```
#include <Keypad.h>
#include <Servo.h>          //include the servo library
#include <Adafruit_Fingerprint.h>

#if (defined(__AVR__) || defined(ESP8266)) && !
defined(__AVR_ATmega2560__)
SoftwareSerial mySerial(2, 3);
#endif

int potPosition;           //this variable will store the position
of the potentiometer
int servoPosition;         //the servo will move to this position

Servo myservo;              //create a servo object

char const *password = "123231";
int position = 0;
const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'0','F','E','D'}
};

byte rowPins[ROWS] = { 5, 4, 3, 2 };//Pin may change according to
sutability
byte colPins[COLS] = { 9, 8, 7, 6 };
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS,
COLS );
int RedpinLock = 12;
int GreenpinUnlock = 13;

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);

void setup()
```

```
{
pinMode(RedpinLock, OUTPUT);
pinMode(GreenpinUnlock, OUTPUT);
LockedPosition(true);
myservo.attach(11);
Serial.begin(9600);
}

void loop()
{
char key = keypad.getKey();
Serial.println(key);
if (key == 'A' || key == 'B')
{
position = 0;
LockedPosition(true);
}
if (key == password[position])
{ while (!Serial);
delay(100);
  Serial.println("\n\nAdafruit finger detect test");

  // set the data rate for the sensor serial port
  finger.begin(57600);
  delay(5);
  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1) { delay(1); }
  }

  Serial.println(F("Reading sensor parameters"));
  finger.getParameters();
  Serial.print(F("Status: 0x")); Serial.println(finger.status_reg,
HEX);
  Serial.print(F("Sys ID: 0x")); Serial.println(finger.system_id,
HEX);
  Serial.print(F("Capacity: ")); Serial.println(finger.capacity);
  Serial.print(F("Security level: ")); Serial.println(finger.
security_level);
  Serial.print(F("Device address: ")); Serial.println(finger.
device_addr, HEX);
  Serial.print(F("Packet len: ")); Serial.println(finger.
packet_len);
  Serial.print(F("Baud rate: ")); Serial.println(finger.baud_rate);

  finger.getTemplateCount();

  if (finger.templateCount == 0) {
    Serial.print("Sensor doesn't contain any fingerprint data.
Please run the 'enroll' example.");
  }
  else {
    Serial.println("Waiting for valid finger...");
      Serial.print("Sensor contains "); Serial.print(finger.
templateCount); Serial.println(" templates");
  }
}

void loop()                        // run over and over again
{
  getFingerprintID();
  delay(50);            //don't ned to run this at full speed.
}

uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println("No finger detected");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  // OK success!

  p = finger.image2Tz();
  switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
      Serial.println("Image too messy");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
      Serial.println("Could not find fingerprint features");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
      Serial.println("Could not find fingerprint features");
      return p;
    default:
      Serial.println("Unknown error");
      return p;
  }

  // OK converted!
```

```
  p = finger.fingerSearch();
  if (p == FINGERPRINT_OK) {
    Serial.println("Found a print match!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_NOTFOUND) {
    Serial.println("Did not find a match");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }

  // found a match!
  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.
confidence);

  return finger.fingerID;
}

// returns -1 if failed, otherwise returns ID #
int getFingerprintIDez() {
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.image2Tz();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK)  return -1;

  // found a match!
  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.
confidence);
```

```
  return finger.fingerID;
}
}
if (finger.fingerID == 1)
{
position ++;
}
if (position == 6)
{
LockedPosition(false);
}
delay(100);
Serial.println(position);
}
void LockedPosition(bool locked)
{
if (locked)
{
digitalWrite(RedpinLock, HIGH);
digitalWrite(GreenpinUnlock, LOW);
myservo.write(105);
}
else
{
digitalWrite(RedpinLock, LOW);
digitalWrite(GreenpinUnlock, HIGH);
myservo.write(20);
}
}
```