“Klasifikasi Jeruk Kunci dan Jeruk Santang Madu Berdasarkan Warna dan Berat Menggunakan Metode KNN”

Disusun oleh:

Abdan Idza Hurmuzi	195150300111015
Abimanyu Sri Setyo	195150300111005
Cut Fahrani Dhania	195150307111016
Ghifari Adlil Baqi	195150301111004

Sistem Pengenalan Pola
Kelas B

PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2021

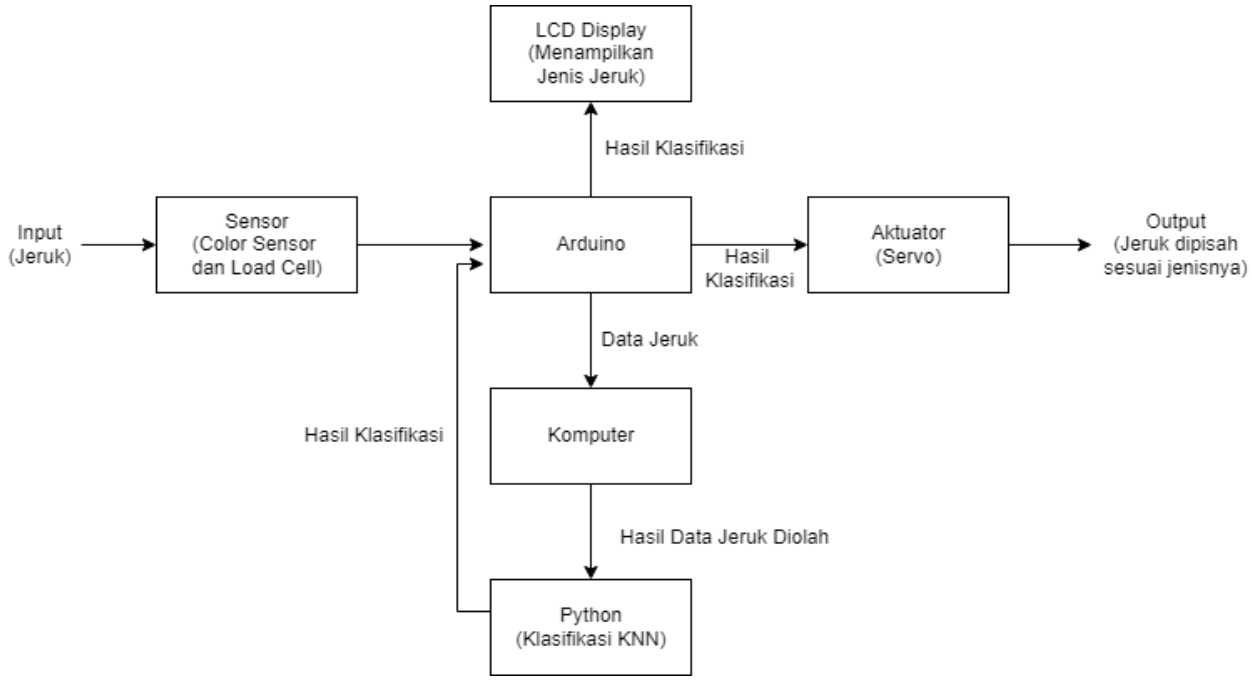
DAFTAR ISI

RINGKASAN PROJEK AKHIR

Pada projek akhir ini kami melakukan Klasifikasi terhadap Jeruk Santang Madu dan Jeruk Kunci yang mana pada input warna terbagi menjadi tiga spektrum warna, yaitu Red, Green, dan Blue. Sedangkan untuk input berat kami menggunakan satuan gram sebagai bahan ukur pada projek akhir ini. Hardware yang digunakan adalah Arduino Uno sebagai mikrokontroler, Servo SG90 untuk pergerakan arah kelas jeruk, Color Sensor TCS3200 untuk menampilkan spektrum warna yang terdapat pada objek, Load Cell untuk menimbang objek yang diuji, dan I2C LCD untuk menampilkan hasil objek yang telah diuji. Data yang digunakan sebagai data uji pada projek akhir ini adalah 50 yang terdapat pada tabel 3.1 dan metode pengerjaan yang digunakan adalah K-Nearest Neighbor dengan nilai K=5 karena nilai tersebut lebih akurat untuk diuji sehingga menghasilkan nilai akurasi sebesar 0,96 atau 96% (dalam persen) pada perhitungannya.

BAB I
BLOK DIAGRAM SISTEM

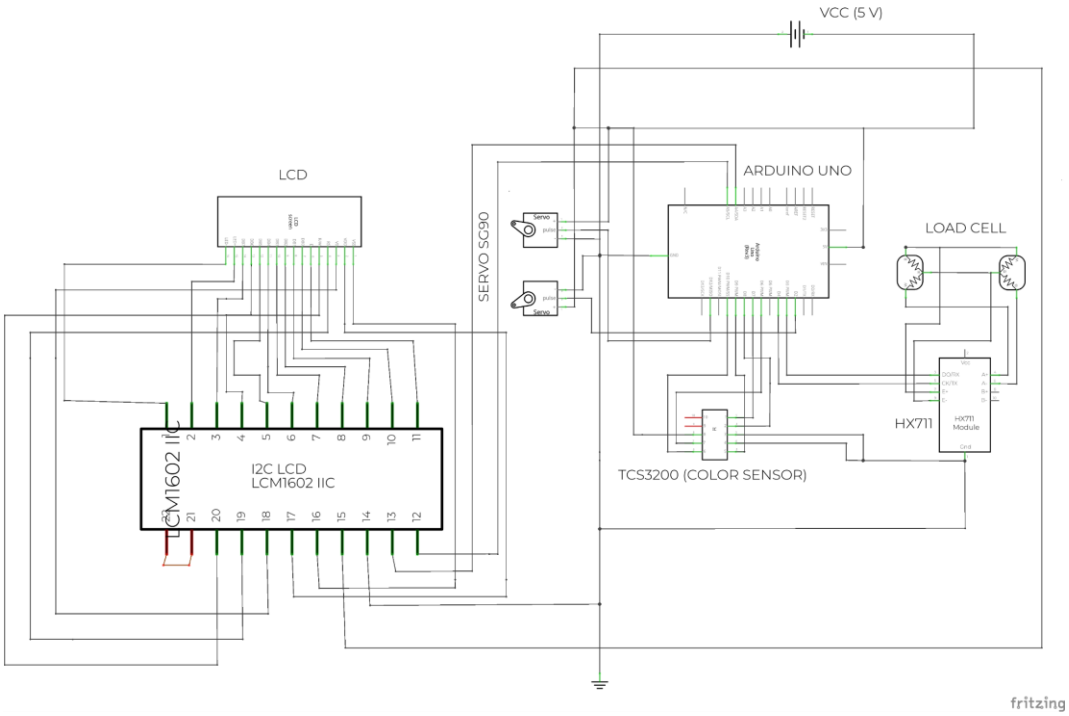
Blok Diagram Sistem Kasifikasi Jeruk Kunci dan Jeruk Santang Madu berdasarkan Warna dan Berat menggunakan Metode KNN dapat dilihat pada gambar berikut:



Gambar 1.1 Blok Diagram Sistem

BAB II
SPESIFIKASI PERANGKAT KERAS

2.1 Schematic Diagram



Gambar 2.1 Schematic Diagram

2.2 Tabel Pin

2.2.1 Color Sensor (TCS3200)

Arduino	Color Sensor TCS3200							
	S0	S1	S2	S3	OUT	OE	GND	VCC
Pin	7	8	9	10	6	GND	GND	5 V

2.2.2 HX711 (Load Cell)

Arduino	HX711 (Load Cell)			
	DOUT	SCK	GND	VCC
Pin	3	4	GND	5 V

2.2.3 Servo SG90

Arduino	Servo					
	Atas			Bawah		
	Signal	GND	VCC	Signal	GND	VCC
Pin	12	GND	5 V	2	GND	5 V

2.2.4 I2C LCD

Arduino	I2C LCD			
	SCL	SDA	GND	VCC
Pin	SCL	SDA	GND	5 V

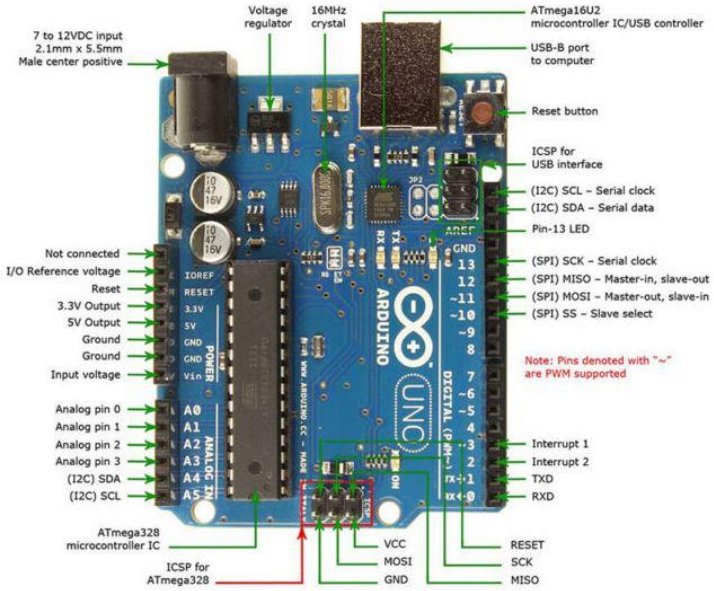
2.3 Hardware dan Konfigurasi Pin

2.3.1 Arduino Uno

Arduino Uno adalah board mikrokontroler berbasis ATmega328 (datasheet). Memiliki 14 pin input dari output digital dimana 6 pin input tersebut dapat digunakan sebagai output PWM dan 6 pin input analog, 16 MHz osilator kristal, koneksi USB, jack power, ICSP header, dan tombol reset. Untuk mendukung mikrokontroler agar dapat digunakan, cukup hanya menghubungkan Board Arduino Uno ke komputer dengan menggunakan kabel USB atau listrik dengan AC yang-ke adaptor-DC atau baterai untuk menjalankannya. Setiap 14 pin digital pada arduino uno dapat digunakan sebagai input dan output, menggunakan fungsi pinMode(), digitalwrite(), dan digitalRead(). Fungsi fungsi tersebut beroperasi di tegangan 5 volt, Setiap pin dapat memberikan atau menerima suatu arus maksimum 40 mA dan mempunyai sebuah resistor pull-up (terputus secara default) 20-50 kOhm.

Gambar 2.2 Arduino Uno

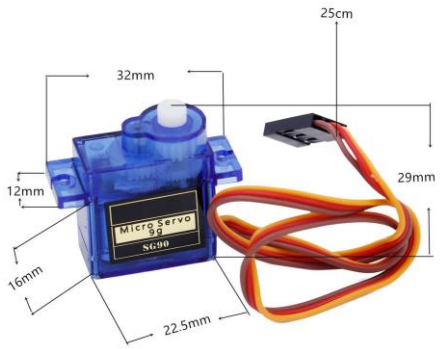
Berikut konfigurasi pin yang terdapat pada Arduino Uno beserta keterangan pada setiap pin nya yang terdapat pada gambar 2.3



Gambar 2.3 Konfigurasi Pin Arduino Uno

2.3.2 Servo SG90

Motor servo adalah motor DC dengan sistem umpan balik tertutup di mana posisi rotor-nya akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor ini terdiri dari sebuah motor DC, rangkaian gear, tensiometer, dan rangkaian kontrol. Potensiometer berfungsi sebagai penentu batas sudut dari putaran servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor servo. Alat ini juga dapat berotasi sekitar 180 derajat dan bisa bekerja seperti servo standar lainnya hanya saja ukurannya lebih kecil.

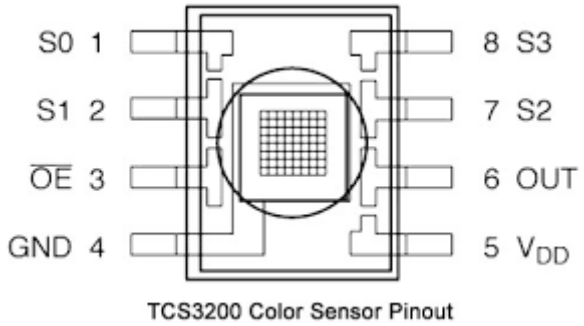


Gambar 2.4 Servo SG90

2.3.3 Color Sensor TCS3200

Sensor warna adalah sensor yang digunakan pada aplikasi mikrokontroler untuk pendeteksian suatu objek benda atau warna dari objek yang dimonitor. Salah satu jenis sensor warna yaitu **TCS 3200**. **TCS3200** merupakan konverter yang diprogram untuk mengubah warna menjadi frekuensi yang tersusun atas konfigurasi silicon photodiode dan konverter arus ke frekuensi dalam IC CMOS monolithic yang tunggal. Keluaran dari sensor ini adalah gelombang kotak (*duty cycle* 50%) frekuensi yang berbanding lurus dengan intensitas cahaya (*irradiance*).

Gambar 2.5 Sensor TCS3200



Gambar 2.6 Pin Sensor TCS3200

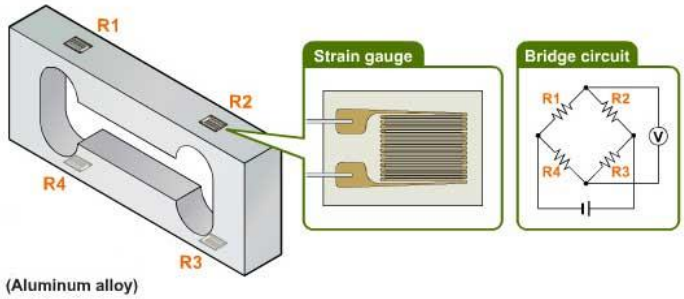
Sensor warna TCS 3200 memiliki konfigurasi pin dengan fungsi yang berbeda yang terdapat pada gambar 2.6 dan 2.7.

Nama	No Kaki IC	I/O	Fungsi Pin
GND	4	-	Sebagai Ground pada power supply
OE	3	I	Output enable, sebagai input untuk frekuensi output skala rendah
OUT	6	O	Sebagai output frekuensi
S0,S1	1,2	I	Sebagai saklar pemilih pada frekuensi output skala Tinggi
S2,S3	7,8	I	Sebagai saklar pemilih 4 kelompok dioda
Vdd	5	-	Supply tegangan

Gambar 2.7 Tabel PIN Sensor TCCS3200

2.3.4 Load Cell (5 Kg)

Sensor load cell merupakan sensor yang dirancang untuk mendeteksi tekanan atau berat sebuah beban, sensor load cell umumnya digunakan sebagai komponen utama pada sistem timbangan digital dan dapat diaplikasikan pada jembatan timbangan yang berfungsi untuk menimbang berat dari truk pengangkut bahan baku, pengukuran yang dilakukan oleh *Load Cell* menggunakan prinsip tekanan. Selama proses penimbangan akan mengakibatkan reaksi terhadap elemen logam pada load cell yang mengakibatkan gaya secara elastis. Gaya yang ditimbulkan oleh regangan ini dikonversikan kedalam sinyal elektrik oleh *strain gauge* (pengukur regangan) yang terpasang pada load cell. Prinsip kerja load cell berdasarkan rangkaian Jembatan Wheatstone.



Gambar 2.8 Load Cell

2.3.5 Kabel Jumper

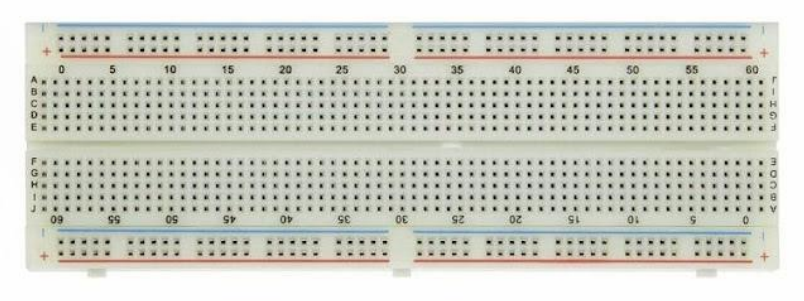
Kabel jumper adalah suatu istilah kabel yang ber-diameter kecil yang di dalam dunia elektronika digunakan untuk menghubungkan dua titik atau lebih dan dapat juga untuk menghubungkan 2 komponen elektronika. Fungsi kabel ini biasa digunakan untuk menghubungkan kabel dengan PCB dan juga komponen-komponen elektronik pada projek breadboard. Ada beberapa jenis kabel jumper yang dibedakan berdasarkan konektor kabelnya, yaitu: **Male-Male**, **Male-Female**, dan **Female-Female**.

Gambar 2.9 Kabel Jumper

2.3.6 Breadboard

Breadboard adalah board yang digunakan untuk membuat rangkaian elektronik sementara dengan tujuan uji coba atau prototipe tanpa harus menyolder. Dengan memanfaatkan breadboard, komponen-komponen elektronik yang dipakai tidak akan rusak dan dapat

digunakan kembali untuk membuat rangkaian yang lain.dan dengan demikian dapat digunakan untuk prototype sementara serta membantu dalam bereksperimen desain sirkuit elektronika.



Gambar 2.10 Breadboard

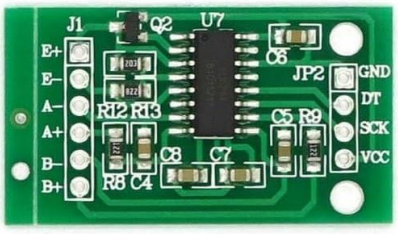
2.3.7 Kabel Type B

USB tipe-B adalah varian dari USB dipergunakan pada peripheral komputer, seperti pada Printer dan Scanner. Pada projek ini, kabel ini berfungsi untuk menyalurkan source code yang telah di-compile ke Arduino Uno.

Gambar 2.11 Kabel Type B

2.3.8 Modul HX711

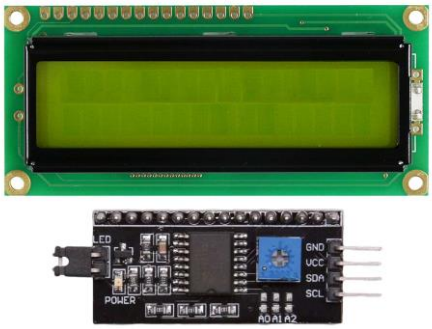
Merupakan modul amplifier (penguat sinyal) sekaligus modul Analog to Digital Converter (ADC) yang berfungsi untuk mengkondisikan sinyal analog dari sensor load cell sekaligus mengkonversikannya menjadi sinyal digital dan dihubungkan ke mikrokontroler maka kita dapat membaca perubahan resistansi dari load cell sehingga proses kalibrasi kita akan memperoleh pengukuran berat dengan keakuratan yang tinggi. Selain itu, modul yang memudahkan kita membaca load cell dalam pengukuran berat.



Gambar 2.12 Modul HX711

2.3.9 LCD I2C

LCD *(Liquid Crystal Display)* adalah suatu jenis media tampilan yang menggunakan kristal cair sebagai penampil utama. *LCD (Liquid Crystal Display)* bisa menampilkan suatu gambar/karakter dikarenakan terdapat banyak sekali titik cahaya (piksel) yang terdiri dari satu buah kristal cair sebagai titik cahaya. Pada LCD 16×2 pada umumnya menggunakan 16 pin sebagai kontrolnya, tentunya akan sangat boros apabila menggunakan 16 pin tersebut. Karena itu, digunakan driver khusus sehingga LCD dapat dikontrol dengan modul I2C atau Inter-Integrated Circuit. Dengan modul I2C, maka LCD 16x2 hanya memerlukan dua pin untuk mengirimkan data dan dua pin untuk pemasok tegangan.



Gambar 2.13 LCD I2C

Berikut konfigurasi pin yang terdapat pada LCD I2C beserta keterangan pada setiap pin nya.

Pin No	Symbol	Details
1	GND	Ground
2	Vcc	Supply Voltage +5V
3	Vo	Contrast adjustment
4	RS	0->Control input, 1-> Data input
5	R/W	Read/ Write
6	E	Enable
7 to 14	D0 to D7	Data
15	VB1	Backlight +5V
16	VB0	Backlight ground

Gambar 2.14 Tabel Konfigurasi LCD I2C

2.4 Dokumentasi Pengerjaan dan Implementasi Alat

2.4.1 Objek

- **Jeruk Kunci**

Gambar 2.15 Jeruk Kunci

- **Jeruk Santang Madu**

Gambar 2.16 Jeruk Santang Madu

2.4.2 Dokumentasi Pengerjaan

Gambar 2.17 Proses Pengerjaan Program

Gambar 2.18 Proses Pengerjaan Rangkaian Elektronika

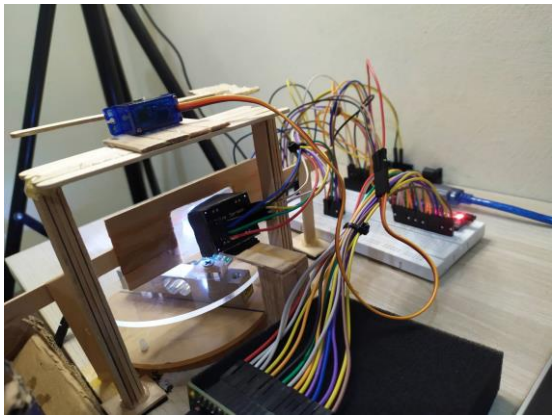
Gambar 2.19 Proses Pengerjaan

2.4.3 Implementasi Alat

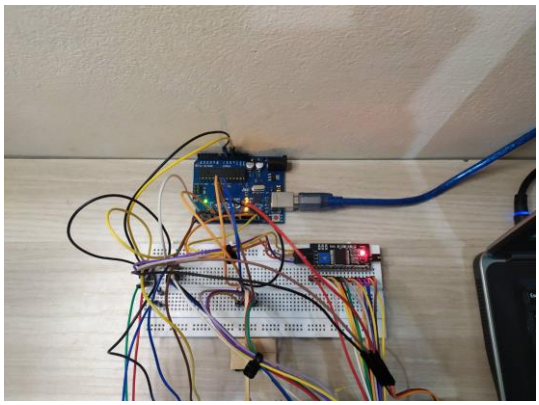
Gambar 2.20 Alat Pemisah Jeruk

Gambar 2.21 LCD memberikan informasi

Gambar 2.22 Kondisi Ruang Percobaan



Gambar 2.23 Rangkaian Sistem Sensor dan Aktuator



Gambar 2.24 Rangkaian Sistem Arduino

BAB III
DATA

3.1 Penjelasan Data

Data yang digunakan adalah Jeruk Santang Madu dan Jeruk Kunci. Dengan Berat dan Warna sebagai Variabel Input dan jenis jeruk, yakni Santang Madu dan Kunci sebagai Kelas Output. Data berjumlah 50 data, dan merupakan data input berupa data numerik dan data output berupa data kategorik. Semua data merupakan data primer. Selain itu, untuk pengumpulan datanya terdiri dari berat kedua jeruk tersebut dalam satuan gram, sedangkan warna yang diukur berdasarkan tiga spektrum warna (Red, Green, Blue). Pada masing-masing jeruk terdapat perpotongan warna antar jeruk, seperti, pada Jeruk Santang Madu terdapat jeruk yang mempunyai warna mirip dengan Jeruk Kunci dan pada Jeruk Kunci juga memiliki warna yang mirip dengan Jeruk Santang Madu.

3.2 Tabel Data

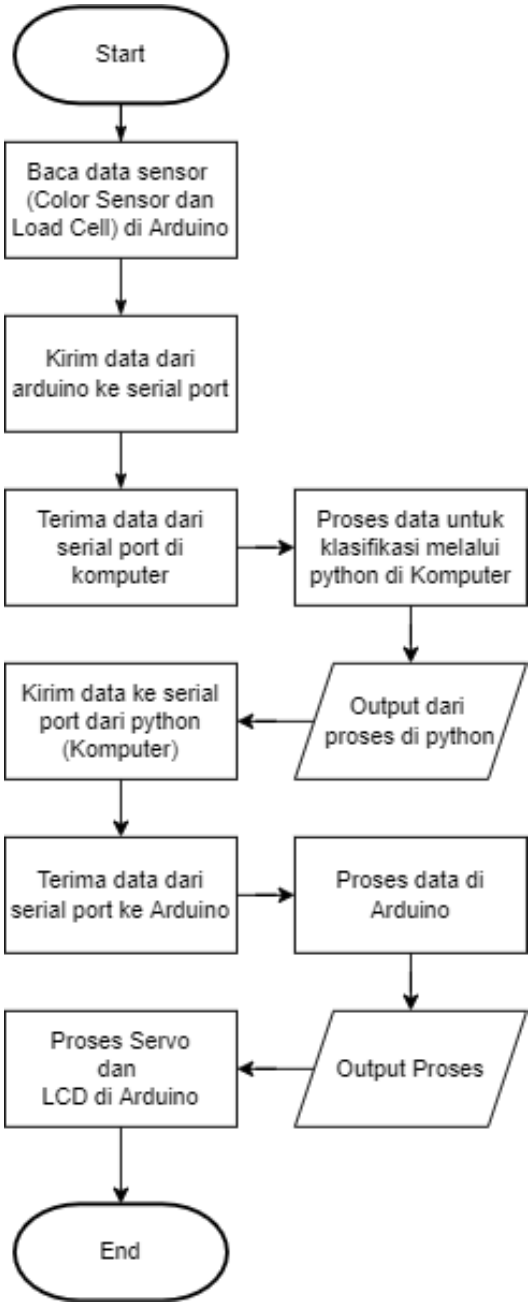
Object	Berat (gram)	R	G	B	Kelas
1	21	81	73	44	S.Madu
2	17	78	64	43	S.Madu
3	17	76	63	45	S.Madu
4	18	75	60	41	S.Madu
5	20	68	57	38	S.Madu
6	18	71	61	41	S.Madu
7	19	77	64	46	S.Madu
8	17	72	62	41	S.Madu
9	19	75	63	39	S.Madu
10	16	76	60	41	S.Madu
11	18	68	56	39	S.Madu
12	20	58	52	39	S.Madu
13	25	53	49	35	S.Madu
14	25	57	50	35	S.Madu
15	13	72	61	40	S.Madu
16	15	70	58	40	S.Madu
17	16	75	51	40	S.Madu
18	12	70	58	40	S.Madu
19	26	56	49	35	S.Madu
20	28	59	55	40	S.Madu
21	20	77	64	43	S.Madu

22	18	79	65	47	S.Madu
23	14	76	62	39	S.Madu
24	23	75	60	40	S.Madu
25	11	51	48	35	Kunci
26	9	48	44	37	Kunci
27	11	53	50	35	Kunci
28	11	52	48	36	Kunci
29	11	51	50	35	Kunci
30	12	50	46	36	Kunci
31	11	46	39	28	Kunci
32	11	52	45	36	Kunci
33	12	50	44	34	Kunci
34	10	49	45	39	Kunci
35	12	52	50	40	Kunci
36	11	50	48	34	Kunci
37	11	48	44	33	Kunci
38	11	51	48	39	Kunci
39	11	51	45	35	Kunci
40	10	54	54	38	Kunci
41	10	52	49	40	Kunci
42	10	52	46	36	Kunci
43	11	53	50	38	Kunci
44	10	52	49	36	Kunci
45	10	50	49	35	Kunci
46	8	44	40	38	Kunci
47	11	54	50	36	Kunci
48	9	69	56	39	Kunci
49	10	71	59	50	Kunci
50	10	53	49	36	Kunci

BAB IV
METODE

4.1 Flowchart

Flowchart Sistem Kasifikasi Jeruk Kunci dan Jeruk Santang Madu berdasarkan Warna dan Berat menggunakan Metode KNN dapat dilihat di gambar berikut.



Gambar 4.1 Flowchart Sistem

4.2 K-Nearest Neighbor (KNN)

Metode yang digunakan adalah Metode Klasifikasi dengan menggunakan Algoritma K-Nearest Neighbor. Algoritma K-Nearest Neighbor adalah algoritma supervised learning dimana hasil dari instance yang baru diklasifikasikan berdasarkan mayoritas dari kategori k-tetangga terdekat.

Tujuan dari algoritma ini adalah untuk mengklasifikasikan objek baru berdasarkan atribut dan sampel-sampel dari training data. Algoritma KNN (k-Nearest Neighbor) ini adalah algoritma klasifikasi berdasarkan tetangga terdekat. Selain itu, Algoritma K-Nearest Neighbor menggunakan Neighborhood Classification sebagai nilai prediksi dari nilai instance yang baru.

Di sistem ini, pada fase *training* algoritma pada sistem akan melakukan penyimpanan vektor-vektor fitur dan klasifikasi dari *data training*. Kemudian, pada fase klasifikasi fitur-fitur yang sama dihitung untuk mendapatkan *data test* yang merupakan data yang kelas klasifikasinya belum diketahui. Jarak dari vektor yang baru tersebut terhadap *data training* akan dihitung dan diambil nilai sejumlah K yang paling mendekati. Kelas klasifikasi *data test* yang baru akan didapatkan dari nilai voting antara sejumlah K yang digunakan dimana klasifikasinya didapat dengan hasil voting kelas terbanyak dari kelas tetangganya. Pada projek ini kami menggunakan nilai K=5 karena menghasilkan nilai akurat yang lebih tinggi saat diproses data yang diujinya.

4.3 Kode Program Arduino IDE dan Python

4.3.1 Arduino

No	arduino
1	`#include "HX711.h"`
2	`#include <LiquidCrystal_I2C.h>`
3	`#include <Servo.h>`
4	
5	`//Pin Define`
6	`#define LOADCELL_DOUT_PIN  3`
7	`#define LOADCELL_SCK_PIN  4`
8	`#define SERVO_ATAS 12`
9	`#define SERVO_BAWAH 2`
10	`#define S0_CS 7`
11	`#define S1_CS 8`
12	`#define S2_CS 9`
13	`#define S3_CS 10`
14	`#define OUT_CS 6`
15	
16	`// Class object init`
17	`HX711 scale;`
18	`LiquidCrystal_I2C lcd(0x27, 16, 2);`
19	`Servo myServo_bot, myServo_top;`
20	
21	`//Calibration State`
22	`//Load Cell`
23	`float calibration_factor = 360;`
24	`//Color Sensor`
25	`int redMin = 76;`
26	`int greenMin = 86;`
27	`int blueMin = 55;`
28	`int redMax = 320;`
29	`int greenMax = 348;`
30	`int blueMax = 233;`
31	

```
32  //Variables for Color Pulse Width Measurements
33  int redPW = 0;
34  int greenPW = 0;
35  int bluePW = 0;
36
37  int redValue = 0;
38  int greenValue = 0;
39  int blueValue = 0;
40
41  byte average = 0;
42  char output;
43  int i = 0;
44
45  byte state = 0;
46  static unsigned long t_awal;
47  static unsigned long t_setelah;
48  static unsigned long treshold = 2000;
49
50  void send_data(int red, int green, int blue, int w){
51    delay(500);
52    Serial.println('s');
53    String data = "";
54    data = (String(w) + "," + String(red) + "," + String(green) + "," +
55  String(blue));
56    delay(500);
57    Serial.println(data);
58    while(!(Serial.read() == 'f'));
59  }
60
61  void reset_all(){
62    lcd.clear();
63    lcd.setCursor(0,0);
64    redValue = 0;
65    greenValue = 0;
66    blueValue = 0;
67    myServo_bot.write(90);
68    myServo_top.write(00);
69    i = 0;
70  }
71
72  void process_bottom_servo(char output){
73    delay(500);
74    (output == 'S') ? myServo_bot.write(90 - 70) : myServo_bot.write(90
75  + 70);
76    delay(500);
77  }
78
79  void servo_init(){
80    myServo_bot.attach(2);
81    myServo_top.attach(12);
82    myServo_top.write(00);
83    myServo_bot.write(90);
84  }
85
86  void process_top_servo(){
87    delay(500);
88    myServo_top.write(150);
89    delay(500);
90    myServo_top.write(00);
91  }
92
93  char receive_data(){
94    String x;
95    char jenis;
96    while(!Serial.available());
97    x = Serial.readStringUntil('\n');
98    (x == "s") ? jenis = 'S' : jenis = 'K';
```

```
  return jenis;
}

void readRGB(){
  for (int i = 0; i < 5; i++)
  {
       redPW = getRedPW();
       redValue = map(redPW, redMin, redMax, 255, 0);
       if(redValue < 0) redValue = 0;
       delay(500);

       // Read Green Pulse Width
       greenPW = getGreenPW();
       greenValue = map(greenPW, greenMin, greenMax, 255, 0);
       if(greenValue < 0)         greenValue = 0;
       delay(500);
       // Read Blue Pulse Width
       bluePW = getBluePW();
       blueValue = map(bluePW, blueMin, blueMax, 255, 0);
       if(blueValue < 0)   blueValue = 0;
       delay(500);

       redValue += redValue;
       greenValue += greenValue;
       blueValue += blueValue;
  }
  redValue /= 5;
  greenValue /= 5;
  blueValue /= 5;
}

int cekWeight(){
  int berat = 0;
  scale.set_scale(calibration_factor);
  berat = scale.get_units();
  return berat;
}

int readWeight(){
  average = 0;
  scale.set_scale(calibration_factor);
  for(int i = 0; i < 5; i++){
       average += scale.get_units();
  }
  average /= 5;
  return average;
}

void setup_LCD(){
  lcd.begin();
  lcd.backlight();
}

void setup_loadCell(){
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  scale.set_scale();
  scale.tare(); //Reset the scale to 0
  long zero_factor = scale.read_average(); //Get a baseline reading
  Serial.println("Zero factor: ");
  Serial.println(zero_factor);
}

void setup_colorSensor(){
  // Set S0 - S3 as outputs
  pinMode(S0_CS, OUTPUT);
  pinMode(S1_CS, OUTPUT);
  pinMode(S2_CS, OUTPUT);
```

```
  pinMode(S3_CS, OUTPUT);
  // Set Pulse Width scaling to 20%
  digitalWrite(S0_CS,HIGH);
  digitalWrite(S1_CS,LOW);
  // Set Sensor output as input
  pinMode(OUT_CS, INPUT);
}

int getGreenPW() {
  // Set sensor to read Green only
  digitalWrite(S2_CS,HIGH);
  digitalWrite(S3_CS,HIGH);
  int PW;
  PW = pulseIn(OUT_CS, LOW);
  return PW;
}

int getBluePW() {
  // Set sensor to read Blue only
  digitalWrite(S2_CS,LOW);
  digitalWrite(S3_CS,HIGH);
  int PW;
  PW = pulseIn(OUT_CS, LOW);
  return PW;
}

int getRedPW() {
  // Set sensor to read Red only
  digitalWrite(S2_CS,LOW);
  digitalWrite(S3_CS,LOW);
  int PW;
  PW = pulseIn(OUT_CS, LOW);
  return PW;
}

void FSM(){
  t_setelah = millis() - t_awal;
  switch(state){
       case 0:
       if(i == 0){
         lcd.print("Put object on");
         lcd.setCursor(0,1);
         lcd.print("top of Load Cell");
         i++;
       }
       Serial.println("Put object on top of load cell");
       if(cekWeight() > 2 && t_setelah > treshold * 4){
         lcd.clear();
         lcd.setCursor(0,0);
         lcd.print("Dont move object");
         lcd.setCursor(0,1);
         lcd.print("Processing...");
         state = 1;
         t_awal = millis();
       }
       break;

       case 1:
       if(t_setelah > treshold && cekWeight() > 2){
         lcd.clear();
         lcd.setCursor(0,0);
         lcd.print("Weight:");
         Serial.println("Begin reading weight...");
         Serial.println("Reading weight...");
         int w = readWeight();
         Serial.print("Weight: ");
         Serial.print(String(w) + "gram");
```

```
          lcd.print(String(w) + " Gram");
          delay(2000);
          state = 2;
          t_awal = millis();
        }
        break;

        case 2:
        if(t_setelah > treshold && cekWeight() > 2){
          lcd.clear();
          lcd.setCursor(0,0);
          lcd.print("Color:");
          Serial.println("Begin reading color...");
          Serial.println("Reading color...\n");
          Serial.print("Color:\n");
          readRGB();
          lcd.setCursor(0,1);
          lcd.print("R:");
          lcd.print(String(redValue));
          delay(1000);
          lcd.print(",G:");
          lcd.print(String(greenValue));
          delay(1000);
          lcd.print(",B:");
          lcd.print(String(blueValue));
          delay(3000);
          state = 3;
          t_awal = millis();
        }
        break;

        case 3:
        if(t_setelah > treshold){
          lcd.clear();
          lcd.setCursor(0,0);
          lcd.print("Begin Classifica-");
          lcd.setCursor(0,1);
          lcd.print("tion...");
          Serial.println("Begin classification...");
          send_data(redValue, greenValue, blueValue, average);
          output = receive_data();
          if(output == 'S'){
          lcd.clear();
          lcd.setCursor(0,0);
          lcd.print("Jeruk");
          lcd.setCursor(0,1);
          lcd.print("Santang Madu");
        } else {
          lcd.clear();
          lcd.setCursor(0,0);
          lcd.print("Jeruk");
          lcd.setCursor(0,1);
          lcd.print("Kunci");
        }
         state = 4;
         t_awal = millis();
        }
       break;

        case 4:
        if((t_setelah > treshold) && (output == 'K' || output ==
'S')){
          Serial.print("Servo Bottom move...");
          process_bottom_servo(output);
          state = 5;
          t_awal = millis();
        }
```

```
300       break;
301
302        case 5:
303        if(t_setelah > treshold){
304          Serial.print("Servo Top move...");
305          process_top_servo();
306          state = 6;
307          t_awal = millis();
308        }
309       break;
310
311        case 6:
312        if(t_setelah > treshold * 2){
313        Serial.print("End of process...");
314        if(cekWeight() < 3){
315        reset_all();
316        state = 0;
317        t_awal = millis();
318        }
319        }
320        break;
321        }
322   }
323
324 void setup(){
325   Serial.begin(9600);
326   Serial.setTimeout(1);
327   while(!Serial.available()) {
328        Serial.println("waiting to start...");
329        delay(1000);
330   }
331   setup_LCD();
332   lcd.setCursor(0,0);
333   Serial.println("LCD OK...");
334   delay(1000);
335   lcd.print("Setting Up...");
336   Serial.println("Setting up...");
337   delay(1000);
338   setup_loadCell();
339   Serial.println("Load Cell OK...");
340   delay(1000);
341   setup_colorSensor();
342   Serial.println("Color Sensor OK...");
343   delay(1000);
344   servo_init();
345   Serial.println("Servo OK...");
346   delay(1000);
347   t_awal = millis();
348   lcd.clear();
349   lcd.print("All set, Ready!");
350   delay(2000);
351   Serial.println("All set, Ready!");
352   delay(100);
353   lcd.clear();
354 }
355
356 void loop(){
357   FSM();
358 }
```

Blok program 1

```
 1  #include "HX711.h"
 2  #include <LiquidCrystal_I2C.h>
 3  #include <Servo.h>
 4
 5  //Pin Define
 6  #define LOADCELL_DOUT_PIN  3
 7  #define LOADCELL_SCK_PIN  4
 8  #define SERVO_ATAS 12
 9  #define SERVO_BAWAH 2
10  #define S0_CS 7
11  #define S1_CS 8
12  #define S2_CS 9
13  #define S3_CS 10
14  #define OUT_CS 6
15
16  // Class object init
17  HX711 scale;
18  LiquidCrystal_I2C lcd(0x27, 16, 2);
19  Servo myServo_bot, myServo_top;
20
21  //Calibration State
22  //Load Cell
23  float calibration_factor = 360;
24  //Color Sensor
25  int redMin = 76;
26  int greenMin = 86;
27  int blueMin = 55;
28  int redMax = 320;
29  int greenMax = 348;
30  int blueMax = 233;
31
32  //Variables for Color Pulse Width Measurements
33  int redPW = 0;
34  int greenPW = 0;
35  int bluePW = 0;
36
37  int redValue = 0;
38  int greenValue = 0;
39  int blueValue = 0;
40
41  byte average = 0;
42  char output;
43  int i = 0;
44
45  byte state = 0;
46  static unsigned long t_awal;
47  static unsigned long t_setelah;
48  static unsigned long treshold = 2000;
```

```
1   Memanggil library untuk driver(HX711) dari Load Cell.
2   Memanggil library untuk interface I2C LCD
3   Memanggil library untuk Servo
4
5   //Pin Define
6   Mendefinisikan pin 3 pada arduino sebagai DOUT pin pada modul HX711
    (driver load cell)
7   Mendefinisikan pin 4 pada arduino sebagai SCK pin pada modul HX711
    (driver load cell)
8   Mendefinisikan pin 12 pada arduino sebagai signal input ke servo atas
9   Mendefinisikan pin 2 pada arduino sebagai signal input ke servo bawah
10  Mendefinisikan pin 7 pada arduino sebagai S0 pada Sensor warna
    (TCS3200)
11  Mendefinisikan pin 8 pada arduino sebagai S1 pada Sensor warna
```

```
     (TCS3200)
12   Mendefinisikan pin 9 pada arduino sebagai S2 pada Sensor warna
     (TCS3200)
13   Mendefinisikan pin 10 pada arduino sebagai S3 pada Sensor warna
     (TCS3200)
14   Mendefinisikan pin 6 pada arduino sebagai OUT pada Sensor warna
     (TCS3200)
15
16   // Class object init
17   Instance object dari class pada library HX711
18   Instance object dari class pada library I2C LCD
19   Instance 2 object untuk servo atas dan bawah dari class pada library
     Servo
20
21   //Calibration State
22   //Load Cell
23   Inisialiasi nilai calibration factor untuk menghitung berat (gram)
24   //Color Sensor
25   Inisialiasi nilai calibration red minimum pada color sensor
26   Inisialiasi nilai calibration green minimum pada color sensor
27   Inisialiasi nilai calibration blue minimum pada color sensor
28   Inisialiasi nilai calibration red maximum pada color sensor
29   Inisialiasi nilai calibration green maximum pada color sensor
30   Inisialiasi nilai calibration blue maximum pada color sensor
31
32   //Variables for Color Pulse Width Measurements
33   Inisialiasi variabel untuk menyimpan nilai red pulse width dari color
     sensor
34   Inisialiasi variabel untuk menyimpan nilai green pulse width dari
     color sensor
35   Inisialiasi variabel untuk menyimpan nilai blue pulse width dari
     color sensor
36
37   Inisialiasi variabel untuk menyimpan nilai red dari color sensor
38   Inisialiasi variabel untuk menyimpan nilai green dari color sensor
39   Inisialiasi variabel untuk menyimpan nilai blue dari color sensor
40
41   Inisialiasi variabel untuk menyimpan nilai average berat yang dibaca
42   Inisialiasi variabel untuk menyimpan nilai output hasil proses
     klasifikasi
43   Inisialiasi variabel bantu untuk penanda pada salah satu proses di
     FSM()
44
45   Inisialiasasi variabel untuk state yang akan digunakan pada fungsi
     FSM()
46   Inisialiasi variabel untuk menyimpan nilai millis untuk proses pada
     FSM()
47   Inisialiasi variabel untuk menyimpan nilai millis untuk proses pada
     FSM()
48   Inisialiasi nilai treshold dalam millis untuk proses pada FSM()
```

Blok program 2

```
50   void send_data(int red, int green, int blue, int w){
51     delay(500);
52     Serial.println('s');
53     String data = "";
54     data = (String(w) + "," + String(red) + "," + String(green) + "," +
55   String(blue));
56     delay(500);
57     Serial.println(data);
58     while(!(Serial.read() == 'f'));
59   }
```

```
50  Mendeklarasikan fungsi untuk mengirim data ke python (komputer)
    melalui serial port.
51  Delay 500 ms.
52  Mengirim data di serial port ke python untuk menandakan bahwa arduino
    akan mengirim data (weight, R, G, B)
53  Mendeklarasikan variabel untuk menyimpan data
54  Membuat data dalam string dengan menambahkan tiap data warna dan
55  berat untuk menjadi string dengan format ex: “22,72,62,48”
56  Delay 500 ms
57  Menampilkan data pada serial port
58  Menunggu python (komputer) selesai memproses data (klasifikasi),
    apabila sudah maka lanjut ke proses berikutnya
59  Menutup fungsi
```

Blok program 3

```
61  void reset_all(){
62    lcd.clear();
63    lcd.setCursor(0,0);
64    redValue = 0;
65    greenValue = 0;
66    blueValue = 0;
67    myServo_bot.write(90);
68    myServo_top.write(00);
69    i = 0;
70  }
```

```
61  Mendeklarasikan fungsi untuk mereset semua state hardware dan nilai
    pada variabel
62  Memanggil fungsi untuk membersihkan layar lcd
63  Memanggil fungsi untuk posisi cursor layar lcd
64  Memberi nilai 0 pada variabel red (color sensor)
65  Memberi nilai 0 pada variabel green (color sensor)
66  Memberi nilai 0 pada variabel blue (color sensor)
67  Memanggil fungsi write pada servo atas untuk kembali ke posisi semula
68  Memanggil fungsi write pada servo bawah untuk kembali ke posisi
    semula
69  Memberi nilai 0 pada variabel i
70  Menutup fungsi reset_all
```

Blok program 4

```
72  void process_bottom_servo(char output){
73    delay(500);
74    (output == 'S') ? myServo_bot.write(90 - 70) : myServo_bot.write(90
75  + 70);
76    delay(500);
77  }
78
79  void servo_init(){
80    myServo_bot.attach(2);
81    myServo_top.attach(12);
82    myServo_top.write(00);
83    myServo_bot.write(90);
84  }
85
86  void process_top_servo(){
87    delay(500);
88    myServo_top.write(150);
89    delay(500);
90    myServo_top.write(00);
91  }
```

```
 72  Mendeklarasikan fungsi untuk memproses servo atas
 73  Delay 500 ms
 74  Melakukan seleksi kondisi untuk pergerakan servo bawah
 75  (sorter/pemilah) (apabila jeruk kunci servo akan mengarakan ke kanan,
     dan sebaliknya)
 76  Delay 500 ms
 77  Menutup fungsi process_bottom_servo
 78
 79  Mendeklarasikan fungsi untuk menginisialisasi masing-masing servo
 80  Memanggil fungsi untuk mengaktifkan servo pada pin 2 (servo bawah)
 81  Memanggil fungsi untuk mengaktifkan servo pada pin 12 (servo atas)
 82  Memanggil fungsi untuk posisi awal servo 0 derajat (servo atas)
 83  Memanggil fungsi untuk posisi awal servo 90 derajat (servo bawah)
 84  Menutup fungsi servo_init
 85
 86  Mendeklarasikan fungsi untuk memproses servo atas
 87  Delay 500 ms
 88  Memanggil fungsi untuk posisi servo atas menendang buah jeruk
 89  Delay 500 ms
 90  Memanggil fungsi untuk posisi servo mereset ke posisi awal setelah
     menendang
 91  Menutup fungsi process_top_servo
```

Blok program 5

```
 93  char receive_data(){
 94    String x;
 95    char jenis;
 96    while(!Serial.available());
 97    x = Serial.readStringUntil('\n');
 98    (x == "s") ? jenis = 'S' : jenis = 'K';
 99    return jenis;
100  }
```

```
 93  Mendeklarasikan fungsi untuk menerima data dari python (komputer)
 94  Mendeklarasikan variabel x untuk menyimpan data yang masuk dari
     serial port
 95  Mendeklarasikan variabel untuk jenis jeruk
 96  Menunggu sampai data dikirimkan ke serial port
 97  Memberi nilai x sebagai data yang masuk ke serial port dari python
     (komputer) dengan memanggil fungsi untuk membaca string sampai '\n'
 98  Ternary operator untuk seleksi kondisi nilai untuk variabel jenis
     jeruk berdasarkan nilai x (data yang masuk)
 99  Mengembalikan fungsi dengan nilai pada variabel jenis (jenis jeruk)
100  Menutup fungsi receive_data
```

Blok program 6

```
102  void readRGB(){
103    for (int i = 0; i < 5; i++){
104         redPW = getRedPW();
105         redValue = map(redPW, redMin, redMax, 255, 0);
106         if(redValue < 0) redValue = 0;
107         delay(500);
108
109         // Read Green Pulse Width
110         greenPW = getGreenPW();
111         greenValue = map(greenPW, greenMin, greenMax, 255, 0);
112         if(greenValue < 0) greenValue = 0;
113         delay(500);
```

```
114
115         // Read Blue Pulse Width
116         bluePW = getBluePW();
117         blueValue = map(bluePW, blueMin, blueMax, 255, 0);
118         if(blueValue < 0)   blueValue = 0;
119         delay(500);
120
121         redValue += redValue;
122         greenValue += greenValue;
123         blueValue += blueValue;
124     }
125     redValue /= 5;
126     greenValue /= 5;
127     blueValue /= 5;
128 }
```

```
102 Mendeklarasikan fungsi untuk mengambil data RGB dari color sensor
103 Membuat perulangan sebanyak 5 kali untuk mencari nilai rata-rata dari
    pembacaan sensor selama 5 kali
104 Menyimpan nilai red pulse width dari fungsi getRedPW() pada variabel
    redPW
105 Memberi nilai pada variabel redValue (red) dengan fungsi map yang
    berisi parameter yaitu: pulse width, red minimum, red maximum, 255
    dan 0
106 Memberi seleksi kondisi apabila nilai red pada variabel redValue
    bernilai negatif maka nilai akan menjadi 0
107 Delay 500 ms
108
109
110 Menyimpan nilai green pulse width dari fungsi getGreeenPW() pada
    variabel greenPW
111 Memberi nilai pada variabel greenValue (green) dengan fungsi map yang
    berisi parameter yaitu: pulse width, green minimum, green maximum,
    255 dan 0
112 Memberi seleksi kondisi apabila nilai green pada variabel greenValue
    bernilai negatif maka nilai akan menjadi 0
113 Delay 500 ms
114
115
116 Menyimpan nilai blue pulse width dari fungsi getBluePW() pada
    variabel bluePW
117 Memberi nilai pada variabel blueValue (blue) dengan fungsi map yang
    berisi parameter yaitu: pulse width, blue minimum, blue maximum, 255
    dan 0
118 Memberi seleksi kondisi apabila nilai blue pada variabel blueValue
    bernilai negatif maka nilai akan menjadi 0
119 Delay 500 ms
120
121 Menjumlahkan nilai redValue dari pembacaan tiap perulangan dan
    disimpan dalam variabel redValue juga
122 Menjumlahkan nilai greenValue dari pembacaan tiap perulangan dan
    disimpan dalam variabel greenValue juga
123 Menjumlahkan nilai blueValue dari pembacaan tiap perulangan dan
    disimpan dalam variabel blueValue juga
124 Menutup fungsi for (perulangan)
125 Membagi nilai redValue dengan 5 untuk mencari rata-rata dari
    pembacaan sensor (5 kali)
126 Membagi nilai greenValue dengan 5 untuk mencari rata-rata dari
    pembacaan sensor (5 kali)
127 Membagi nilai blueValue dengan 5 untuk mencari rata-rata dari
    pembacaan sensor (5 kali)
128 Menutup fungsi readRGB
```

Blok program 7

```
130  int cekWeight(){
131    int berat = 0;
132    scale.set_scale(calibration_factor);
133    berat = scale.get_units();
134    return berat;
135  }
136
137  int readWeight(){
138    average = 0;
139    scale.set_scale(calibration_factor);
140    for(int i = 0; i < 5; i++){
141         average += scale.get_units();
142    }
143    average /= 5;
144    return average;
145  }
```

```
130  Mendeklarasikan fungsi untuk mengecek berat
131  Mendeklarasikan variabel untuk menyimpan berat
132  Memanggil fungsi dari object pada class HX711 untuk set nilai
     kalibrasi untuk perhitungan berat pada load cell
133  Memberi nilai pada variabel berat dengan memanggil fungsi untuk
     mendapatkan nilai berat dari object pada class HX711
134  Mengembalikan fungsi dengan nilai berat
135  Menutup fungsi cekWeight
136
137  Mendeklarasikan fungsi untuk mendapatkan data berat untuk diproses
138  Memberi nilai pada variabel average = 0
139  Memanggil fungsi dari object pada class HX711 untuk set nilai
     kalibrasi untuk perhitungan berat pada load cell
140  Membuat perulangan sebanyak 5 kali untuk mendapatkan nilai rata-rata
     dari pembacaan berat (5 kali)
141  Menjumlahkan variabel average dengan fungsi scale.get_unit() untuk
     menyimpan nilai dan disimpan lagi di variabel average
142  Menutup for (perulangan)
143  Membagi nilai average dengan 5 untuk mendapatkan nilai rata-rata dari
     pembacaan sensor berat (5 kali)
144  Mengembalikan fungsi dengan nilai average
145  Menutup fungsi readWeight
```

Blok program 8

```
147  void setup_LCD(){
148    lcd.begin();
149    lcd.backlight();
150  }
151
152  void setup_loadCell(){
153    scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
154    scale.set_scale();
155    scale.tare(); //Reset the scale to 0
156    long zero_factor = scale.read_average();
157    Serial.println("Zero factor: ");
158    Serial.println(zero_factor);
159  }
160
161  void setup_colorSensor(){
162    // Set S0 - S3 as outputs
163    pinMode(S0_CS, OUTPUT);
164    pinMode(S1_CS, OUTPUT);
165    pinMode(S2_CS, OUTPUT);
```

```
166   pinMode(S3_CS, OUTPUT);
167   // Set Pulse Width scaling to 20%
168   digitalWrite(S0_CS,HIGH);
169   digitalWrite(S1_CS,LOW);
170   // Set Sensor output as input
171   pinMode(OUT_CS, INPUT);
172 }
```

```
147 Mendeklarasikan fungsi untuk setup LCD
148 Memanggil fungsi untuk memulai LCD dengan lcd.begin()
149 Memanggil fungsi untuk menyalakan lampu backlight pada LCD
150 Menutup fungsi setup_lcd
151
152 Mendeklarasikan fungsi untuk setup load cell
153 Memanggil fungsi untuk memulai load cell dengan fungsi begin() yang
    berisi parameter dari pin yang sudah di definiskan pada awal program
    (DOUT dan SCK)
154 Memanggil fungsi untuk set skala pada load cell
155 Memanggi fungsi untuk mereset skala ke 0 pada load cell
156 Mendeklarasikan variabel zero_factor untuk membantu dalam pembacaan
    berat
157 Menulis pada serial monitor
158 Menulis pada serial monitor dengan variabel zero_factor
159 Menutup fungsi setup_loadcell
160
161 Mendeklarasikan fungsi untuk setup color sensor
162
163 Memanggil fungsi untuk mode pin pada variabel S0_CS sebagai output
164 Memanggil fungsi untuk mode pin pada variabel S1_CS sebagai output
165 Memanggil fungsi untuk mode pin pada variabel S2_CS sebagai output
166 Memanggil fungsi untuk mode pin pada variabel S3_CS sebagai output
167
168 Memanggil fungsi untuk memberi nilai pada variabel S0_CS, HIGH
169 Memanggil fungsi untuk memberi nilai pada variabel S1_CS, LOW
170
171 Memanggil fungsi untuk mode pin pada variabel OUT_CS sebagai input
172 Menutup fungsi setup_colorSensor
```

Blok program 9

```
174 int getGreenPW() {
175   // Set sensor to read Green only
176   digitalWrite(S2_CS,HIGH);
177   digitalWrite(S3_CS,HIGH);
178   int PW;
179   PW = pulseIn(OUT_CS, LOW);
180   return PW;
181 }
182
183 int getBluePW() {
184   // Set sensor to read Blue only
185   digitalWrite(S2_CS,LOW);
186   digitalWrite(S3_CS,HIGH);
187   int PW;
188   PW = pulseIn(OUT_CS, LOW);
189   return PW;
190 }
191
192 int getRedPW() {
193   // Set sensor to read Red only
194   digitalWrite(S2_CS,LOW);
195   digitalWrite(S3_CS,LOW);
196   int PW;
197   PW = pulseIn(OUT_CS, LOW);
```

```
198   return PW;
199 }
```

```
174 Mendeklarasikan fungsi untuk mendapatkan nilai pulse width green
175
176 Memanggil fungsi untuk menulis pada pin S2_CS, HIGH
177 Memanggil fungsi untuk menulis pada pin S3_CS, HIGH
178 Mendeklarasikan variabel untuk menyimpan nilai pulse width
179 Memberi nilai pada variabel PW dengan mengambil pulse width dengan
    fungsi pulseIn dengan parameter pin OUT_CS, LOW
180 Mengembalikan fungsi dengan nilai PW
181 Menutup fungsi getGreenPW
182
183 Mendeklarasikan fungsi untuk mendapatkan nilai pulse width blue
184
185 Memanggil fungsi untuk menulis pada pin S2_CS, LOW
186 Memanggil fungsi untuk menulis pada pin S3_CS, HIGH
187 Mendeklarasikan variabel untuk menyimpan nilai pulse width
188 Memberi nilai pada variabel PW dengan mengambil pulse width dengan
    fungsi pulseIn dengan parameter pin OUT_CS, LOW
189 Mengembalikan fungsi dengan nilai PW
190 Menutup fungsi getBluepW
191
192 Mendeklarasikan fungsi untuk mendapatkan nilai pulse width red
193
194 Memanggil fungsi untuk menulis pada pin S2_CS, LOW
195 Memanggil fungsi untuk menulis pada pin S3_CS, LOW
196 Mendeklarasikan variabel untuk menyimpan nilai pulse width
197 Memberi nilai pada variabel PW dengan mengambil pulse width dengan
    fungsi pulseIn dengan parameter pin OUT_CS, LOW
198 Mengembalikan fungsi dengan nilai PW
199 Menutup fungsi getRedPW
```

Blok program 10

```
201 void FSM(){
202   t_setelah = millis() - t_awal;
203   switch(state){
204        case 0:
205        if(i == 0){
206          lcd.print("Put object on");
207          lcd.setCursor(0,1);
208          lcd.print("top of Load Cell");
209          i++;
210        }
211        Serial.println("Put object on top of load cell");
212        if(cekWeight() > 2 && t_setelah > treshold * 4){
213          lcd.clear();
214          lcd.setCursor(0,0);
215          lcd.print("Dont move object");
216          lcd.setCursor(0,1);
217          lcd.print("Processing...");
218          state = 1;
219          t_awal = millis();
220        }
221        break;
222
223        case 1:
224        if(t_setelah > treshold && cekWeight() > 2){
225          lcd.clear();
226          lcd.setCursor(0,0);
227          lcd.print("Weight:");
228          Serial.println("Begin reading weight...");
229          Serial.println("Reading weight...");
```

```
230           int w = readWeight();
231           Serial.print("Weight: ");
232           Serial.print(String(w) + "gram");
233           lcd.print(String(w) + " Gram");
234           delay(2000);
235           state = 2;
236           t_awal = millis();
237         }
238         break;
239 
240         case 2:
241         if(t_setelah > treshold && cekWeight() > 2){
242           lcd.clear();
243           lcd.setCursor(0,0);
244           lcd.print("Color:");
245           Serial.println("Begin reading color...");
246           Serial.println("Reading color...\n");
247           Serial.print("Color:\n");
248           readRGB();
249           lcd.setCursor(0,1);
250           lcd.print("R:");
251           lcd.print(String(redValue));
252           delay(1000);
253           lcd.print(",G:");
254           lcd.print(String(greenValue));
255           delay(1000);
256           lcd.print(",B:");
257           lcd.print(String(blueValue));
258           delay(3000);
259           state = 3;
260           t_awal = millis();
261         }
262         break;
263 
264         case 3:
265         if(t_setelah > treshold){
266           lcd.clear();
267           lcd.setCursor(0,0);
268           lcd.print("Begin Classifica-");
269           lcd.setCursor(0,1);
270           lcd.print("tion...");
271           Serial.println("Begin classification...");
272           send_data(redValue, greenValue, blueValue, average);
273           output = receive_data();
274           if(output == 'S'){
275           lcd.clear();
276           lcd.setCursor(0,0);
277           lcd.print("Jeruk");
278           lcd.setCursor(0,1);
279           lcd.print("Santang Madu");
280         } else {
281           lcd.clear();
282           lcd.setCursor(0,0);
283           lcd.print("Jeruk");
284           lcd.setCursor(0,1);
285           lcd.print("Kunci");
286         }
287          state = 4;
288          t_awal = millis();
289         }
290        break;
291 
292         case 4:
293         if((t_setelah > treshold) && (output == 'K' || output ==
294 'S')){
295           Serial.print("Servo Bottom move...");
296           process_bottom_servo(output);
```

```
297           state = 5;
298           t_awal = millis();
299         }
300       break;
301
302        case 5:
303        if(t_setelah > treshold){
304           Serial.print("Servo Top move...");
305           process_top_servo();
306           state = 6;
307           t_awal = millis();
308        }
309       break;
310
311        case 6:
312        if(t_setelah > treshold * 2){
313        Serial.print("End of process...");
314        if(cekWeight() < 3){
315        reset_all();
316        state = 0;
317        t_awal = millis();
318        }
319        }
320        break;
321        }
322   }
```

```
201 | Mendeklarasikan fungsi untuk keseluruhan sistem dengan Finite
    | State Machine
202 | Memberi nilai pada t_setelah untuk mendapatkan selisih waktu dari
    | waktu sekarang dikurangi waktu terakhir untuk proses pada tiap
    | blok case di FSM
203 | Membuat switch case dengan parameter dari variabel state
204 | Membuat case apabila state bernilai 0 (mulai case)
205 | Membuat seleksi kondisi untuk membantu penampilkan keterangan pada
    | layar lcd
206 | Menampilkan tulisan pada layar lcd
207 | Set cursor pada posisi 0,0 pada layar lcd
208 | Menampilkan tulisan pada layar lcd
209 | Increment nilai i
210 | Menutup seleksi kondisi (if)
211 | Menulis tulisan pada serial monitor
212 | Memberi seleksi kondisi apabila berat > 2 dan waktu lebih dari 8
    | detik (bertujuan untuk memberi delay agar ada waktu untuk menaruh
    | objek pada load cell
213 | Membersihkan layar lcd
214 | Set cursor pada posisi 0,0 pada layar lcd
215 | Menampilkan tulisan pada layar lcd
216 | Set cursor pada posisi 0,1 pada layar lcd
217 | Menampilkan tulisan pada layar lcd
218 | Memberi nilai pada state = 1, untuk mengubah state agar masuk ke
    | proses berikutnya pada FSM
219 | Memberi nilai pada t_awal dengan waktu saat ini (millis) agar
    | mendapatkan selisih waktu pada baris awal di fungsi FSM()
220 | Menutup seleksi kondisi (if)
221 | Break untuk keluar dari case 0 (akhir case)
222 |
223 | Membuat case apabila state bernilai 1 (mulai case)
224 | Memberi seleksi kondisi apabila berat masih > 2 dan waktu lebih
    | dari 2 detik (threshold)
225 | Membersihkan layar lcd
226 | Set cursor pada posisi 0,0 pada layar lcd
227 | Menampilkan tulisan pada layar lcd
228 | Menampilkan tulisan pada serial monitor
229 | Menampilkan tulisan pada serial monitor
230 | Mendeklarasikan variabel w untuk menyimpan nilai berat dari fungsi
    | readWeight()
231 | Menampilkan tulisan pada serial monitor
232 | Menampilkan tulisan pada serial monitor
233 | Menampilkan tulisan pada lcd (weight)
234 | Delay 2000 ms
235 | Memberi nilai pada state = 1, untuk mengubah state agar masuk ke
    | proses berikutnya pada FSM
236 | Memberi nilai pada t_awal dengan waktu saat ini (millis) agar
    | mendapatkan selisih waktu pada baris awal di fungsi FSM()
237 | Menutup seleksi kondisi (if)
238 | Break untuk keluar dari case 1 (akhir case)
239 |
240 | Membuat case apabila state bernilai 2 (mulai case)
241 | Memberi seleksi kondisi apabila berat masih > 2 dan waktu lebih
    | dari 2 detik (threshold)
242 | Membersihkan layar lcd
243 | Set cursor pada posisi 0,0 pada layar lcd
244 | Menampilkan tulisan pada layar lcd
245 | Menampilkan tulisan pada serial monitor
246 | Menampilkan tulisan pada serial monitor
247 | Menampilkan tulisan pada serial monitor
248 | Memanggil fungsi readRGB untuk mendapatkan nilai R,G dan B
249 | Set cursor pada posisi 0,1 pada layar lcd
250 | Menampilkan tulisan pada layar lcd
251 | Menampilkan tulisan pada layar lcd (red)
252 | Delay 1000 ms
253 | Menampilkan tulisan pada layar lcd
254 | Menampilkan tulisan pada layar lcd (green)
```

```
255 | Delay 1000 ms
256 | Menampilkan tulisan pada layar lcd
257 | Menampilkan tulisan pada layar lcd (blue)
258 | Delay 3000 ms
259 | Memberi nilai pada state = 3, untuk mengubah state agar masuk ke
    | proses berikutnya pada FSM
260 | Memberi nilai pada t_awal dengan waktu saat ini (millis) agar
    | mendapatkan selisih waktu pada baris awal di fungsi FSM()
261 | Menutup seleksi kondisi (if)
262 | Break untuk keluar dari case 2 (akhir case)
263 |
264 | Membuat case apabila state bernilai 3 (mulai case)
265 | Memberi seleksi kondisi apabila waktu lebih dari 2 detik
266 | (threshold)
267 | Membersihkan layar lcd
268 | Set cursor pada posisi 0,0 pada layar lcd
269 | Menampilkan tulisan pada layar lcd
270 | Set cursor pada posisi 0,1 pada layar lcd
271 | Menampilkan tulisan pada layar lcd
272 | Menampilkan tulisan pada serial monitor
    | Memanggil fungsi send_data dengan parameter dari variabel yang
273 | menyimpan nilai R,G,B dan weight (berat)
    | Memberi nilai pada variabel output dari hasil memanggil fungsi
274 | receive_data yang akan mengembalikan nilai berupa char jenis jeruk
275 | Membuat seleksi kondisi apabila output bernilai 'S' (Santang Madu)
276 | Membersihkan layar lcd
277 | Set cursor pada posisi 0,0 pada layar lcd
278 | Menampilkan tulisan pada layar lcd
279 | Set cursor pada posisi 0,1 pada layar lcd
280 | Menampilkan tulisan pada layar lcd
281 | Apabila bukan 'S' (Kunci)
282 | Membersihkan layar lcd
283 | Set cursor pada posisi 0,0 pada layar lcd
284 | Menampilkan tulisan pada layar lcd
285 | Set cursor pada posisi 0,1 pada layar lcd
286 | Menampilkan tulisan pada layar lcd
287 | Menutup seleksi kondisi (if)
    | Memberi nilai pada state = 4, untuk mengubah state agar masuk ke
288 | proses berikutnya pada FSM
    | Memberi nilai pada t_awal dengan waktu saat ini (millis) agar
289 | mendapatkan selisih waktu pada baris awal di fungsi FSM()
290 | Menutup seleksi kondisi (if)
291 | Break untuk keluar dari case 3 (akhir case)
292 |
293 | Membuat case apabila state bernilai 4 (mulai case)
    | Memberi seleksi kondisi waktu lebih dari 2 detik (threshold) dan
294 | variabel output sama dengan 'K' atau 'S'
295 | Menampilkan tulisan pada serial monitor
    | Memanggil fungsi process_bottom_servo dengan parameter output
296 | untuk proses pergerakan servo
    | Memberi nilai pada state = 5, untuk mengubah state agar masuk ke
297 | proses berikutnya pada FSM
298 | Memberi nilai pada t_awal dengan waktu saat ini (millis) agar
299 | mendapatkan selisih waktu pada baris awal di fungsi FSM()
300 | Menutup seleksi kondisi (if)
301 | Break untuk keluar dari case 4 (akhir case)
302 |
303 | Membuat case apabila state bernilai 5 (mulai case)
304 | Memberi seleksi kondisi waktu lebih dari 2 detik
305 | Menampilkan tulisan pada serial monitor
    | Memanggil fungsi process_top_servo dengan parameter output untuk
306 | proses pergerakan servo
    | Memberi nilai pada state = 6, untuk mengubah state agar masuk ke
307 | proses berikutnya pada FSM
    | Memberi nilai pada t_awal dengan waktu saat ini (millis) agar
    | mendapatkan selisih waktu pada baris awal di fungsi FSM()
308 | Menutup seleksi kondisi (if)
```

```
309  Break untuk keluar dari case 4 (akhir case)
310
311  Membuat case apabila state bernilai 6 (mulai case)
312  Memberi seleksi kondisi waktu lebih dari 4 detik (threshold)
313  Menampilkan tulisan pada serial monitor
314  Memberi seleksi kondisi apabila cekWeight > 3 untuk memastikan
315  bahwa objek sudah tidak ada diatas load cell
     Memanggil fungsi reset_all() untuk mereset semua state pada
316  hardware dan nilai pada variabel
     Memberi nilai pada state = 0, untuk mengubah state agar masuk ke
317  proses berikutnya pada FSM (kembali ke proses awal) (end of
     process)
     Memberi nilai pada t_awal dengan waktu saat ini (millis) agar
318  mendapatkan selisih waktu pada baris awal di fungsi FSM()
     Menutup seleksi kondisi (if)
319  Menutup seleksi kondisi (if)
320  Break untuk keluar dari case 6 (akhir case)
321  Menutup switch case
322  Menutup fungsi FSM
```

Blok program 11

```
324  void setup(){
325    Serial.begin(9600);
326    Serial.setTimeout(1);
327    while(!Serial.available()) {
328         Serial.println("waiting to start...");
329         delay(1000);
330    }
331    setup_LCD();
332    lcd.setCursor(0,0);
333    Serial.println("LCD OK...");
334    delay(1000);
335    lcd.print("Setting Up...");
336    Serial.println("Setting up...");
337    delay(1000);
338    setup_loadCell();
339    Serial.println("Load Cell OK...");
340    delay(1000);
341    setup_colorSensor();
342    Serial.println("Color Sensor OK...");
343    delay(1000);
344    servo_init();
345    Serial.println("Servo OK...");
346    delay(1000);
347    t_awal = millis();
348    lcd.clear();
349    lcd.print("All set, Ready!");
350    delay(2000);
351    Serial.println("All set, Ready!");
352    delay(100);
353    lcd.clear();
354  }
355
356  void loop(){
357    FSM();
358  }
```

```
Mendeklarasikan fungsi setup arduino
Memulai serial monitor dengan inisialiasi baudrate (9600)
Memberi timeout 1 detik pada arduino untuk membantu proses antara arduino dan komputer
Membuat perulangan apabila serial monitor belum tersedia
Menampilkan tulisan pada serial monitor menandakan bahwa alat belum siap/dimulai
Delay 1000 ms
Menutup perulangan
Memanggil fungsi setup_LCD untuk inisialiasi lcd
Set cursor 0,0 pada layar lcd
Menampilkan tulisan pada serial monitor
Delay 1000 ms
Menampilkan tulisan pada layar lcd
Menampilkan tulisan pada serial monitor
Delay 1000 ms
Memanggil fungsi setup_loadcell untuk inisialiasi load cell (HX711)
Menampilkan tulisan pada serial monitor
Delay 1000 ms
Memanggil fungsi setup_colorSensor untuk inisialiasi color sensor
Menampilkan tulisan pada serial monitor
Delay 1000 ms
Memanggil fungsi servo_init untuk inisialiasi semua servo
Menampilkan tulisan pada serial monitor
Delay 1000 ms
Memberi nilai pada t_awal dengan waktu saat ini (millis) agar mendapatkan selisih waktu pada baris awal di fungsi FSM()
Membersihkan layar lcd
Menampilkan tulisan pada layar lcd
Delay 2000 ms
Menampilkan tulisan pada serial monitor
Delay 100 ms
Membersihkan layar lcd
Menutup fungsi setup()

Mendeklarasikan fungsi loop() arduino
Memanggil fungsi FSM() untuk menjalankan inti proses pada sistem
Menutup fungsi loop()
```

4.3.2 Python

No | **python**

```
1  from re import T
2  import time
3  from numpy import byte
4  import serial
5  import argparse
6  import pandas as pd
7
8  from numpy.random.mtrand import uniform
9  from sklearn import *
10
11 SERIAL_PORT = "/dev/ttyACM0"
12 BAUDRATE = 9600
13
14 # Init Serial Communication
15 ser = serial.Serial(SERIAL_PORT, BAUDRATE, timeout=1)
16
17 # Argument parser for n neighbours from terminal argument
18 ap = argparse.ArgumentParser()
19 ap.add_argument("-n", "--neighbours", type=int, default=3, help="set
20 the number of neighbours for voting purpose")
21 arguments = vars(ap.parse_args())
```

```
22
23  # Variable for data
24  dep_data = 0
25  indep_data = 0
26  incoming_bytes = b''
27
28  def read_data():
29        global indep_data, dep_data
30        data_path =
31  '/home/danidzz/Development/pengenalan_pola_project/python/asset/jeruk_
32  baru.csv'
33        df = pd.read_csv(data_path)
34        # memisahkan data independent (fitur) dan dependent (class)
35        indep_data = df.iloc[:, 1:5]
36        dep_data = df.iloc[:, 5]
37
38  def process_incoming_data(data):
39        if(data == b's\r\n'):
40        arduino_data = ""
41        arr = []
42        data = ser.readline().decode('ascii')
43        while(not data):
44              data = ser.readline().decode('ascii')
45              pass
46        data = data[0:len(data)-2]
47        for x in data:
48              arduino_data += x
49        arr = arduino_data.split(",")
50        for i in range(0, len(arr)):
51              arr[i] = int(arr[i])
52        print(arr)
53        return knn_classifier(arr, arguments["neighbours"])
54        else:
55        print(data)
56        while(not data):
57              data = ser.readline().decode('ascii')
58              pass
59        return None
60
61  def knn_classifier(data, neighbours):
62        global indep_data, dep_data, output
63        knn = neighbors.KNeighborsClassifier(n_neighbors=neighbours,
64  weights="uniform")
65        knn.fit(indep_data, dep_data)
66        predict_knn = knn.predict([data])
67        if predict_knn == "Kunci":
68          time.sleep(1)
69          ser.write(b'f')
70          time.sleep(1)
71          ser.write("k".encode())
72          print("Jeruk Kunci")
73        else:
74          time.sleep(1)
75          ser.write(b'f')
76          time.sleep(1)
77          ser.write("s".encode())
78          print("Jeruk S Madu")
79        time.sleep(1)
80
81  if __name__ == "__main__":
82        read_data()
83        time.sleep(2)
84        ser.write([1])
85        while True:
86        incoming_bytes = process_incoming_data(ser.readline())
87        print(incoming_bytes)
```

Blok program 1

```
from re import T
import time
from numpy import byte
import serial
import argparse
import pandas as pd

from numpy.random.mtrand import uniform
from sklearn import *

SERIAL_PORT = "/dev/ttyACM0"
BAUDRATE = 9600

# Init Serial Communication
ser = serial.Serial(SERIAL_PORT, BAUDRATE, timeout=1)

# Argument parser for n neighbours from terminal argument
ap = argparse.ArgumentParser()
ap.add_argument("-n", "--neighbours", type=int, default=3, help="set
the number of neighbours for voting purpose")
arguments = vars(ap.parse_args())

# Variable for data
dep_data = 0
indep_data = 0
incoming_bytes = b''
```

```
Import library
Import library untuk delay
Import library numpy untuk data byte
Import library untuk komunikasi serial
Import library untuk argument parser di terminal untuk nilai K pada
KNN
Import library pandas untuk memproses dataset

Import library untuk uniform pada parameter fungsi KNNClassifier
Import library sklearn untuk mengolah data menggunakan KNN

Mendeklarasikan serial port yang sama dengan arduino
Mendeklarasikan baudrate yang sama dengan arduino


Mendeklarasikan variabel yang akan digunakan untuk proses komunikasi
data di serial port

Mendeklarasikan variabel untuk argument parser
Menambahkan argument berupa K pada KNN yang nantinya akan diinputkan
pada terminal argmuent
Mendeklarasikan variabel untuk mengaktifkan argument parse


Mendeklarasikan nilai dep_data untuk kelas pada dataset
Mendeklarasikan nilai dep_data untuk fitur pada dataset
Mendeklarasikan variabel untuk menyimpan data berupa bytes yang masuk
ke serial port dari arduino
```

Blok program 2

```
28  def read_data():
29        global indep_data, dep_data
30        data_path =
31  '/home/danidzz/Development/pengenalan_pola_project/python/asset/jeruk
32  _baru.csv'
33        df = pd.read_csv(data_path)
34        # memisahkan data independent (fitur) dan dependent (class)
35        indep_data = df.iloc[:, 1:5]
36        dep_data = df.iloc[:, 5]
```

```
28  Mendeklarasikan fungsi untuk membaca dataset
29  Assign variabel indep_data dan dep_data di dalam fungsi sebagai
    variabel global
30  Mendeklarasikan data_path yang berisi path dari dataset pada komputer
31
32
33  Mendeklarasikan variabel df untuk membaca csv dari variabel data_path
    yang berisi path dari dataset
34
35  Memisahkan fitur dari dataset(csv) pada kolom 1-4
36  Memisahkan kelas dari dataset(csv) pada kolom 5
```

Blok program 3

```
38  def process_incoming_data(data):
39        if(data == b's\r\n'):
40        arduino_data = ""
41        arr = []
42        data = ser.readline().decode('ascii')
43        while(not data):
44              data = ser.readline().decode('ascii')
45              pass
46        data = data[0:len(data)-2]
47        for x in data:
48              arduino_data += x
49        arr = arduino_data.split(",")
50        for i in range(0, len(arr)):
51              arr[i] = int(arr[i])
52        print(arr)
53        return knn_classifier(arr, arguments["neighbours"])
54        else:
55        print(data)
56        while(not data):
57              data = ser.readline().decode('ascii')
58              pass
59        return None
```

```
38  Mendeklarasikan fungsi untuk memproses data yang masuk ke serial port
39  Membuat seleksi kondisi apabila data yang masuk = b's\r\n'
40  Mendeklarasikan variabel arduino_data untuk menyimpan data dari
    arduino
41  Mendeklarasikan array untuk mengubah format data dari arduino yang
    masuk ke serial port
42  Membaca data yang masuk ke serial port kemudian di decode(ascii) dan
    dimasukkan ke variabel data
43  Membuat perulangan while apabila data yang diinginkan belum masuk ke
    serial port
44  Membaca data yang masuk ke serial port kemudian di decode(ascii) dan
    dimasukkan ke variabel data
45  Melanjutkan perulangan
```

	Mentup perulangan (while)
46	Mengurangi panjang data dari arduino agar hanya mendapatkan nilainya saja tanpa character '\r\n' (clean data)
47	Membuat perulangan for untuk memasukkan setiap character dari data arduino yang sudah bersih ke variabel arduino_data untuk diubah menjadi string
48	Menambahkan setiap character pada data arduino ke variabel arduino_data Menutup perulangan (for)
49	Memberi nilai pada variabel arr dengan mengambil data angka saja, dengan menggunakan fungsi split untuk membuang character ',' dari data string arduino_data
50	Membuat perulangan untuk mengubah array pada variabel arr dari bentuk string ke int agar bisa diproses KNN
51	Memberi nilai setiap indeks pada array arr dengan tipe int dari nilai tiap indeks pada array arr
52	Menampilkan data arr
53	Mengembalikan fungsi dengan nilai dari fungsi knn_classifier dengan parameter nilai arr dan argument pada terminal Endof if state
54	Apabila data yang masuk bukan data b's\r\n'
55	Menampilkan data
56	Membuat perulangan while agar tidak menerima data kosong (b'')
57	Membaca data yang masuk ke serial port kemudian di decode(ascii) dan dimasukkan ke variabel data
58	Melanjutkan perulangan Menutup perulangaan (while)
59	Mengembalikan nilai None (NULL)

Blok program 4

```
61  def knn_classifier(data, neighbours):
62        global indep_data, dep_data, output
63        knn = neighbors.KNeighborsClassifier(n_neighbors=neighbours,
64  weights="uniform")
65        knn.fit(indep_data, dep_data)
66        predict_knn = knn.predict([data])
67        if predict_knn == "Kunci":
68          time.sleep(1)
69          ser.write(b'f')
70          time.sleep(1)
71          ser.write("k".encode())
72          print("Jeruk Kunci")
73        else:
74          time.sleep(1)
75          ser.write(b'f')
76          time.sleep(1)
77          ser.write("s".encode())
78          print("Jeruk S Madu")
79        time.sleep(1)
```

61	Mendeklarasikan fungsi untuk memproses data dengan KNN dengan parameter data dan nilai K (neighbours)
62	Mendeklarasikan variabel indep_data dan dep_data pada fungsi sebagai variabel global
63	Mendeklarasikan variabel knn sebagai instance dari class neighbors.KNeighborsClassifier
64	
65	Melatih knn dengan data fitur dan kelas pada variabel indep_data dan dep_data dengan fungsi knn.fit()
66	Menyimpan hasil prediksi data dari fungsi prediksi pada KNN di variabel predict_knn
67	Membuat seleksi kondisi apabila hasil prediksi adalah "Kunci"
68	Delay 1 s

69	Menulis data ‘f’ pada serial port untuk menandakan bahwa proses klasifikasi sudah selesai dilakukan sehingga arduino dapat memproses proses selanjutnya
70	Delay 1 s
71	Menulis data “k” pada serial port sebagai output dari proses klasifikasi untuk kemudian di proses di arduino
72	Menampilkan tulisan (End of if)
73	Apabila hasil prediksi bukan “Kunci” (S.Madu)
74	Delay 1 s
75	Menulis data ‘f’ pada serial port untuk menandakan bahwa proses klasifikasi sudah selesai dilakukan sehingga arduino dapat memproses proses selanjutnya
76	Delay 1 s
77	Menulis data “s” pada serial port sebagai output dari proses klasifikasi untuk kemudian di proses di arduino
78	Menampilkan tulisan
79	Delay 1 s

Blok program 5

```
81  if __name__ == "__main__":
82        read_data()
83        time.sleep(2)
84        ser.write([1])
85        while True:
86          incoming_bytes = process_incoming_data(ser.readline())
87          print(incoming_bytes)
```

81	Membuat blok fungsi main (blok hanya akan dijalankan pada program) untuk menjalankan fungsi-fungsi pada program
82	Memanggil fungsi read_data untuk membaca dataset
83	Delay 2 detik
84	Menulis data 1 (int) untuk memulai proses sistem (sebagai penanda untuk arduino bahwa proses (sistem) bisa dimulai
85	Membuat perulangan untuk jalannya program utama secara terus menerus
86	Memberi nilai pada variabel incoming_bytes dengan fungsi process_incoming_data dengan parameter ser.readline() (untuk membaca data masuk pada serial port)
87	Menampilkan tulisan dari variabel incoming_data

BAB V
HASIL PENGUJIAN

5.1 Confusion Matrix

Pencarian Akurasi untuk Klasifikasi antara Jeruk Kunci dan Jeruk Santang Madu dilakukan dengan Confusion Matrix pada kedua kelas dicari nilai aktual dari Jeruk Kunci dan Jeruk Santang Madu. Kemudian, nilai prediksi dari Jeruk Kunci dan Jeruk Santang berdasarkan percobaan yang dilakukan dengan 50 data Jeruk di Tabel Data pada alat.

Tabel Confusion Matrix

		Prediction	
		Kunci	Santang Madu
Actual	Kunci	24 (TN)	2 (FP)
	Santang Madu	0 (FN)	24 (TP)

Penjelasan mengenai confusion matrix diatas adalah sebagai berikut.

- TN (True Negative) adalah Jeruk Kunci yang hasil prediksinya adalah Jeruk Kunci.
- FN (False Negative) adalah Jeruk Santang Madu yang hasil prediksinya adalah Jeruk Kunci.
- FP (False Positive) adalah Jeruk Kunci yang hasil prediksinya adalah Jeruk Santang Madu.
- TP (True Positive) adalah Jeruk Santang Madu yang hasil prediksinya adalah Jeruk Santang Madu.

5.2 Hasil Akurasi

Nilai-nilai pada Confusion Matrix tersebut dimasukkan ke dalam rumus Akurasi sebagai berikut:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Accuracy = \frac{24 + 24}{24 + 24 + 0 + 2}$$

$$Accuracy = \frac{48}{50}$$

$$Accuracy = 0{,}96$$

Maka, hasil akurasi dari percobaan Klasifikasi antara Jeruk Kunci dan Jeruk Santang Madu menggunakan Metode K-Nearest Neighbors dengan K = 5, didapatkan hasil akurasi sebesar 0.96 atau 96 % (apabila dalam persen).