ALGORITMO PERCEPTRÓN PARA LA SEPARACIÓN DE COLORES CON ARDUINO

AUTOR:

JHAN CARLOS ARANGO USUGA

INFORME:

INTELIGENCIA ARTIFICIAL

DOCENTE: JESUS EMILIO PINTO LOPERA

UNIVERSIDAD DE LA AMAZONIA

FACULTAD DE INGENIERIA

INGENIERIA DE SISTEMA

FLORENCIA - CAQUETA

2023

Tabla de Contenido

[INTRODUCCION 2](#_Toc629506369)

[DISEÑO DEL PROTOTIPO DEL SEPARADOR 3](#_Toc1874146638)

[METODOLOGIA DE DESARROLLO 4](#_Toc643184280)

[Clase Perceptrón 6](#_Toc1902516010)

[Clase Sensor de Color 10](#_Toc1795217814)

[Clase Servo Motor 12](#_Toc470822626)

[Clase Led 13](#_Toc1085863511)

[Clase Main 14](#_Toc86234894)

[RESULTADOS DE ENTRENAMIENTO 16](#_Toc443278556)

[RESULTADOS DE LA CLASIFICACION DE MODELO FINAL 17](#_Toc1930442128)

[ANÁLISIS DE LOS RESULTADOS 20](#_Toc1723798123)

[CONCLUSION 21](#_Toc992228205)

[Bibliografía 23](#_Toc391253875)

[ANEXOS 23](#_Toc1416718145)

Tabla de Ilustraciones

[Ilustración 1 Diseño Separador, Vista interna 5](#_Toc129763357)

[Ilustración 2.Diseño Completo del separador de color. 5](#_Toc129763358)

[Ilustración 3 Datos Linealmente separable 6](#_Toc129763359)

[Ilustración 4 Esquema del Perceptrón 6](#_Toc129763360)

[Ilustración 5 Datos obtenidos del Sensor color 7](#_Toc129763361)

[Ilustración 6.Clase Perceptrón, librerías. 8](#_Toc129763362)

[Ilustración 7. Declaración de Variables. 8](#_Toc129763363)

[Ilustración 8.Método llenar 9](#_Toc129763364)

[Ilustración 9. Método Perceptron\_Entrenado. 9](#_Toc129763365)

[Ilustración 10. Método Entrenar\_perceptron. 10](#_Toc129763366)

[Ilustración 11. Algoritmo Perceptrón. 11](#_Toc129763367)

[Ilustración 12.Métodos para retornar pesos. 11](#_Toc129763368)

[Ilustración 13.Variables de la clase sensor. 12](#_Toc129763369)

[Ilustración 14. Métodos de Retorno de color. 13](#_Toc129763370)

[Ilustración 15. Variables clase servomotor. 14](#_Toc129763371)

[Ilustración 16. Clase Led. 15](#_Toc129763372)

[Ilustración 17. Clase Main. 16](#_Toc129763373)

[Ilustración 18. Datos del primer color 16](#_Toc129763374)

[Ilustración 19. Datos de Segundo Color 17](#_Toc129763375)

[Ilustración 20. Entrenamiento Perceptrón 17](#_Toc129763376)

[Ilustración 21. Clasificación del Color 17](#_Toc129763377)

[Ilustración 22. Resultados del entrenamiento 18](#_Toc129763378)

[Ilustración 23. Color Rosa 19](#_Toc129763379)

[Ilustración 24. Color verde. 20](#_Toc129763380)

[Ilustración 25. Color azul. 21](#_Toc129763381)

[Ilustración 26. Datos Finales 22](#_Toc129763382)

# INTRODUCCION

En el presente informe se presenta el desarrollo de la práctica del reconocimiento y clasificación de colores realizada por medio de técnica de inteligencia artificial, específicamente el algoritmo del perceptrón usando el lenguaje de programación c++ incorporado en Arduino. Todo esto con el fin de combinar la capacidad de la computadora para aprender mediante un algoritmo de aprendizaje supervisado y el potencial de Arduino para usar microcontroladores y operar componentes electrónicos para lograr de forma física clasificar, en este caso colores mediante un servomotor guiado por un aprendizaje de máquina.

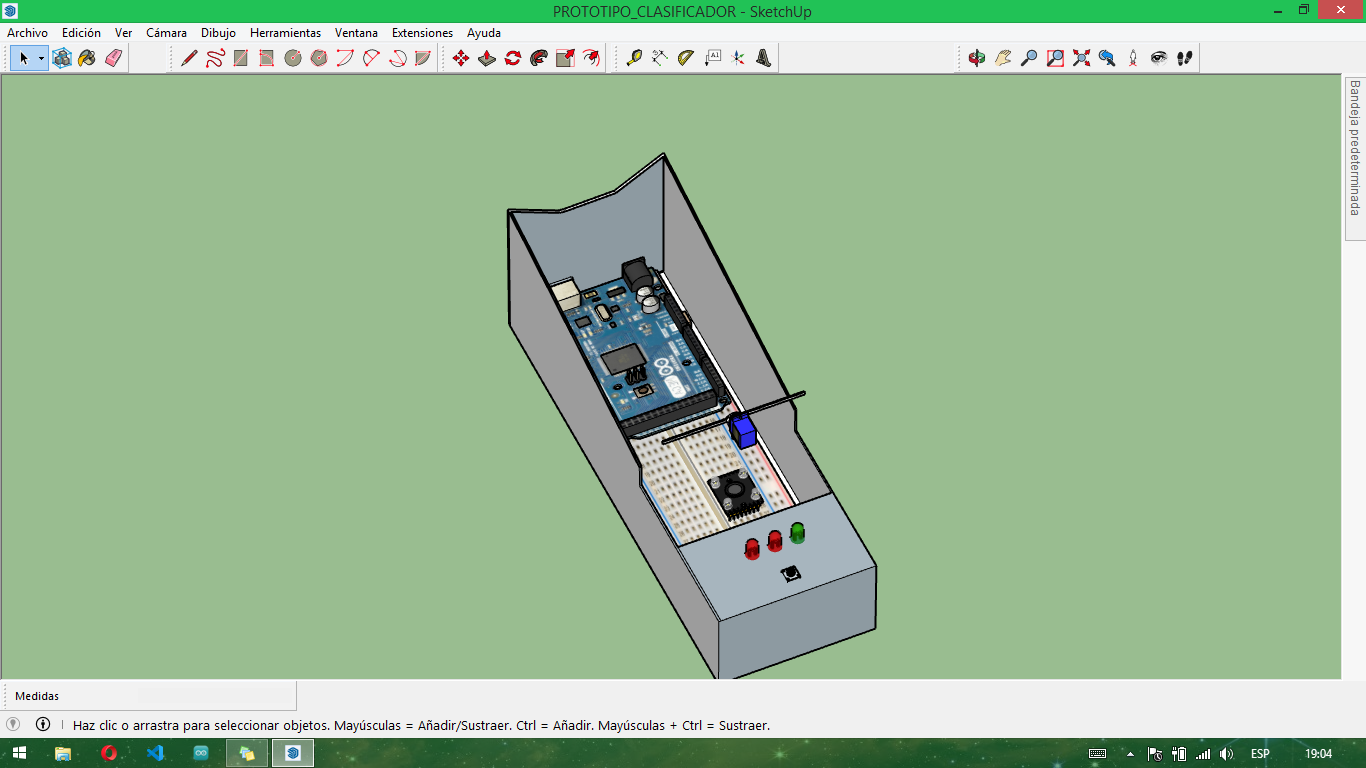
Esto es posible ya que la inteligencia artificial puede definirse como una sucesión de métodos de aprendizaje que busca igualar o imitar el proceso cognitivo de la mente humana, pero en realidad este solo simula el proceso de razonamiento humano (Soto-Peñaranda et al., 2019).

Siendo así las tecnologías que implementas estos tipos de algoritmos una mejora en la vida de los humanos ya que permite un mejor beneficio y eficiencia en la mayoría de los procesos presentes en el día a día o en procesos específicos como el ámbito laboral(Rouhiainen, 2018).

# DISEÑO DEL PROTOTIPO DEL SEPARADOR

Para el diseño del prototipo se usó SketchUp el cual es un programa para el diseño y modelado en tres dimensiones, usado para todo tipo de proyectos desde arquitectura hasta diseño escénico entre otros (Wikipedia.org, 2023). Gracias a esta herramienta se obtuvo un diseño sencillo para construir con pocos materiales y de bajos recursos.

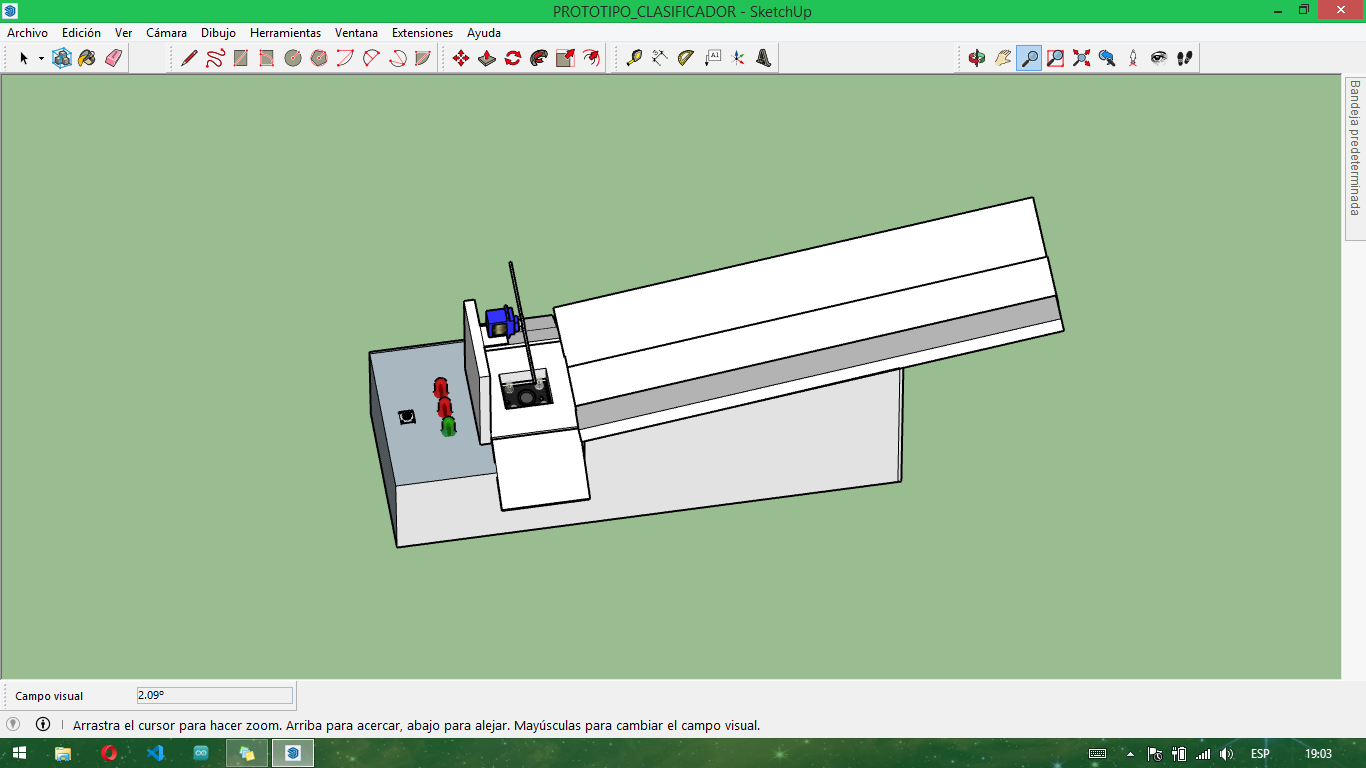
La siguiente ilustración se observa la parte interna del separador de color físico, en el cual dentro de el se encuentra Arduino.

Ilustración 1 Diseño Separador, Vista interna 

***Fuente:*** *elaboración propia.*

La ilustración 2 muestra el diseño final de separador de color para su posterior construcción en plano real para usarlo como clasificador (Véase anexos).

Ilustración 2. Diseño Completo del separador de color.



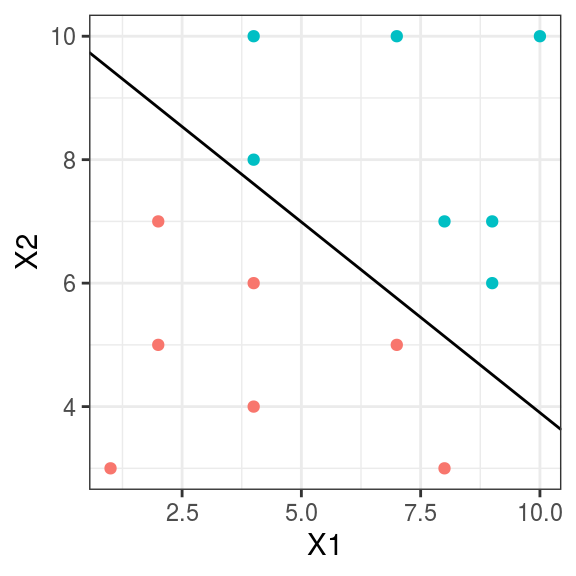
***Fuente:*** *elaboración propia.*

# METODOLOGIA DE DESARROLLO

Para trabajar con el perceptrón este debe operar datos que son linealmente separables, es decir que los mediante una línea se pueda determinar si pertenecen a una clase o no gracias a esto el modelo puede funcionar correctamente de lo contrario este no podría tener una solución lineal.

En la siguiente ilustración se puede ver como los datos representados en un plano bidimensional pueden estar separados por una línea, dando así dos conjuntos de datos distintos.

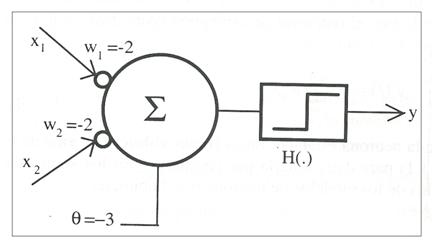
Ilustración 3 Datos Linealmente separables



(Amat Rodrigo, 2018)

Ahora bien, las entradas del perceptrón llevan unos pesos asociados que son los que deben ser ajustados y que son los que nos permiten junto al Bias encontrar la sumatoria para que la función de activación nos de salida perteneciente a una clase.

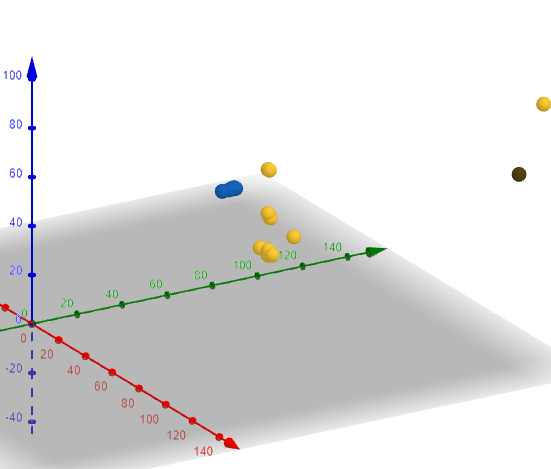
Ilustración 4 Esquema del Perceptrón



(Grupo.us, 2023)

Como se mencionó en las ilustraciones anteriores para que el modelo funcione correctamente este debe trabajar con datos separables, para verificar que los datos obtenidos de los sensores son separables que uso una herramienta para graficar las entradas del modelo, en la siguiente ilustración se puede observar cómo los conjuntos de datos están linealmente separables

Ilustración 5 Datos obtenidos del Sensor color



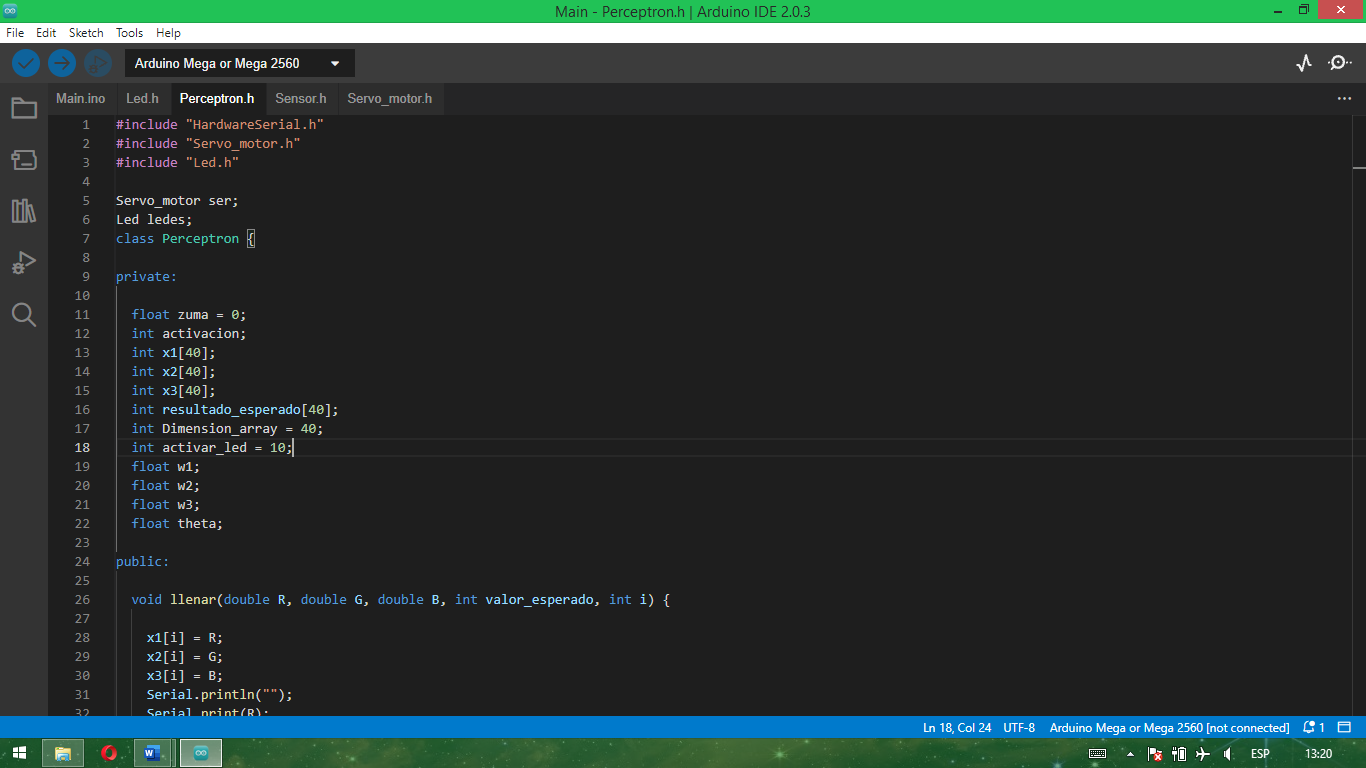
***Fuente:*** *elaboración propia.*

## Clase Perceptrón

Para empezar con la codificación la ilustración 6 corresponde a la clase Perceptron.h contiene el algoritmo del perceptrón, el cual recibe los datos obtenidos del sensor de color para su entrenamiento.

Ahora bien, las siguientes librerías son la Importaciones de las clases Led.h y Servo\_motor.h el cual son instancias que sirven como objetos en la clase perceptrón.h para usar sus métodos disponibles.

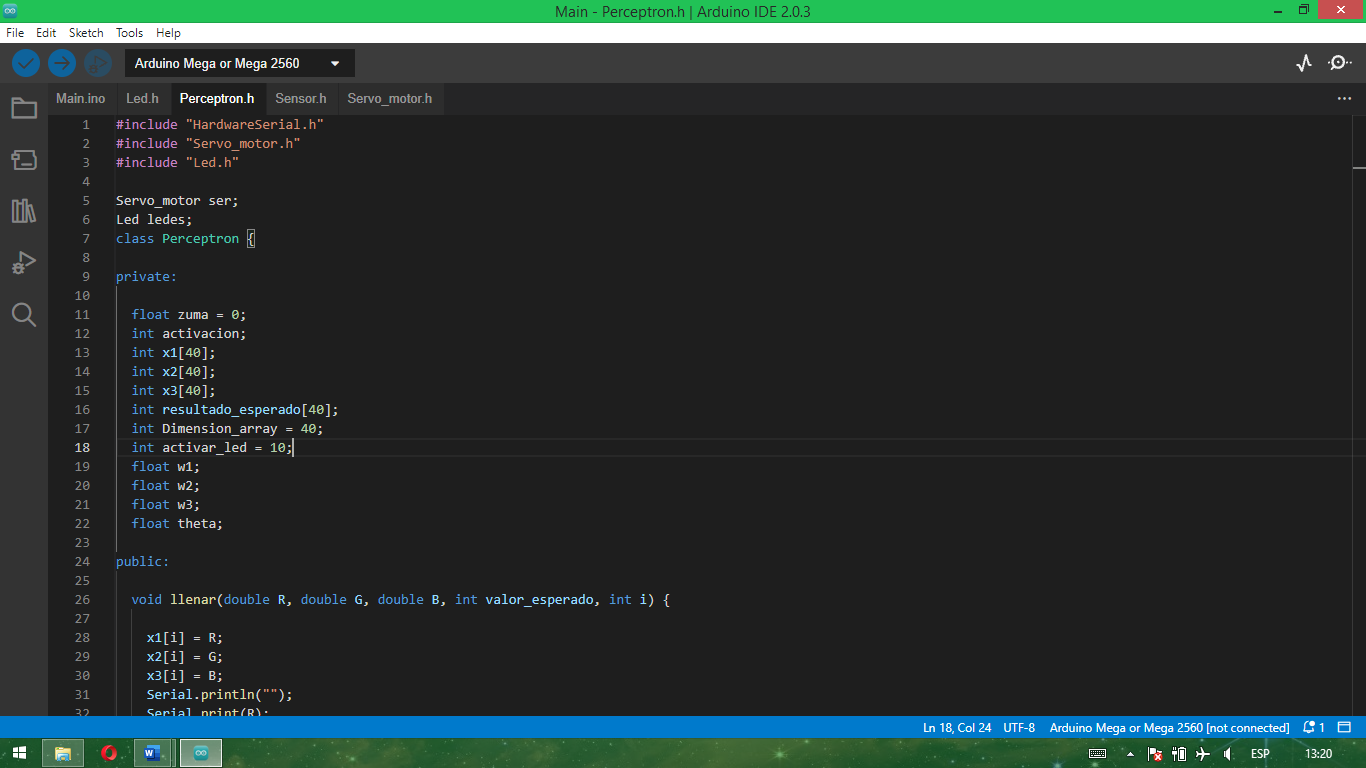
Ilustración 6. Clase Perceptrón, librerías.



***Fuente:*** *elaboración propia.*

La ilustracion 7 corresponde a la declaracion de variables que fueron utilizadas para implementar el algoritmo del perceptron, el cual consta de cuatro vectores x1, x2, x3, resultados\_esperados con un tamaño de 40 donde en los tres primeros se almacenaron los datos obtenidos del sensor y en el ultimo la clase a la cual pertenecen los datos, en este caso fueron 1 y -1. Las Variables w1,w2,w3 corresponden a los pesos y al Bias se denomino theta en la variable zuma, se almacena la sumatoria de los pesos por las entradas para posteriormente ser utilizadas para separar.

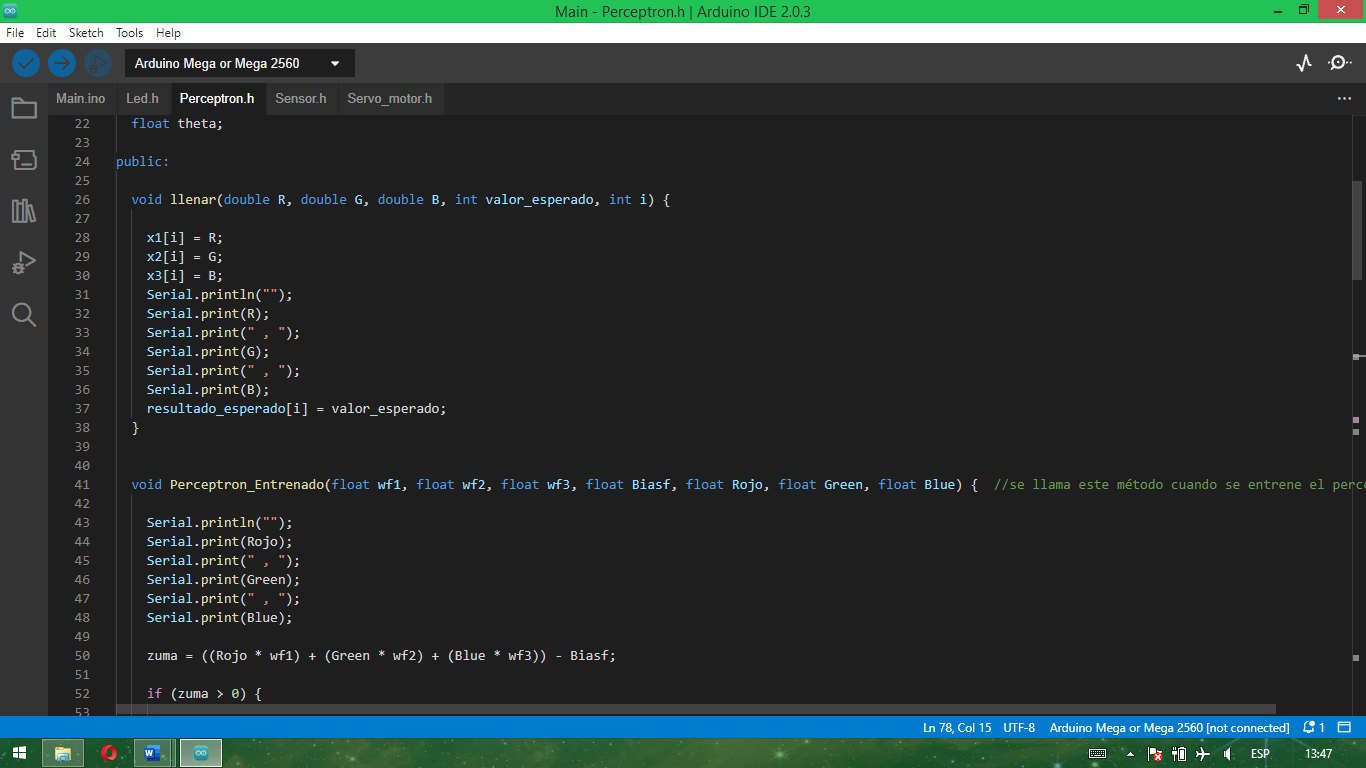
Ilustración 7. Declaración de Variables.

***Fuente:*** *elaboración propia.*

Para poder obtener los datos del sensor de color se creo el metodo llenar, el cual recibe 5 parametros R, G, B que corressponden a la composicion del color.

Esto es proporcionado por el sensor y luego se almaceno en los vectores x1,x2,x3 que son las entradas del perceptron, el parametro valor\_esperado recibe la clase que sera asignada para el color actual y por último el parametro i recibe la posicion del vector. ver ilustracion 8.

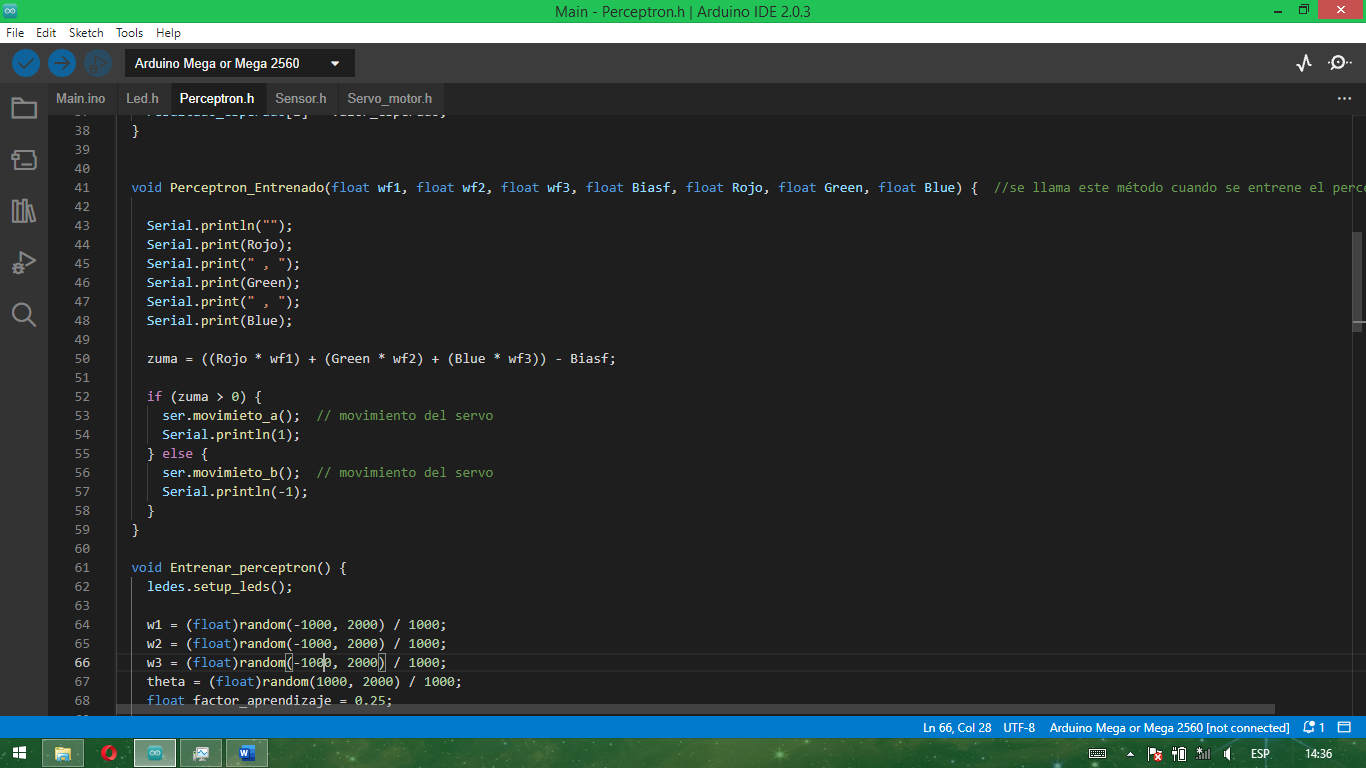
Ilustración 8. Método llenar



***Fuente:*** *elaboración propia.*

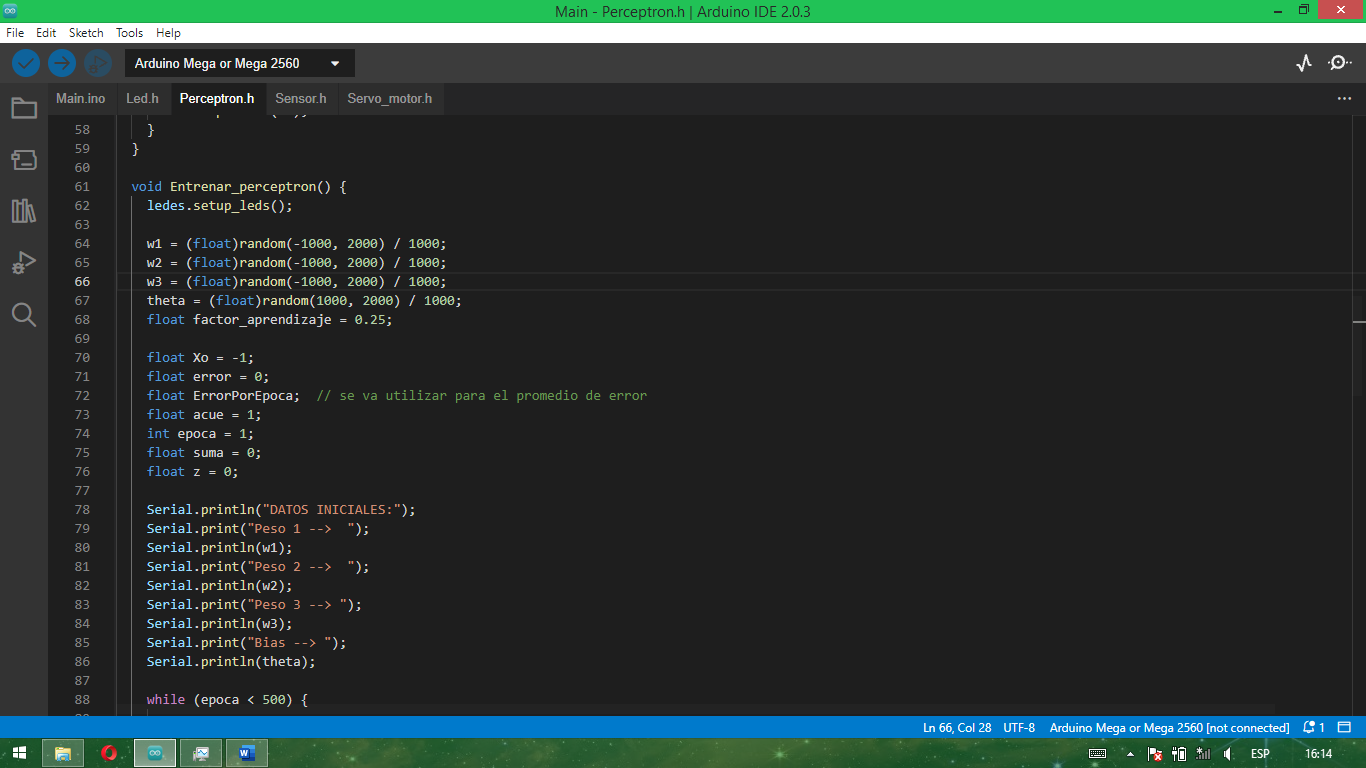
El método Perceptron\_entrenado en la ilustración 9 recibe 7 parámetros los cuales son usados para calcular la sumatoria en la ecuación de la recta y posteriormente clasificar según las entradas del modelo, si la sumatoria es mayor que el umbral el giro del servo es hacia la derecha de lo contrario hacia la izquierda, este método emplea un objeto de la Clase servo\_motor.h el cual dispone dos métodos para la rotación del servo; Ver ilustración 15,

Ilustración 9. Método Perceptron\_Entrenado.

***Fuente:*** *elaboración propia.*

En la ilustración 10, se dio inicio al algoritmo del perceptrón, eje central de este proyecto. Se definió los pesos y el Bias de manera aleatoria en un rango -1000 y 2000 con una tasa de aprendizaje del 0.25, se crean las variables X0 el cual es una constante para actualizar el Bias, la variable Error el cual es calculado por el valor deseado menos el obtenido dando así el error por cada época, la variable ErrorPorEpoca donde se almacena la suma del error de todas las épocas para calcular su promedio, la variable suma donde se guarda el resultado de sumar las entradas por los pesos menos el Bias, la variable z donde se guarda lo que retorna la función de activación en este caso es 1,-1 por último la línea de código 62 en el cual se encuentra el método setup\_ledes sirve para iniciar la configuración del led, ver ilustración 15.

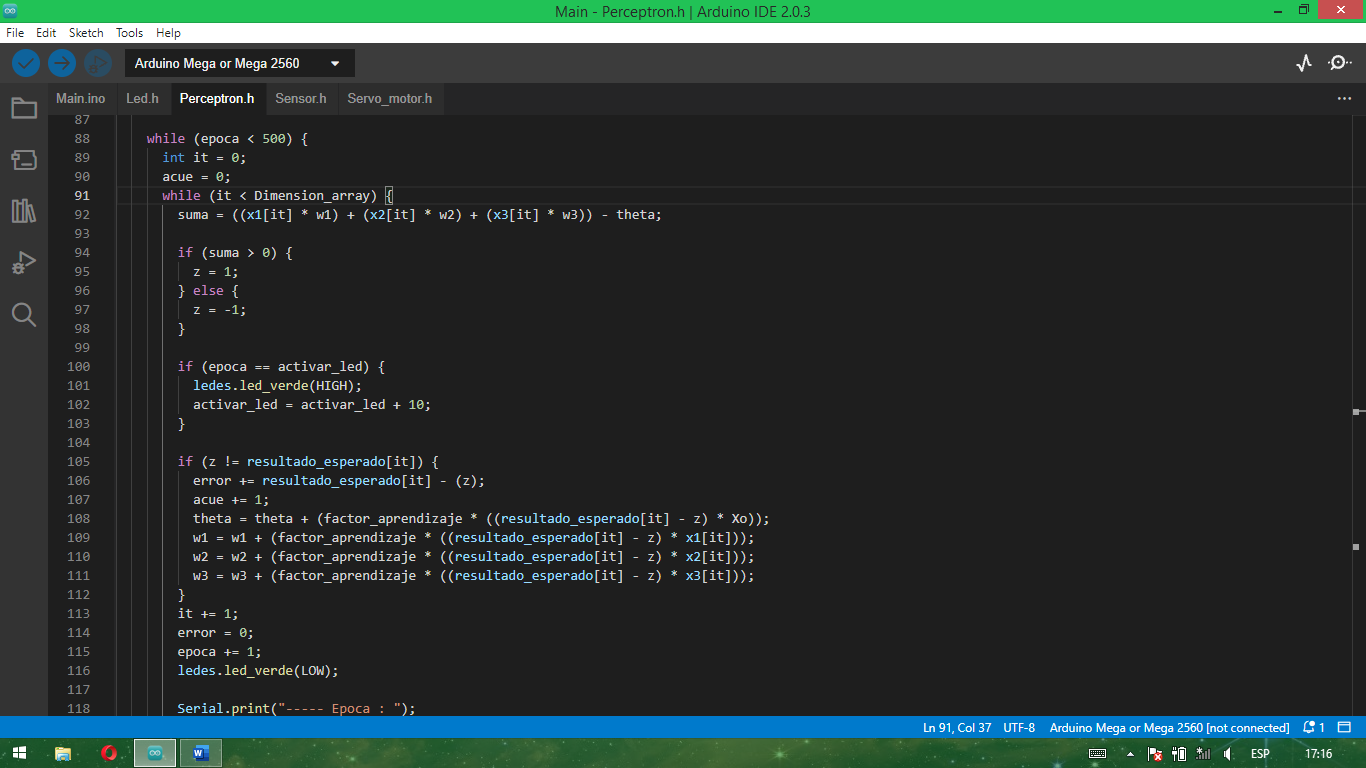
Ilustración 10. Método Entrenar\_perceptron.



***Fuente:*** *elaboración propia.*

En la Ilustración 11, para el entrenamiento del perceptrón se determina un numero de épocas menor a 500, ya que, al no ajustar correctamente los pesos del perceptrón, este debe tener un punto de parada (línea 88). En la línea 91 mientras la iteración sea menor al tamaño del vector que se encuentra en la variable Dimensio\_array. Se debe multiplicar cada posición por los pesos correspondientes para obtener la sumatoria de cada instancia y poder determinar la salida de la función escalón, ya que, gracias a este en la línea 105 se pudo determinar si no corresponde a los datos esperados dado que al no ser correctos el modelo deberá ser ajustado. Desde la línea 108 hasta la 111 se ajusta el Bias y los pesos

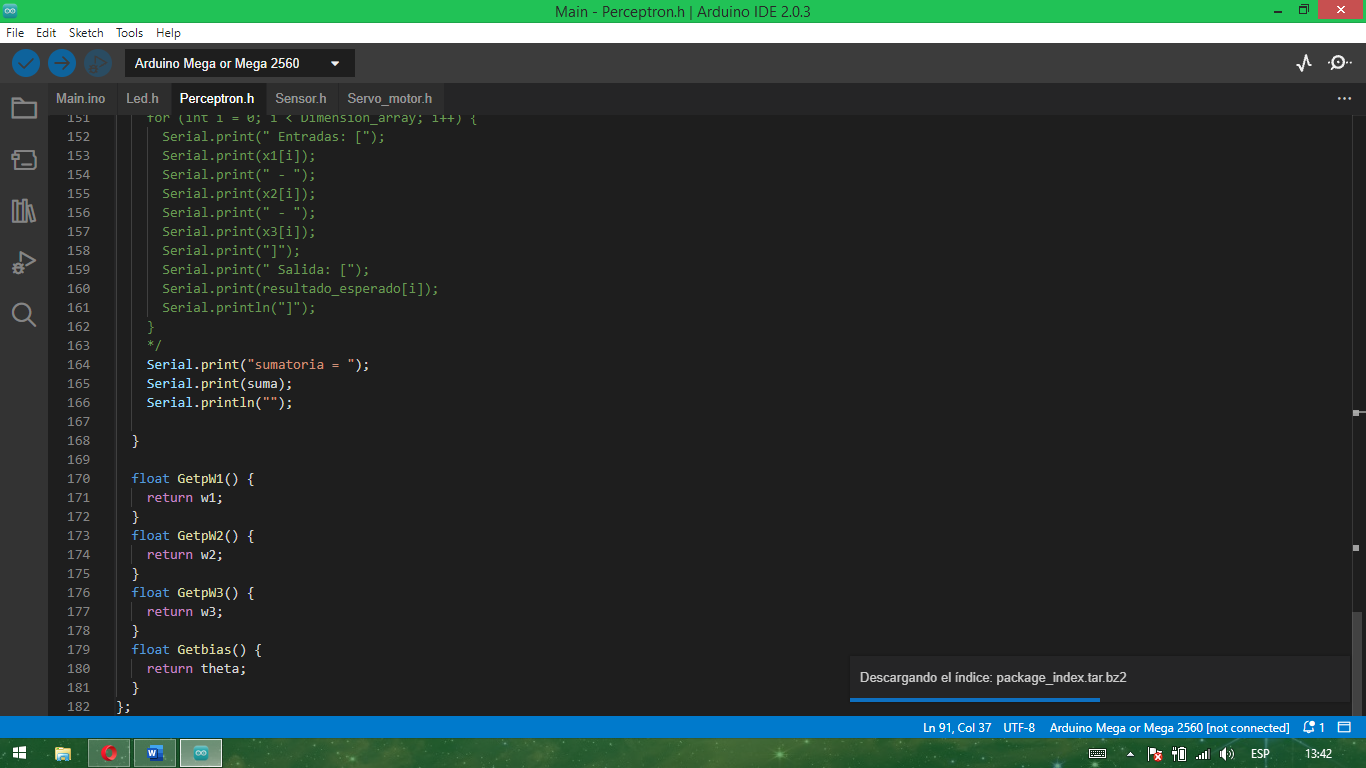
Ilustración 11. Algoritmo Perceptrón.



***Fuente:*** *elaboración propia.*

Por último, referente a la clase Perceptrón se crearon cuatro métodos que nos permito retornar los pesos y el Bias

Ilustración 12. Métodos para retornar pesos.

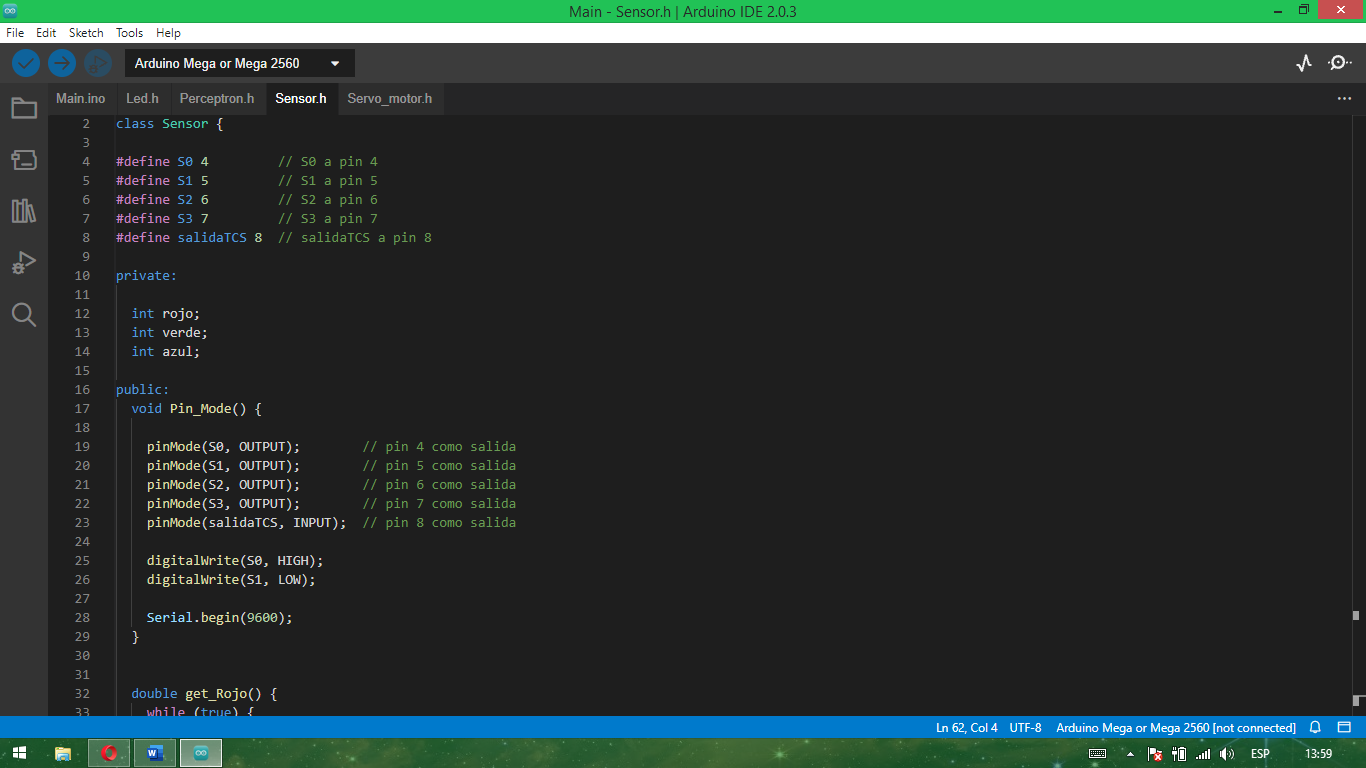


***Fuente:*** *elaboración propia.*

## Clase Sensor de Color

Para la clase Sensor en la ilustración 13 se definieron los pines 4,5,6,7,8 para conectar el sensor de color con Arduino, las variables rojo, verde y azul se definieron como entero para guardar los datos del sensor. El método pin\_mode inicia la comunicación con Arduino y la velocidad del serial en este caso 9600 Baudios por segundo.

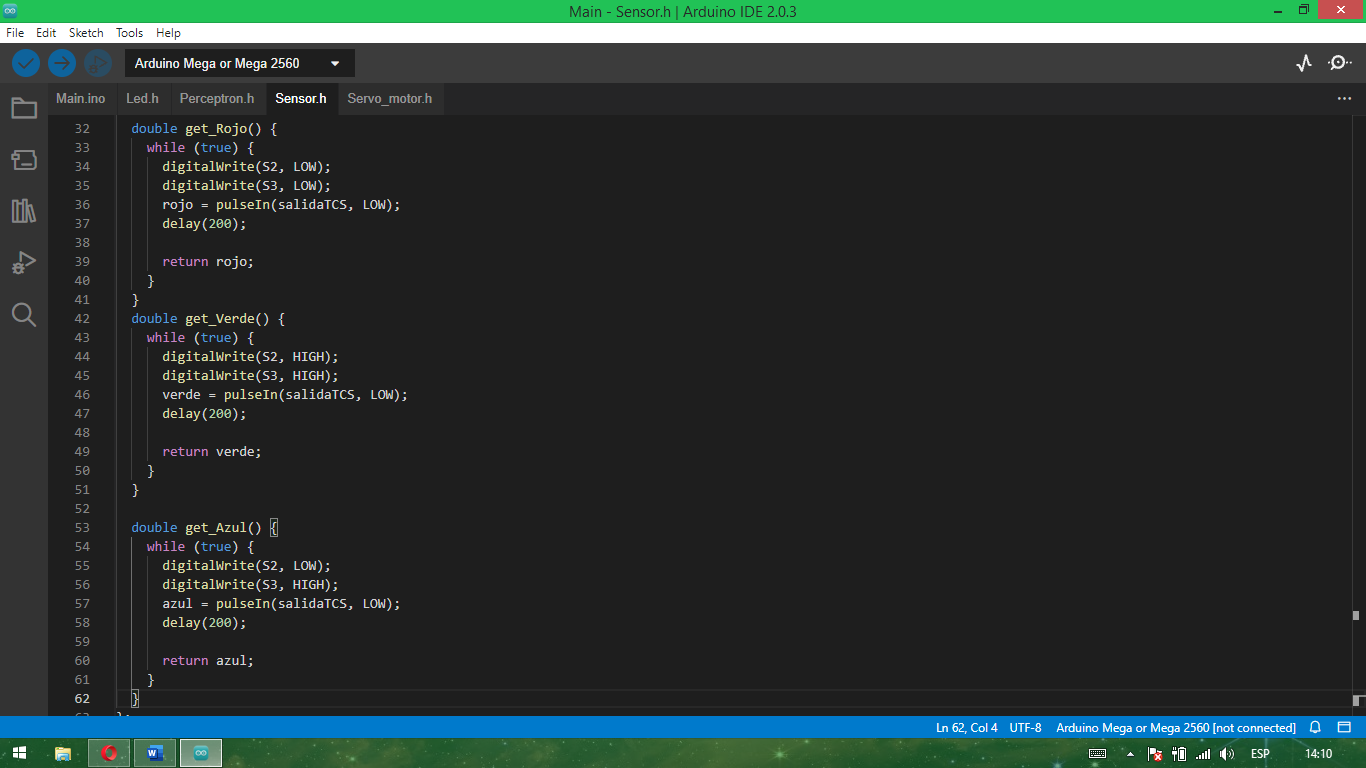
Ilustración 13. Variables de la clase sensor.



***Fuente:*** *elaboración propia.*

En la ilustración # , se usaron tres métodos correspondiente a cada color que arroja el sensor, cada método retorna el valor del sensor para se usado en la clase Main. ver ilustracio 14.

Ilustración 14. Métodos de Retorno de color.

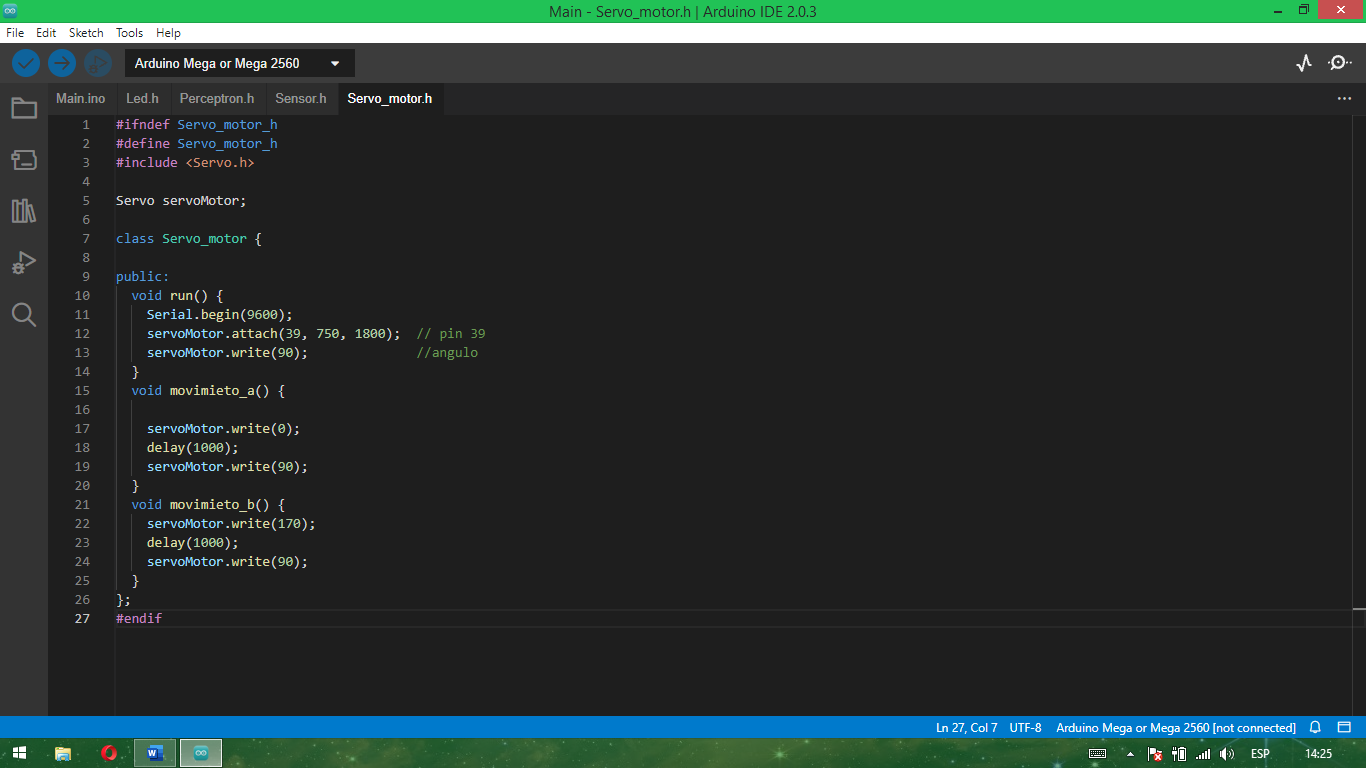


***Fuente:*** *elaboración propia.*

## Clase Servo Motor

En la ilustración 15, se encuentra la clase servo Motor el cual contiene el pin de comunicación 9 y el anglo mínimo y máximo y su velocidad de comunicación 9600 Baudios, por último, se creó dos métodos para el movimiento A, de 0 grados a 90 grados y el movimiento B de 90 grados a 180 grados.

Ilustración 15. Variables clase servomotor.

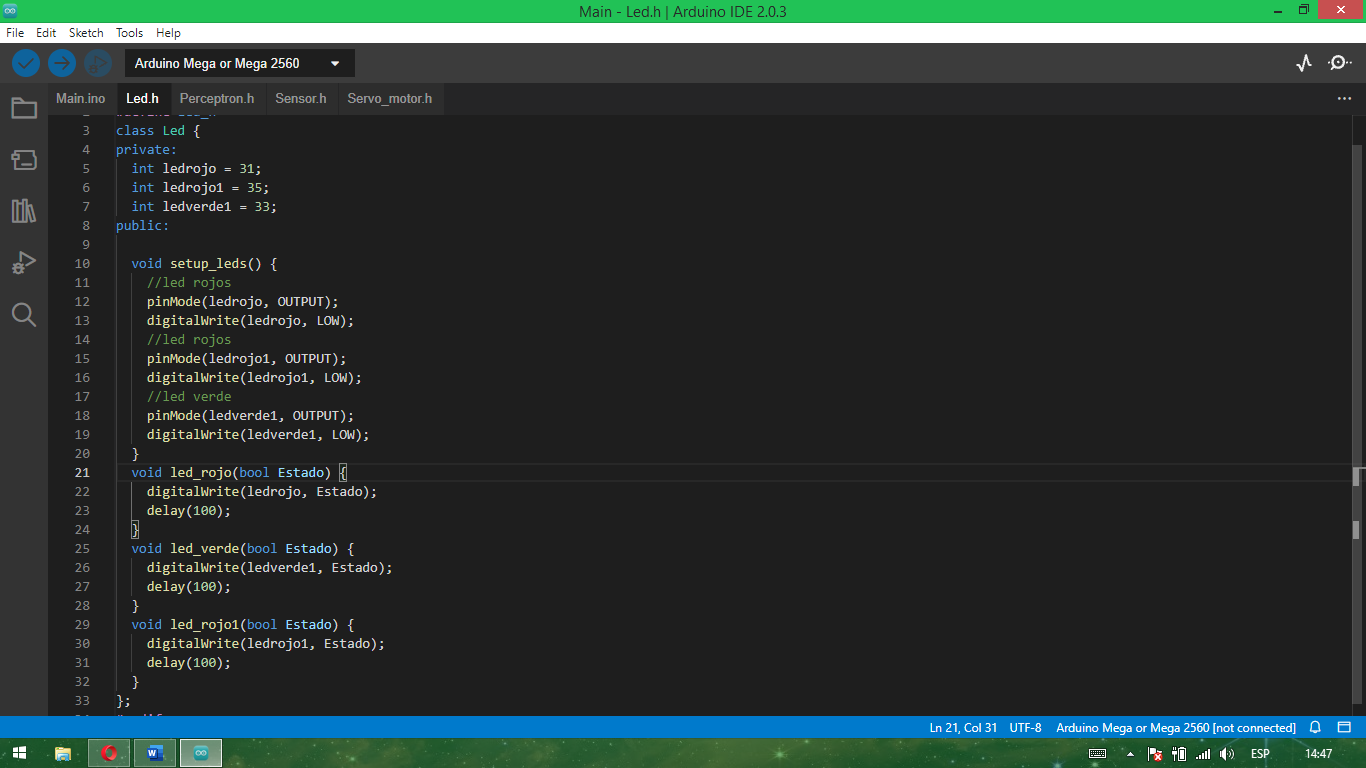


***Fuente:*** *elaboración propia.*

## Clase Led

Para la clase Led, ilustración 16, se declararon tres variables que corresponden a los pines de comunicación, esta clase solo se usa para encender y apagar los leds dependiendo de cada color entrenado. Contiene cuatro métodos, el setup\_leds nos permitió iniciar la comunicación con el Arduino y los otros tres métodos iluminan un led especifico estos tres últimos métodos reciben un parámetro para determinar si este encendido o apagado.

Ilustración 16. Clase Led.

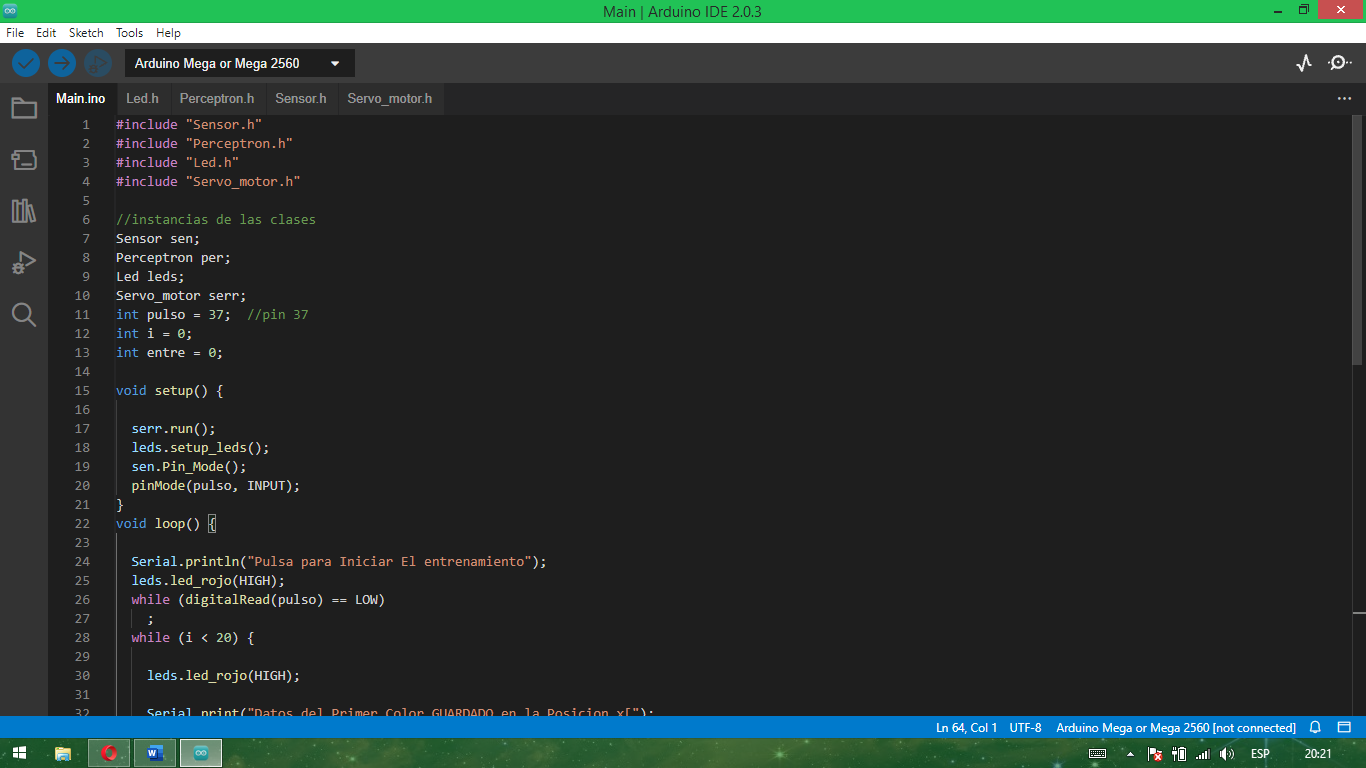


***Fuente:*** *elaboración propia.*

## Clase Main

Por últimos en la ilustración 17 se encuentra la clase Main en el cual se instancian las demás clases descritas anteriormente. En el primer método se llaman los métodos principales de cada clase que enciende la comunicación con el Arduino.

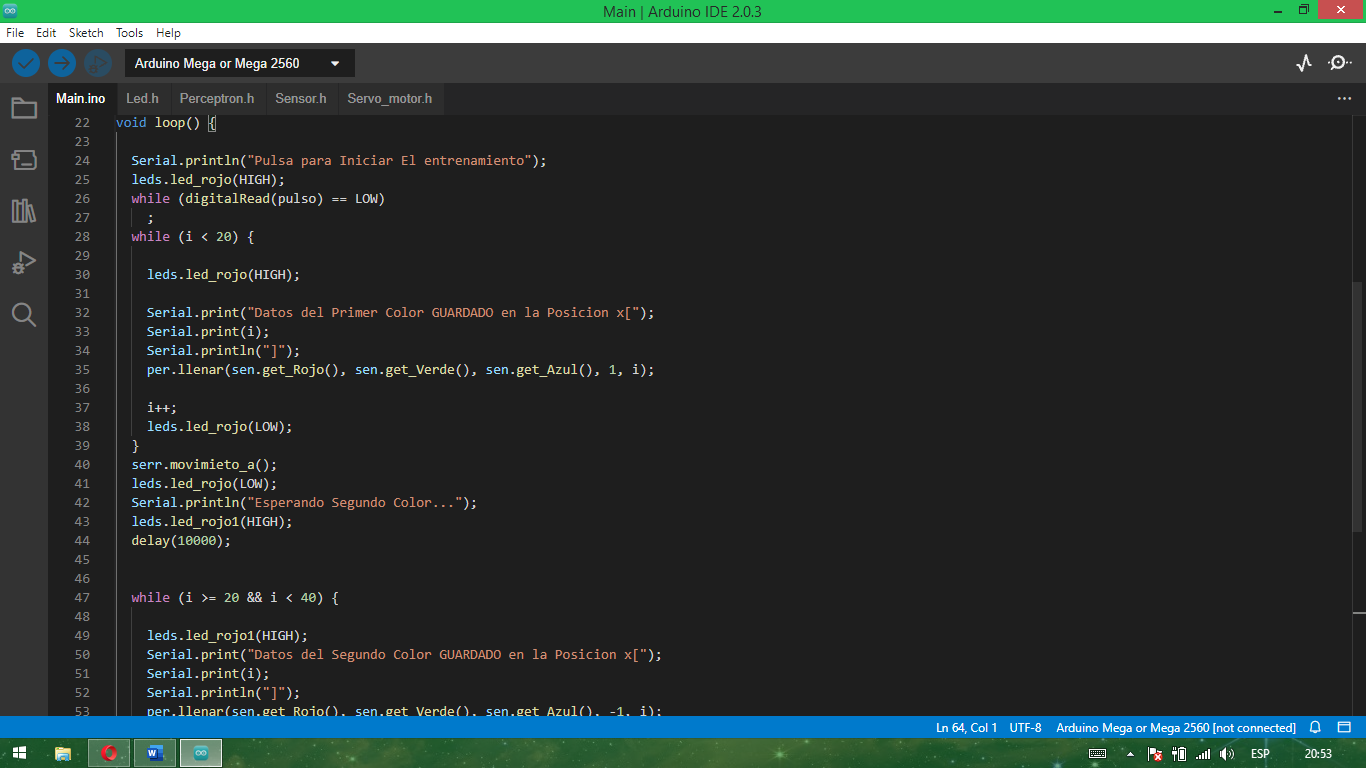
Ilustración 17. Clase Main.



***Fuente:*** *elaboración propia.*

En la siguiente ilustración comienza la ejecución de todo el código, el cual al presionar un botón se empieza a capturar los datos del sensor, estos datos corresponden al primer color dado que solo se almacenan 20 datos, para lograr esto se usa el objeto de la clase perceptrón y sensor. Para el objeto perceptrón se llama el método llenar y el objeto sensor los métodos que capturan los datos del sensor de color, línea 35. Estos métodos fueron descritos anteriormente.

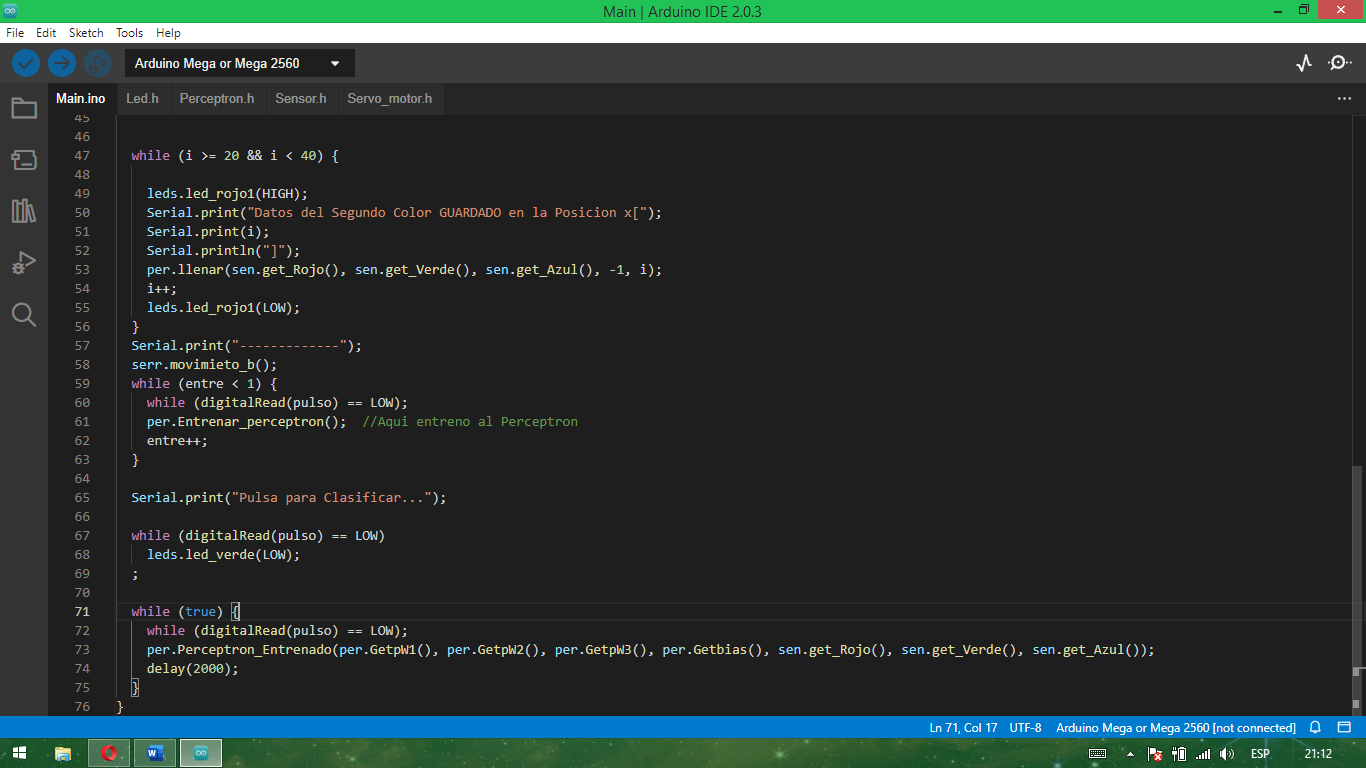
Ilustración 18. Datos del primer color



***Fuente:*** *elaboración propia.*

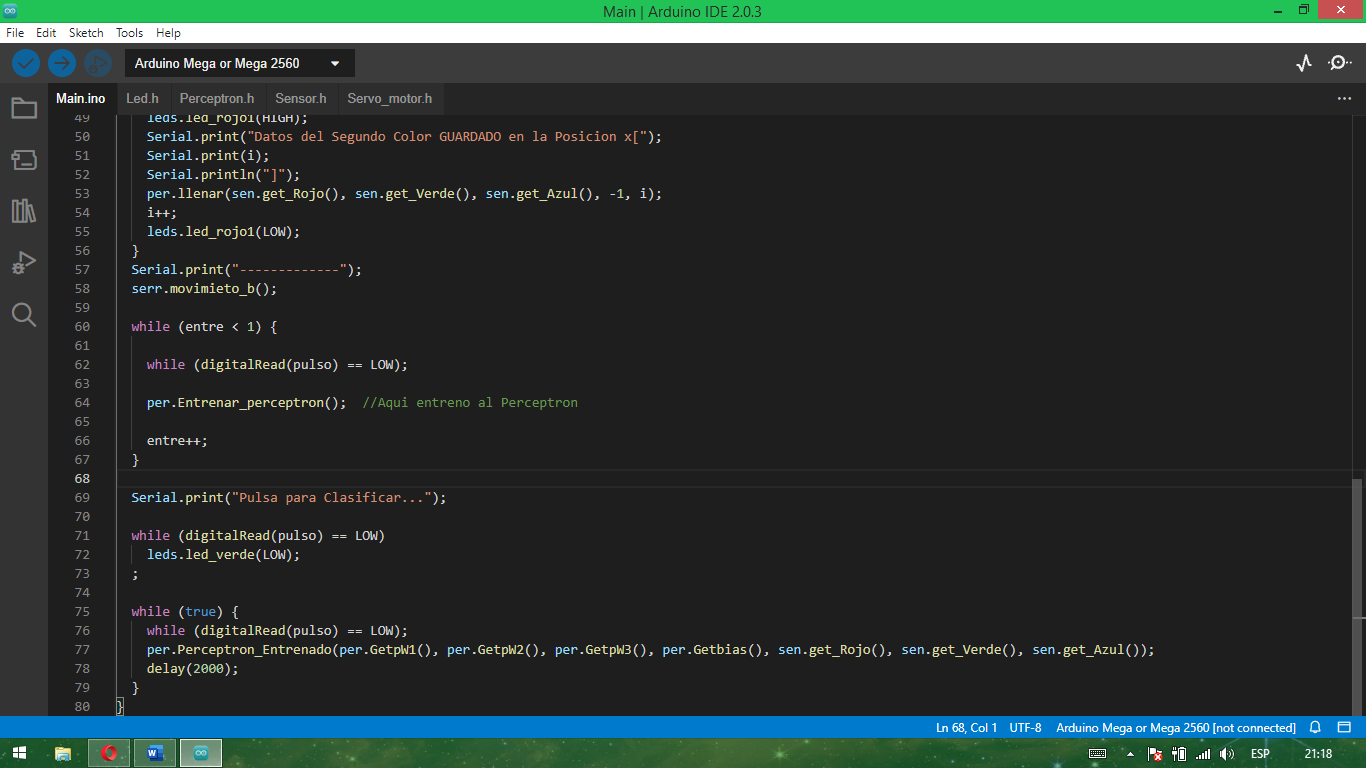
Luego de capturar los primeros datos se procede a capturar los datos del siguiente color que también fueron 20. Ver ilustración 19.

Ilustración 19. Datos de Segundo Color



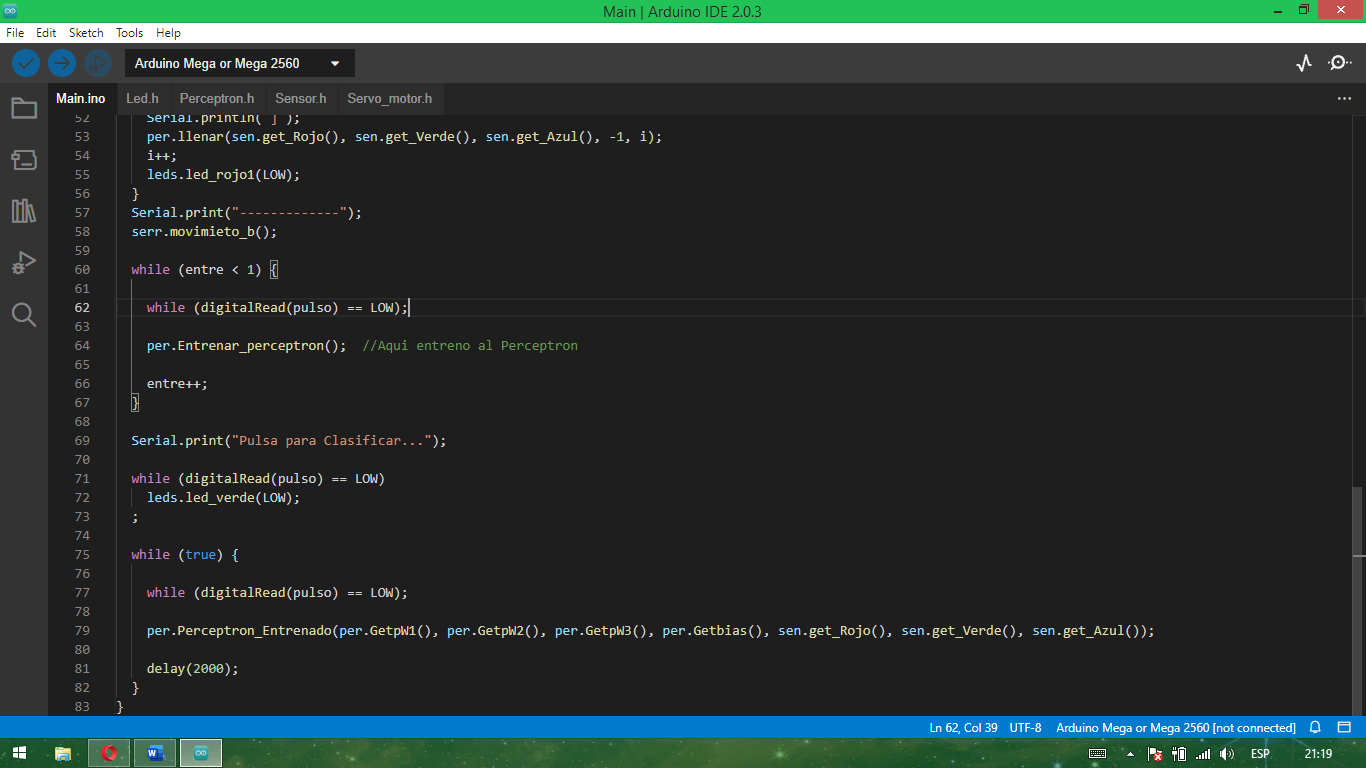
En la Ilustración 20 una vez capturados los datos de los colores se procede a entrenar el modelo

Ilustración 20. Entrenamiento Perceptrón



Ya por ultimo la ilustracion 21 es donde el modelo esta entrenado y solo recibe los pesos, bias y datos del color para clasificar los colores.

Ilustración 21. Clasificación del Color



# RESULTADOS DE ENTRENAMIENTO

En la tabla 1, da como resultado un máximo de 500 épocas realizadas por el perceptrón, los pesos al iniciar se manejan aleatoriamente al igual que el Bias.

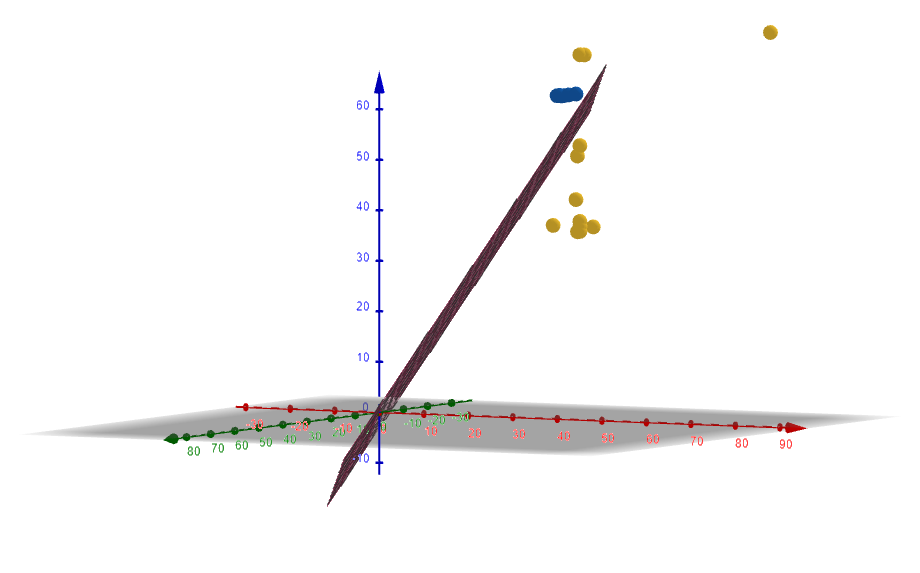
Tabla 1. Resultado del entrenamiento

|  |  |  |  |
| --- | --- | --- | --- |
| Épocas | 500 | | |
| Pesos Iniciales | 1 | 1 | 1 |
| Pesos Finales | 205,32 | -104,75 | 151,57 |
| Bias Inicial | 1 | | |
| Bias Final | -1,34 | | |

***Fuente:*** *elaboración propia.*

La ilustración 22 grafica los datos obtenidos por el sensor de color y el plano que fue resultado del entrenamiento.

Ilustración 22. Resultados del entrenamiento



***Fuente:*** *elaboración propia.*

# RESULTADOS DE LA CLASIFICACION DE MODELO FINAL

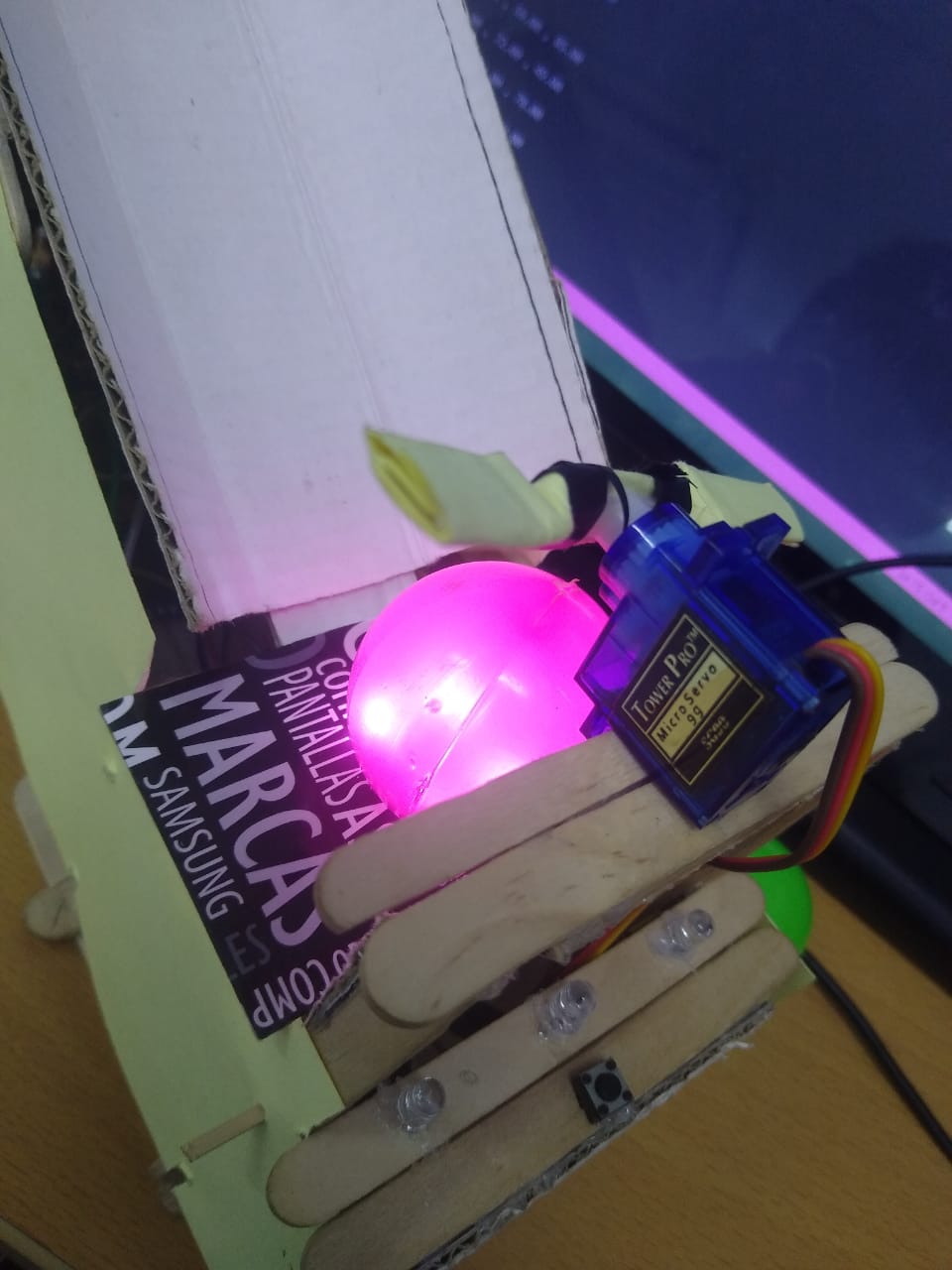
En la ilustración 23 se tomaron los datos del color rosa, los datos arrojas por el sensor fueron

R= 78, G = 59, B = 85

R= 85, G = 59, B = 86

R= 80, G = 59, B = 89

Ilustración 23. Color Rosa



***Fuente:*** *elaboración propia.*

En la ilustración 24 se tomaron los datos del color verde, los datos arrojas por el sensor fueron

R= 46, G = 69, B = 47

R= 41, G = 104, B = 70

Ilustración 24. Color verde.



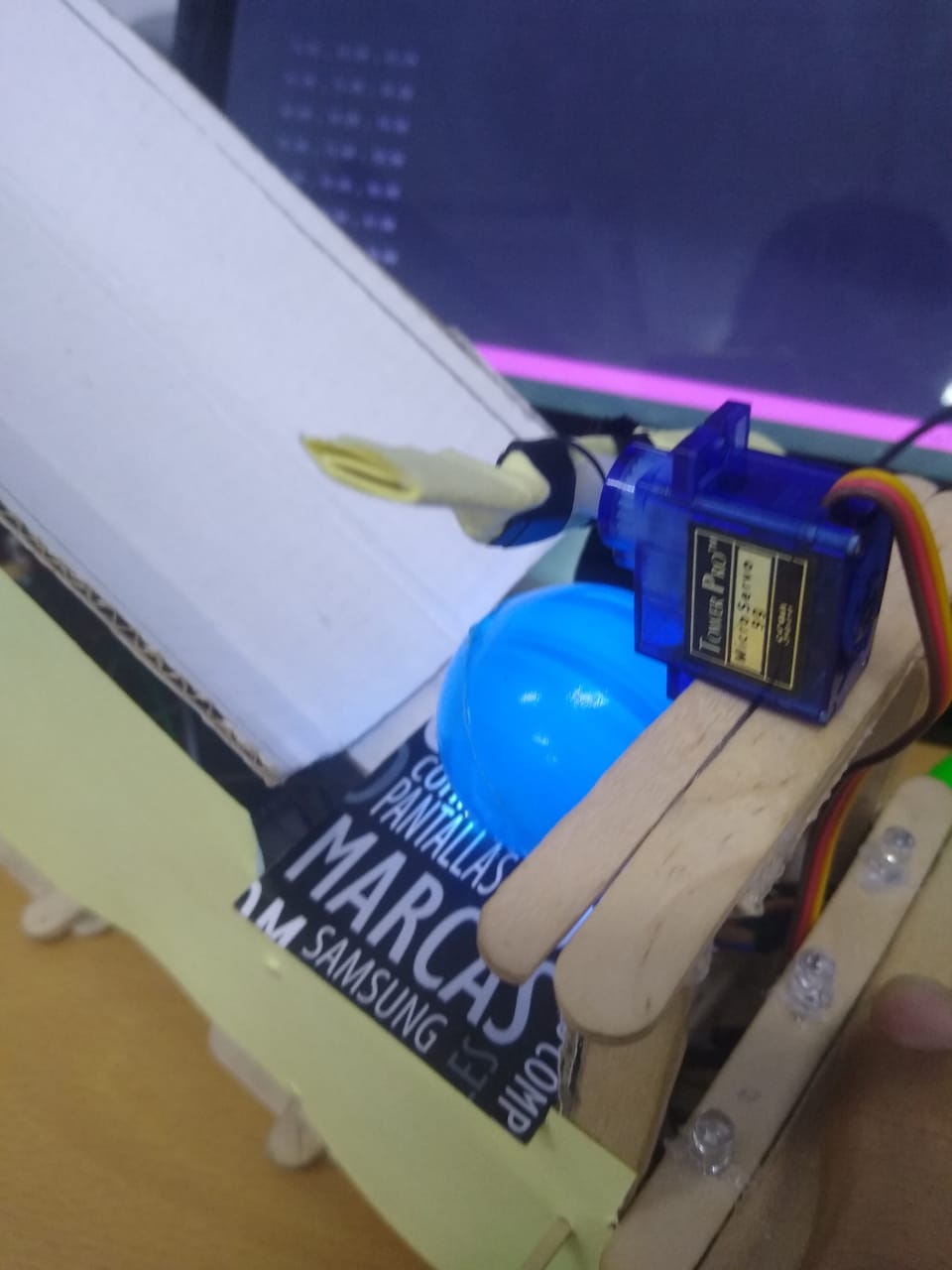
***Fuente:*** *elaboración propia.*

En la ilustración 25 se tomaron los datos del color azul, los datos arrojas por el sensor fueron

R = 95, G = 72, B = 49

R = 99, G = 83, B = 104

Ilustración 25. Color azul.



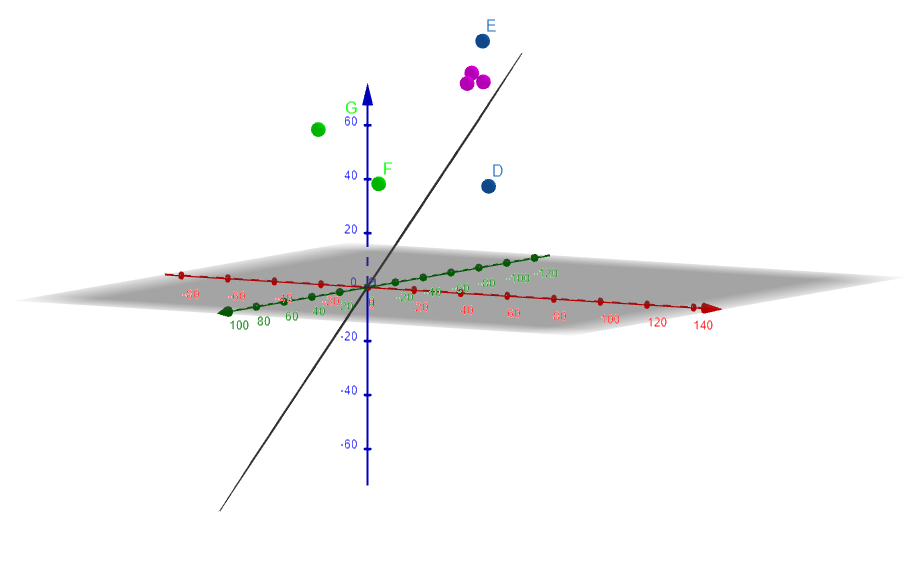
***Fuente:*** *elaboración propia.*

# ANÁLISIS DE LOS RESULTADOS

En primer lugar en la ilustración 22, el resultado del entrenamiento encontró la recta que separabas los datos, pero no logro separa los datos correctamente, dado que el valor arrojado por el sensor fue en el color amarillo un poco dispersos lo cual afecto el modelo dado que el entrenamiento llego al tope de iteraciones 500, aun así, los datos fueron separados en mayoría mediante recta que era el objetivo principal, aunque el modelo no fue 100% preciso de igual manera se debe tener un margen de error.

Por otro lado, los datos graficados en la ilustración 26, se evidencia que los datos de los colores no son muy distantes lo cual hace que no esté muy bien clasificados.

Ilustración 26. Datos Finales

 ***Fuente:*** *elaboración propia.*

# CONCLUSION

Según las gráficas se puede evidenciar la falta de entrenamiento del perceptrón debido a que este supero las iteraciones limites lo cual no logra ajustar los pesos, aunque el proceso de creación del modelo y el diseño del prototipo de interesante para el proceso de aprendizaje académico cabe resaltar que el modelo del perceptrón puede ser mejorado para su mejor clasificación.

# Bibliografía

Rouhiainen, L. (2018). *Inteligencia artificial : 101 cosas que debes saber hoy sobre nuestro futuro*. Alienta.

Soto-Peñaranda, H., Molano2, L. A., Zambrano-Arguello3, M., Mery, B., & Rodríguez4, R.-. (2019). *La inteligencia artificial: al alcance de la mano Artificial intelligence: at hand Resumen* (Vol. 6, Issue 12). Online.

Amat Rodrigo, J. (Septiembre de 2018). *Datos separable [Ilustracion]*. Perceptrón: https://www.cienciadedatos.net/documentos/50\_algoritmo\_perceptron#:~:text=El%20algoritmo%20Perceptron%20fue%20publicado,puede%20utilizarse%20para%20clasificaciones%20binarias.

Grupo.us. (Marzo de 2023). *Esquema Perceptrón [Ilustrcion]*. RedesNeuronales: http://grupo.us.es/gtocoma/pid/pid10/RedesNeuronales.htm

Wikipedia.org. (Marzo de 2023). Wikipedia: https://es.wikipedia.org/wiki/SketchUp

# ANEXOS

Construcción del prototipo con materiales caseros y componentes de Arduino para su posterior mecanización que nos permite separar de forma física un objeto con determinado color

