Conclusion

The project successfully achieved its objective of reducing waste and improving efficiency by implementing a conveyor system with two IR modules.

The first module detects the presence of a Pepsi can, causing the conveyor to stop for 2 seconds. During this time, a signal is sent to MATLAB on a laptop to capture an image of the product.

The image processing algorithms, SURF and ORB, are then applied to analyze the image.

Afterward, the conveyor continues moving until the second IR sensor is triggered, prompting it to stop.

MATLAB then sends a signal of either 1 (valid product) or 0 (invalid product) to the robot arm, which initiates the movement of the product to its correct position.

Additionally, the product information is displayed on an I2C LCD.

Overall, this project successfully contributes to waste reduction and efficiency improvement by providing valuable information about the cans daily.

Appendix A

```
#include <AccelStepper.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define STEPS_PER_REVOLUTION 200
#define CONVEYOR_SPEED 500 // Adjust this value to control the speed of the
conveyor

AccelStepper stepper(AccelStepper::DRIVER, 2, 3); // Stepper motor control pins

int irSensor1Pin = 4; // Pin connected to the first IR sensor
int irSensor2Pin = 5; // Pin connected to the second IR sensor

int captureSignalPin = 6; // Pin to send capture signal to MATLAB
int productStatusPin = 7; // Pin to receive product status from MATLAB

int validProductCount = 0;
int invalidProductCount = 0;
int totalCount = 0;

LiquidCrystal_I2C lcd(0x27, 16, 2); // I2C LCD display parameters

void setup() {
  stepper.setMaxSpeed(2000);
  stepper.setAcceleration(1000);

  pinMode(irSensor1Pin, INPUT);
  pinMode(irSensor2Pin, INPUT);

  pinMode(captureSignalPin, OUTPUT);
  pinMode(productStatusPin, INPUT);

  lcd.begin(16, 2);
  lcd.print("Valid: 0 Invalid: 0");
}

void loop() {
  if (digitalRead(irSensor1Pin) == HIGH) {
    stepper.stop();
    delay(2000); // Stop the conveyor for 2 seconds

    digitalWrite(captureSignalPin, HIGH); // Send capture signal to MATLAB
    delay(100);
    digitalWrite(captureSignalPin, LOW);

    int productStatus = digitalRead(productStatusPin); // Receive product status
from MATLAB

    if (productStatus == 1) {
      validProductCount++;
      totalCount++;
      lcd.setCursor(0, 0);
      lcd.print("Valid: ");
      lcd.print(validProductCount);
      lcd.setCursor(0, 1);
      lcd.print("Total: ");
      lcd.print(totalCount);
    } else {
      invalidProductCount++;
      totalCount++;
      lcd.setCursor(0, 0);
      lcd.print("Invalid: ");
      lcd.print(invalidProductCount);
      lcd.setCursor(0, 1);
      lcd.print("Total: ");
      lcd.print(totalCount);
    }

    stepper.setSpeed(CONVEYOR_SPEED);
    stepper.runSpeedToPosition();
  }

  if (digitalRead(irSensor2Pin) == HIGH) {
    stepper.stop();
    delay(15000); // Wait for 15 seconds

    stepper.setSpeed(CONVEYOR_SPEED);
    stepper.runSpeedToPosition();
  }
}
```

Appendix B

Reference code

```
clc
clear
close all;
%webcam = webcamlist;
cam = webcam;
cam.Resolution = '1280x720';
preview(cam);
disp('Press any key to capture a reference image...');
pause;
img = snapshot(cam);
imwrite(img, 'refrence20.jpg');
```

Main code

```
clc
clear
close all;
%  can refrence image -------
```

```
refrence = imread("reference_image.jpg");
refrence= rgb2gray(refrence);
%refrencegreyimage = imadjust(refrencegrey);
%figure,imshow(refrence),title('refrence');

% setup camera connection  --------
 webcamlist;    to show list of connected cams
camera = webcam;

% connect arduino 1   ---input pin
arduino1 = arduino('com3','uno');
arduino1pin = 'd2';
configurepin(arduino1,arduino1pin,'DigitalInput');

% connect arduino 2  ----output pin
arduino2 = arduino('com4','uno');
arduino2pin= 'd3';
configurepin(arduino2,arduino2pin,'DigitalPin');

while true

    % check if arduino sends asignal to take the snap
     arduino1value = readDigitalPin(arduino1,arduino1pin);
      if arduino1value ==1
        tic
       img = snapshot(camera);
       % img=imread("compared2.jpg");
        captured = rgb2gray(img);
        %capturedgreyimage = imadjust(distorted);
        %figure,imshow(captured);
        %title('captured');
    % close camera -- avoid take another image
        clear('camera');

    % ----------- Used algorithms -----------

    % feature extraction using surf and orb algorithms
    % Detect features
       referencepoints = detectSURFFeatures(refrence);
       capturedpoints  = detectSURFFeatures(captured);
```

```
   %----
     orbReferencePoints = detectORBFeatures(refrence);
     orbCapturedPoints = detectORBFeatures(captured);
   % feature description

    [refrencefeatures , validrefrencepoints] =
extractFeatures(refrence,referencepoints);
    [capturedfeatures , validcapturedpoints] =
extractFeatures(captured,capturedpoints);
   %------
    [orbRefFeatures, orbRefPoints] = extractFeatures(refrence,
orbReferencePoints);
    [orbCapturedFeatures, orbCapturedPoints] = extractFeatures(captured,
orbCapturedPoints);
   % features matching by using their descriptors

    indexPairs = matchFeatures(refrencefeatures,capturedfeatures);
    orbIndexPairs = matchFeatures(orbRefFeatures, orbCapturedFeatures);

   % retrive locations of corresponding points
    matchedrefrence = validrefrencepoints(indexPairs(:,1));
    matchedcaptured = validcapturedpoints(indexPairs(:,2));
    %------
    orbMatchedRefPoints = orbRefPoints(orbIndexPairs(:, 1), :);
    orbMatchedCapturedPoints = orbCapturedPoints(orbIndexPairs(:, 2), :);

   % estimate transformation
    [tform, inliercatured, inlierrefrence] =
estimateGeometricTransform(matchedcaptured,matchedrefrence,'similarity');
   %-----
   [orbTform, inlierCapturedPoints, inlierRefPoints] =
estimateGeometricTransform(orbMatchedCapturedPoints, orbMatchedRefPoints,
'similarity');
   % appling the transformation
    outputview = imref2d(size(refrence));
    outputimage = imwarp(captured,tform,'OutputView',outputview);
   %---
   orboutputview = imref2d(size(refrence));
   orbWarpedCapturedImage = imwarp(captured, orbTform,
'OutputView',orboutputview);
```

```
    % diplay refrene and matched captured imgs
 %    figure,imshowpair(refrence,outputimage,'montage');
%     figure,imshowpair(refrence,orbWarpedCapturedImage,'montage');
   % Resize the warped captured images to match the size of the reference image
surfWarpedCapturedImage = imresize(outputimage, size(refrence));
orbWarpedCapturedImage = imresize(orbWarpedCapturedImage, size(refrence));

% Crop both images to the same region of interest
roi = [200 200 1200 900];
refImage = imcrop(refrence, roi);
surfWarpedCapturedImage = imcrop(surfWarpedCapturedImage, roi);
orbWarpedCapturedImage = imcrop(orbWarpedCapturedImage, roi);

% Convert both images to black and white using a threshold value
surfBWRef = im2bw(refImage, graythresh(refImage));
surfBWCaptured = im2bw(surfWarpedCapturedImage,
graythresh(surfWarpedCapturedImage));
orbBWRef = im2bw(refImage, graythresh(refImage));
orbBWCaptured = im2bw(orbWarpedCapturedImage,
graythresh(orbWarpedCapturedImage));

% Calculate the correlation coefficients for both matches
surfCorrCoeff = corr2(surfBWRef, surfBWCaptured);
orbCorrCoeff = corr2(orbBWRef, orbBWCaptured);
   % Calculate the aspect ratio of the object in the captured image
%stats = regionprops(surfBWCaptured, 'BoundingBox');
%boundingBoxes = cat(1, stats.BoundingBox); % Combine all bounding boxes
into a single matrix
%width = boundingBoxes(3);
%height = boundingBoxes(4);
%aspectRatio = width / height;

   % Use a voting system to determine the validity of the object
numMatches = size(indexPairs, 1) + size(orbIndexPairs, 1);
surfMatchQuality = mean(referencepoints.Metric);
orbMatchQuality = mean(orbMatchedRefPoints.Metric);
confidence = (numMatches / 20) + (surfMatchQuality / 100) + (orbMatchQuality /
100);
if confidence >= 73
```

```
    %if aspectRatio > 1.2 || aspectRatio < 0.8
     %   d3 = 0; % indicates distortion

        d3 = 1; % indicates valid object

else
    d3 = 0; % indicates invalid object
end
```

Appendix C

```
#include <Servo.h>
#include <math.h>
#include <softwareserial.h>
#define Echo 7 //"Echo" labeled pin of HC-SR04 ultrasonic module attach to pin 7 on Arduino
UNO board
#define Trigger 8 //"Trig" labeled pin of HC-SR04 ultrasonic module attach to pin 8 on Arduino
UNO board


Long Time;//the time variable for ultrasnonic senor
Float Distance;//the distance variable for ultrasonic sensor

Int image;

Servo servo1; //servo of the 1st joint
Servo servo2; //servo of the 2nd joint
Servo servo3; //servo of the 3rd joint
Servo servo4; //servo of the 4th joint
Servo gripper; //servo to drive the gripping mechanism of gripper module

Float Pi = 3.141592653589793;
Float Yoffset;
Float D;
Float d;
Float R;
Float L1 = 64.50; //height of the first link from surface to 2nd joint position
Float L2 = 105.00; //length of the second link from 2nd joint to 3rd joint
Float L3 = 98.50; //length of the 3rd joint to 4th joint
Float L4 = 115.00; //length of the 4th joint to the tip of the arm (gripper)
Float X_End_Effector; //x axis coordinate of the gripper
Float Y_End_Effector; //y axis coordinate of the gripper
Float Z_End_Effector; //z axis coordinate of the gripper

Int pin_servo1 = 9; //joint 1 servo (yellow cable) attach to pin 9 on the arduino board
Int pin_servo2 = 5; //joint 2 servo (yellow cable) attach to pin 5 on the arduino board
Int pin_servo3 = 10; //joint 3 servo (yellow cable) attach to pin 10 on the arduino board
Int pin_servo4 = 6; //joint 4 servo (yellow cable) attach to pin 6 on the arduino board
Int pin_gripper = 3; //gripper servo (yellow cable) attach to pin 3 on the arduino board

Float alpha1;
Float alpha2;
Float Theta_2;
Float Theta_4;
Float Theta_1;
Float Theta_3;

Void Rumus_IK()
{
 If (X_End_Effector > 0 && Y_End_Effector >= L1)
 {
  D = sqrt(pow(X_End_Effector,2) + pow(Z_End_Effector,2));
  Theta_1 = (atan(Z_End_Effector/X_End_Effector))*(180.00/Pi); //theta 1
  D = D - L4;
  Yoffset = Y_End_Effector - L1;
  R = sqrt(pow(d,2) + pow(Yoffset,2));
  Alpha1 = (acos(d/R))*(180.00/Pi);
  Alpha2 = (acos((pow(L2,2) + pow(R,2) - pow(L3,2))/(2*L2*R)))*(180.00/Pi);
  Theta_2 = (alpha1 + alpha2); //theta 2
  Theta_3 = ((acos((pow(L2,2) + pow(L3,2) - pow(R,2))/(2*L2*L3)))*(180.00/Pi)); //theta 3
  Theta_4 = 180.00 - ((180.00 - (alpha2 + Theta_3)) - alpha1); //theta 4
 }
 Else if (X_End_Effector > 0 && Y_End_Effector <= L1)
 {
  D = sqrt(pow(X_End_Effector,2) + pow(Z_End_Effector,2));
  Theta_1 = (atan(Z_End_Effector/X_End_Effector))*(180.00/Pi); //theta 1
  D = D - L4;
  Yoffset = Y_End_Effector - L1;
  R = sqrt(pow(d,2) + pow(Yoffset,2));
  Alpha1 = (acos(d/R))*(180.00/Pi);
  Alpha2 = (acos((pow(L2,2) + pow(R,2) - pow(L3,2))/(2*L2*R)))*(180.00/Pi);
  Theta_2 = (alpha2 - alpha1); //theta 2
  Theta_3 = ((acos((pow(L2,2) + pow(L3,2) - pow(R,2))/(2*L2*L3)))*(180.00/Pi)); //theta 3
  Theta_4 = 180.00 - ((180.00 - (alpha2 + Theta_3)) + alpha1); //theta 4
```

```
}
Else if (X_End_Effector == 0 && Y_End_Effector >= L1)
{
  D = sqrt(pow(X_End_Effector,2) + pow(Z_End_Effector,2));
  Theta_1 = 90.00; //theta 1
  D = D - L4;
  Yoffset = Y_End_Effector - L1;
  R = sqrt(pow(d,2) + pow(Yoffset,2));
  Alpha1 = (acos(d/R))*(180.00/Pi);
  Alpha2 = (acos((pow(L2,2) + pow(R,2) - pow(L3,2))/(2*L2*R)))*(180.00/Pi);
  Theta_2 = (alpha1 + alpha2); //theta 2
  Theta_3 = ((acos((pow(L2,2) + pow(L3,2) - pow(R,2))/(2*L2*L3)))*(180.00/Pi)); //theta 3
  Theta_4 = 180.00 - ((180.00 - (alpha2 + Theta_3)) - alpha1); //theta 4
}
Else if (X_End_Effector == 0 && Y_End_Effector <= L1)
{
  D = sqrt(pow(X_End_Effector,2) + pow(Z_End_Effector,2));
  Theta_1 = 90.00; //theta 1
  D = D - L4;
  Yoffset = Y_End_Effector - L1;
  R = sqrt(pow(d,2) + pow(Yoffset,2));
  Alpha1 = (acos(d/R))*(180.00/Pi);
  Alpha2 = (acos((pow(L2,2) + pow(R,2) - pow(L3,2))/(2*L2*R)))*(180.00/Pi);
  Theta_2 = (alpha2 - alpha1); //theta 2
  Theta_3 = ((acos((pow(L2,2) + pow(L3,2) - pow(R,2))/(2*L2*L3)))*(180.00/Pi)); //theta 3
  Theta_4 = 180.00 - ((180.00 - (alpha2 + Theta_3)) + alpha1); //theta 4
}
Else if (X_End_Effector < 0 && Y_End_Effector >= L1)
{
  D = sqrt(pow(X_End_Effector,2) + pow(Z_End_Effector,2));
  Theta_1 = 90.00 + (90.00 - abs((atan(Z_End_Effector/X_End_Effector))*(180.00/Pi))); //theta
1
  D = D - L4;
  Yoffset = Y_End_Effector - L1;
  R = sqrt(pow(d,2) + pow(Yoffset,2));
  Alpha1 = (acos(d/R))*(180.00/Pi);
  Alpha2 = (acos((pow(L2,2) + pow(R,2) - pow(L3,2))/(2*L2*R)))*(180.00/Pi);
  Theta_2 = (alpha1 + alpha2); //theta 2
  Theta_3 = ((acos((pow(L2,2) + pow(L3,2) - pow(R,2))/(2*L2*L3)))*(180.00/Pi)); //theta 3
  Theta_4 = 180.00 - ((180.00 - (alpha2 + Theta_3)) - alpha1); //theta 4
}
Else if (X_End_Effector < 0 && Y_End_Effector <= L1)
{
  D = sqrt(pow(X_End_Effector,2) + pow(Z_End_Effector,2));
```

```
    Theta_1 = 90.00 + (90.00 - abs((atan(Z_End_Effector/X_End_Effector))*(180.00/Pi))); //theta
1
    D = D - L4;
    Yoffset = Y_End_Effector - L1;
    R = sqrt(pow(d,2) + pow(Yoffset,2));
    Alpha1 = (acos(d/R))*(180.00/Pi);
    Alpha2 = (acos((pow(L2,2) + pow(R,2) - pow(L3,2))/(2*L2*R)))*(180.00/Pi);
    Theta_2 = (alpha2 - alpha1); //theta 2
    Theta_3 = ((acos((pow(L2,2) + pow(L3,2) - pow(R,2))/(2*L2*L3)))*(180.00/Pi)); //theta 3
    Theta_4 = 180.00 - ((180.00 - (alpha2 + Theta_3)) + alpha1); //theta 4
  }
}

Void setup()
{
  Serial.begin(9600);
  Pinmode(Trigger, OUTPUT);
  Pinmode(Echo, INPUT);

  Gripper.attach(pin_gripper); //gripper
  Servo1.attach(pin_servo1 );
  Servo2.attach(pin_servo2);
  Servo3.attach(pin_servo3);
  Servo4.attach(pin_servo4);
}
Void loop()
{
  Digitalwrite(Trigger, LOW);
  Delaymicroseconds(2);
  Digitalwrite(Trigger, HIGH);
  Delaymicroseconds(10);
  Digitalwrite(Trigger, LOW);
  Time = pulsein(Echo, HIGH);
  Distance = Time*0.034/2;
  If(Serial.available())
  { image= Serial.read();}

  If(Distance > 10.00)
  {
    Gripper.write(150);
    Delay(200);
    X_End_Effector = 180.00;
    Z_End_Effector = 10.00;
    Y_End_Effector = 150.00;
```

```
 Rumus_IK();
 Servo1.write(Theta_1);
 Servo2.write(Theta_2);
 Servo3.write(Theta_3);
 Servo4.write(Theta_4 - 45.00);
 Delay(100);
}
Else if(Distance <= 10.00&& image==1 )
{
 Gripper.write(90);
 For(Y_End_Effector = 150.00; Y_End_Effector >= 40.00; Y_End_Effector -= 0.10)
 {
  X_End_Effector = 180.00;
  Z_End_Effector = 10.00;
  Rumus_IK();
  Servo1.write(Theta_1);
  Servo2.write(Theta_2);
  Servo3.write(Theta_3);
  Servo4.write(Theta_4 - 45.00);
  Delaymicroseconds(100);
 }
 Delay(300);
 Gripper.write(120);
 Delay(300);
 For(Y_End_Effector = 40.00; Y_End_Effector <= 150.00; Y_End_Effector += 0.10)
 {
  X_End_Effector = 180.00;
  Z_End_Effector = 10.00;
  Rumus_IK();
  Servo1.write(Theta_1);
  Servo2.write(Theta_2);
  Servo3.write(Theta_3);
  Servo4.write(Theta_4 - 45.00);
  Delaymicroseconds(100);
 }
 For(Z_End_Effector = 10.00; Z_End_Effector <= 100.00; Z_End_Effector += 0.10)
 {
  Y_End_Effector = 150.00;
  X_End_Effector = 180.00;
  Rumus_IK();
  Servo1.write(Theta_1);
  Servo2.write(Theta_2);
  Servo3.write(Theta_3);
  Servo4.write(Theta_4 - 45.00);
  Delaymicroseconds(100);
}
For(X_End_Effector = 180.00; X_End_Effector >= -50.00; X_End_Effector -= 0.10)
{
  Y_End_Effector = 150.00;
  Z_End_Effector = 100.00;
  Rumus_IK();
  Servo1.write(Theta_1);
  Servo2.write(Theta_2);
  Servo3.write(Theta_3);
  Servo4.write(Theta_4 - 45.00);
  Delaymicroseconds(100);
}
For(Z_End_Effector = 100.00; Z_End_Effector <= 250.00; Z_End_Effector += 0.10)
{
  Y_End_Effector = 150.00;
  X_End_Effector = -50.00;
  Rumus_IK();
  Servo1.write(Theta_1);
  Servo2.write(Theta_2);
  Servo3.write(Theta_3);
  Servo4.write(Theta_4 - 45.00);
  Delaymicroseconds(100);
}
For(Y_End_Effector = 150.00; Y_End_Effector >= 50.00; Y_End_Effector -= 0.10)
{
  X_End_Effector = -50.00;
  Z_End_Effector = 250.00;
  Rumus_IK();
  Servo1.write(Theta_1);
  Servo2.write(Theta_2);
  Servo3.write(Theta_3);
  Servo4.write(Theta_4 - 45.00);
  Delaymicroseconds(100);
}
Delay(100);
Gripper.write(90);
Delay(100);
For(Y_End_Effector = 50.00; Y_End_Effector <= 150.00; Y_End_Effector += 0.10)
{
  X_End_Effector = -50.00;
  Z_End_Effector = 250.00;
  Rumus_IK();
  Servo1.write(Theta_1);
```

```
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
  }
  Delay(100);
}
Else if (Distance <= 10.00 && image==0 )
{
  Gripper.write(90);
  For(Y_End_Effector = 150.00; Y_End_Effector >= 40.00; Y_End_Effector -= 0.10)
  {
    X_End_Effector = 180.00;
    Z_End_Effector = 10.00;
    Rumus_IK();
    Servo1.write(Theta_1);
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
  }
  Delay(300);
  Gripper.write(120);
  Delay(300);
  For(Y_End_Effector = 40.00; Y_End_Effector <= 150.00; Y_End_Effector += 0.10)
  {
    X_End_Effector = 180.00;
    Z_End_Effector = 10.00;
    Rumus_IK();
    Servo1.write(Theta_1);
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
  }
  For(Z_End_Effector = 10.00; Z_End_Effector <= 100.00; Z_End_Effector += 0.10)
  {
    Y_End_Effector = 150.00;
    X_End_Effector = 180.00;
    Rumus_IK();
    Servo1.write(Theta_1);
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
  }
  For(X_End_Effector = 180.00; X_End_Effector >= -150.00; X_End_Effector -= 0.10)
  {
    Y_End_Effector = 150.00;
    Z_End_Effector = 100.00;
    Rumus_IK();
    Servo1.write(Theta_1);
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
  }
  For(Z_End_Effector = 100.00; Z_End_Effector <= 250.00; Z_End_Effector += 0.10)
  {
    Y_End_Effector = 150.00;
    X_End_Effector = -150.00;
    Rumus_IK();
    Servo1.write(Theta_1);
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
  }
  For(Y_End_Effector = 150.00; Y_End_Effector >= 60.00; Y_End_Effector -= 0.10)
  {
    X_End_Effector = -150.00;
    Z_End_Effector = 250.00;
    Rumus_IK();
    Servo1.write(Theta_1);
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
  }
  Delay(100);
  Gripper.write(90);
  Delay(100);
  For(Y_End_Effector = 60.00; Y_End_Effector <= 150.00; Y_End_Effector += 0.10)
  {
    X_End_Effector = -150.00;
    Z_End_Effector = 250.00;
    Rumus_IK();
    Servo1.write(Theta_1);
```

```
    Servo2.write(Theta_2);
    Servo3.write(Theta_3);
    Servo4.write(Theta_4 - 45.00);
    Delaymicroseconds(100);
   }
  Delay(100);
 }
 }


//first dimension = 3,1cm x 3,4cm x 5,1cm
//second dimension = 3,1cm x 3,4cm x 4,6cm
```

Another controlling code

```
#include <Servo.h>
Servo myservo1 ;
Servo myservo2;
Servo myservo3;
Servo myservo4;
Servo myservo5;
Servo gripper;
 Int angle1=90;
Int angle2=40;
Int angle3=40;
Int angle4=160;
Int angle5=90;
Int angle1min=0;
Int angle1max=180;
Int angle2min=0;
Int angle2max=180;
Int angle3min=0;
Int angle3max=180;
Int angle4min=0;
Int angle4max=180;
Int angle5min=0;
Int angle5max=180;
#define Echo 7
#define Trigger 6
Long Time;
Float Distance;
```

```
Int image;

Void setup() {

 Serial.begin(9600);
  Pinmode(Trigger, OUTPUT);
  Pinmode(Echo, INPUT);

Myservo1.attach(8,500,2500);
Myservo2.attach(9,500,2500);
Myservo3.attach(10,500,2500);
Myservo4.attach(11,500,2500);
Myservo5.attach(12,500,2500);
Gripper.attach(13,500,2500);
}
Void loop() {
  Digitalwrite(Trigger, LOW);
  Delaymicroseconds(2);
  Digitalwrite(Trigger, HIGH);
  Delaymicroseconds(10);
  Digitalwrite(Trigger, LOW);
  Time = pulsein(Echo, HIGH);
  Distance = Time*0.034/2;
  If(Serial.available())
  { image= Serial.read();}
If (Distance>10)
{
Myservo1.write(angle1);
Myservo2.write(angle2);
Myservo3.write(angle3);
Myservo4.write(angle4);
Myservo5.write(angle5);

Gripper.write(20);
Delay (1000);
}
 Else if(Distance <= 10.00&& image==1 )
{
 Gripper.write(150);

Delay(5000);
Angle2=10;
Angle3=10;
Angle4=50;
```

```
Angle5=170;

Myservo2.write(angle2);
Myservo3.write(angle3);
Delay(5000);
Myservo4.write(angle4);
Myservo5.write(angle5);
Delay (5000);
Angle1=0;
Myservo1.write(angle1);
Delay( 5000);
Angle2=70;
Angle3=70;
Myservo2.write(angle2);
Myservo3.write(angle3);
Angle4=110;
Angle5=110;
Myservo4.write(angle4);
Myservo5.write(angle5);

Delay(5000);
Gripper.write(30);
Delay(5000);
}
 Else if (Distance <= 10.00 && image==0 )
 {gripper.write(150);
Delay(5000);
Angle2=10;
Angle3=10;
Angle4=50;
Angle5=170;

Myservo2.write(angle2);
Myservo3.write(angle3);
Delay(5000);
Myservo4.write(angle4);
Myservo5.write(angle5);
Delay (5000);
Angle1=180;
Myservo1.write(angle1);
Delay( 5000);
Angle2=70;
Angle3=70;
Myservo2.write(angle2);
```

```
Myservo3.write(angle3);
Angle4=110;
Angle5=110;
Myservo4.write(angle4);
Myservo5.write(angle5);
Delay(5000);
Gripper.write(30);
Delay(5000);
}

Angle2=30;
 Angle3=30;
 Delay(5000);

 Angle4=170;
 Angle5=90;
 Delay(1000);
Angle1=90;
Delay(5000);
        }
```