "HSV" (300,0,1)
"White"
*Main> closestHtmlColor "HSV" (120,0.5,0.5)
"Blue"
*Main> 
```