```
//James Rogers Jan 2022 (c) Plymouth University
#include <iostream>

#include<opencv2/opencv.hpp>
#include<opencv2/opencv_modules.hpp>

using namespace std;
using namespace cv;

int main(){

    //Path of image folder
    string PathToFolder = "../Task1/Car Images/";

    //Loop through the 30 car images
    for(int n=0; n<30; ++n){

        //Each image is named 0.png, 1.png, 2.png, etc. So generate the image file path based on n
and the folder path
        string PathToImage = PathToFolder+to_string(n)+".png";

        cout<<PathToImage<<endl;

        //Load car image at the file paths location
        Mat Car=imread(PathToImage);

        //Your code goes here. The example code below shows you how to read the red, green, and blue
colour values of the
        //pixel at position (0,0). Modify this section to check not just one pixel, but all of them
in the 640x480 image
        //(using for-loops), and using the RGB values classifiy if a given pixel looks red, green,
blue, or other.

        //==============example code, feel free to delete=============
        int rows, cols;                                                 //variables to hold the size
of the matrix
        int bluecount = 0,greencount = 0,redcount = 0;                  //pixel counts
        double lowlimit = 0.2,highlimit=0.8;                            //limits for the enhanced
weighting area
        int x=0;
        int y=0;
        Size s = Car.size();                                            //extract the size of the
matrix
        rows = s.height;                                                //number of rows of pixels
        cols = s.width;                                                 //number of columns of
pixels

        for(x=0; x<cols; x++){
            for(y=0; y<rows; y++){
                Vec3b PixelValue = Car.at<Vec3b>(y,x);                   //get pixel value
                //check which value is higher
                int b,g,r;                                               //b g r values
                b = PixelValue[0];
                g = PixelValue[1];
                r = PixelValue[2];
                if ((b>1.5*g) && (b>1.5*r) && (b>125)){                 //1.5 is the weighting
threshold
                    bluecount++;
                    if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
                        bluecount = bluecount + 3;                      //Enhanced weighting area
counts for triple
                    }
                }
                else if ((g>1.5*b) && (g>1.5*r) && (g>125)){
                    greencount++;
                    if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
                        greencount = greencount +3;
                    }
                }
                else if ((r>1.5*b) && (r>1.5*g) && (r>125)){
                    redcount++;
                    if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
                        redcount = redcount +3;
                    }
                }
            }
        };
        cout << "Blue Count = " << (int)bluecount <<endl;
        cout << "Green Count = " << (int)greencount <<endl;
        cout << "Red Count = " << (int)redcount <<endl;
        if ((bluecount>greencount) && (bluecount>redcount)){
            cout<<"This car is blue" <<endl;
        }
        else if ((greencount>bluecount) && (greencount>redcount)){
            cout<<"This car is green" <<endl;
        }
        else {
            cout<<"This car is red" <<endl;
        }

        //============================================================

        //display the car image untill x is pressed
        while(waitKey(10)!='x'){
            imshow("Car", Car);
        }
    }
    //testing with expanded dataset
    int blue_correct = 0,green_correct = 0,red_correct = 0,result = 0, temp_correct = 0;
    for(int p=0; p<3; ++p){
        temp_correct = 0;
        for(int n=1; n<30; ++n){

            //Each image is named 0.png, 1.png, 2.png, etc. So generate the image file path based on
n and the folder path
            if (p==0){
                PathToFolder = "../Task1/add_blue/";
            } else if (p==1) {
                PathToFolder = "../Task1/add_green/";
            } else {
                PathToFolder = "../Task1/add_red/";
            }
            string PathToImage = PathToFolder+to_string(n)+".png";
            cout<<PathToImage<<endl;
            //Load car image at the file paths location
            Mat Car=imread(PathToImage);
            int rows, cols;                                                 //variables to hold the
size of the matrix
            int bluecount = 0,greencount = 0,redcount = 0;                  //pixel counts
            double lowlimit = 0.2,highlimit=0.8;                            //limits for the
enhanced weighting area
            int x=0;
            int y=0;
            Size s = Car.size();                                            //extract the size of
the matrix
            rows = s.height;                                                //number of rows of
pixels
            cols = s.width;                                                 //number of columns of
pixels

            for(x=0; x<cols; x++){
                for(y=0; y<rows; y++){
                    Vec3b PixelValue = Car.at<Vec3b>(y,x);                   //get pixel value
                    //check which value is higher
                    int b,g,r;                                               //b g r values
                    b = PixelValue[0];
                    g = PixelValue[1];
                    r = PixelValue[2];
                    if ((b>1.5*g) && (b>1.5*r) && (b>125)){
                        bluecount++;
                        if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
                            bluecount = bluecount + 3;
                        }
                    }
                    else if ((g>1.5*b) && (g>1.5*r) && (g>125)){
                        greencount++;
                        if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
                            greencount = greencount +3;
                        }
                    }
                    else if ((r>1.5*b) && (r>1.5*g) && (r>125)){
                        redcount++;
                        if ((x>cols*(lowlimit)) && (x<cols*(highlimit))){
                            redcount = redcount +3;
                        }
                    }
                }
            };
            if ((bluecount>greencount) && (bluecount>redcount)){
                result = 0;
            }
            else if ((greencount>bluecount) && (greencount>redcount)){
                result = 1;
            }
            else {
                result = 2;
            }
            if (p == result){
                temp_correct++;
            } else {
                cout <<"Incorrect - image " << n <<endl;
                cout << "Blue Count = " << (int)bluecount <<endl;
                cout << "Green Count = " << (int)greencount <<endl;
                cout << "Red Count = " << (int)redcount <<endl;

                //display the car image untill x is pressed
                while(waitKey(10)!='x'){
                    imshow("Car", Car);
                }
            }
        }
        //offload temp result into the correct result variable
        if (p==0){
            blue_correct = temp_correct;
        } else if (p==1){
            green_correct = temp_correct;
        } else if (p==2) {
            red_correct = temp_correct;
        }
    }
    cout << "blue correct = " << blue_correct+1<<"/30" <<endl;
    cout << "green correct = " << green_correct+1<<"/30" <<endl;
    cout << "red correct = " << red_correct+1<<"/30" <<endl;

}
```