CODING SENSOR SUARA

```
int Led=13;//Definisi Led pada pin 13 (default)
int OutputDO=2; //Definisi pin 2 sebagai keluaran sensor
int Buzzer1 =12 ; //untuk Buzzer
int Buzzer2 = 11; //untuk Buzzer
int val;//val sebagai buffer data

void setup(){
//Inisialisasi I/O
pinMode(Led,OUTPUT);
pinMode(Buzzer1, OUTPUT);
pinMode(Buzzer2, OUTPUT);
pinMode(OutputDO,INPUT);
digitalWrite(Led,LOW);
digitalWrite(Buzzer1,LOW);
digitalWrite(Buzzer2,LOW);
}

void loop(){
//Membaca sinyal keluaran dari sensor berupa logika 1 atau 0
val=digitalRead(OutputDO);
if(val==HIGH) {
//Jika berlogika 1 maka LED akan menyala
digitalWrite(Led,HIGH);
digitalWrite(Buzzer1, HIGH);
digitalWrite(Buzzer2, HIGH);}
else
{
//jika berlogika 0 maka LED akan mati
digitalWrite(Led,LOW);
digitalWrite(Buzzer1, LOW);
digitalWrite(Buzzer2, LOW);
}?
}
```

Coding Dot Matrix

```
#include <MaxMatrix.h>

#include <avr/pgmspace.h>

const unsigned char CH[] PROGMEM= {

3, 8, B00000000, B00000000, B00000000, B00000000, B00000000, // space

1, 8, B01011111, B00000000, B00000000, B00000000, B00000000, // !

3, 8, B00000011, B00000000, B00000011, B00000000, B00000000, // "

5, 8, B00010100, B00111110, B00010100, B00111110, B00010100, // #

4, 8, B00100100, B01101010, B00101011, B00010010, B00000000, // $

5, 8, B01100011, B00010011, B00001000, B01100100, B01100011, // %

5, 8, B01101100, B10010010, B10101100, B01000000, B10100000, // &

1, 8, B00000011, B00000000, B00000000, B00000000, B00000000, // '

3, 8, B00011100, B00100010, B01000001, B00000000, B00000000, // (

3, 8, B01000001, B00100010, B00011100, B00000000, B00000000, // )

5, 8, B00101000, B00011000, B00001110, B00011000, B00101000, // *

5, 8, B00001000, B00001000, B00111110, B00001000, B00001000, // +

2, 8, B10110000, B01110000, B00000000, B00000000, B00000000, // ,

4, 8, B00001000, B00001000, B00001000, B00001000, B00000000, // -

2, 8, B01100000, B01100000, B00000000, B00000000, B00000000, // .

4, 8, B01100000, B00011000, B00000110, B00000001, B00000000, // /

4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // 0

3, 8, B01000010, B01111111, B01000000, B00000000, B00000000, // 1

4, 8, B01100010, B01010001, B01001001, B01000110, B00000000, // 2

4, 8, B00100010, B01000001, B01001001, B00110110, B00000000, // 3

4, 8, B00011000, B00010100, B00010010, B11111111, B00000000, // 4

4, 8, B00100111, B01000101, B01000101, B00111001, B00000000, // 5
```

```
4, 8, B00111110, B01001001, B01001001, B00110000, B00000000, // 6
4, 8, B01100001, B00010001, B00001001, B00000111, B00000000, // 7
4, 8, B00110110, B01001001, B01001001, B00110110, B00000000, // 8
4, 8, B00000110, B01001001, B01001001, B00111110, B00000000, // 9
2, 8, B00010100, B00000000, B00000000, B00000000, B00000000, // :
2, 8, B10000000, B01010000, B00000000, B00000000, B00000000, // ;
3, 8, B00010000, B00101000, B01000100, B00000000, B00000000, // <
3, 8, B00010100, B00010100, B00010100, B00000000, B00000000, // =
3, 8, B01000100, B00101000, B00010000, B00000000, B00000000, // >
4, 8, B00000010, B01011001, B00001001, B00000110, B00000000, // ?
5, 8, B00111110, B01001001, B01010101, B01011101, B00001110, // @
4, 8, B11111110, B00001001, B00001001, B11111110, B00000000, // A
4, 8, B11111111, B10001001, B10001001, B01110110, B00000000, // B
4, 8, B01111110, B10000001, B10000001, B01000010, B00000000, // C
4, 8, B11111111, B10000001, B10000001, B01111110, B00000000, // D
4, 8, B11111111, B10001001, B10001001, B10000001, B00000001, // E
4, 8, B11111111, B00001001, B00001001, B00000001, B00000000, // F
4, 8, B01111110, B10000001, B10001001, B01111010, B00000000, // G
4, 8, B11111111, B00001000, B00001000, B11111111, B00000000, // H
3, 8, B10000001, B11111111, B10000001, B00000000, B00000000, // I
4, 8, B01100000, B10000000, B10000001, B01111111, B00000000, // J
4, 8, B11111111, B00001000, B00010100, B11100011, B00000000, // K
4, 8, B11111111, B10000000, B10000000, B10000000, B00000000, // L
5, 8, B11111111, B00000010, B00001100, B00000010, B11111111, // M
5, 8, B11111111, B00000100, B00001000, B00010000, B11111111, // N
4, 8, B01111110, B10000001, B10000001, B01111110, B00000000, // O
4, 8, B11111111, B00001001, B00001001, B00000110, B00000000, // P
4, 8, B00111110, B01000001, B01000001, B10111110, B00000000, // Q
4, 8, B11111111, B00001001, B00001001, B11110110, B00000000, // R
```

```
4, 8, B01000110, B10001001, B10001001, B01110010, B00000000, // S
5, 8, B00000001, B00000001, B11111111, B00000001, B00000001, // T
4, 8, B01111111, B10000000, B10000000, B01111111, B00000000, // U
5, 8, B00011111, B01100000, B10000000, B01100000, B00011111, // V
5, 8, B01111111, B10000000, B01110000, B10000000, B01111111, // W
5, 8, B11100011, B00010100, B00001000, B00010100, B01110011, // X
5, 8, B00000111, B00001000, B11110000, B00001000, B00000111, // Y
4, 8, B01100001, B01010001, B01001001, B10000111, B00000000, // Z
2, 8, B01111111, B01000001, B00000000, B00000000, B00000000, // [
4, 8, B00000001, B00000110, B00011000, B01100000, B00000000, // \ backslash
2, 8, B01000001, B01111111, B00000000, B00000000, B00000000, // ]
3, 8, B00000010, B00000001, B00000010, B00000000, B00000000, // hat
4, 8, B01000000, B01000000, B01000000, B01000000, B00000000, // _
2, 8, B00000001, B00000010, B00000000, B00000000, B00000000, // `
4, 8, B01000000, B10101000, B10101000, B11110000, B00000000, // a
4, 8, B11111111, B10001000, B10001000, B01110000, B00000000, // b
4, 8, B01110000, B10001000, B10001000, B01010000, B00000000, // c
4, 8, B01110000, B10001000, B10001000, B11111111, B00000000, // d
4, 8, B01110000, B10101000, B10101000, B10010000, B00000000, // e
3, 8, B00001000, B11111100, B00001010, B00000000, B00000000, // f
4, 8, B10011000, B10100100, B10100100, B01111000, B00000000, // g
4, 8, B11111111, B00001000, B00001000, B11110000, B00000000, // h
3, 8, B10001000, B11111010, B10000000, B00000000, B00000000, // i
4, 8, B01000000, B10000000, B10000100, B01111101, B00000000, // j
4, 8, B11111111, B00100000, B01010000, B10001000, B00000000, // k
3, 8, 100000001, B11111111, B10000000, B00000000, B00000000, // l
5, 8, B11111000, B00001000, B11111000, B00001000, B11110000, // m
4, 8, B11111000, B00001000, B00001000, B11110000, B00000000, // n
4, 8, B01110000, B10001000, B10001000, B01110000, B00000000, // o
```

```
4, 8, B11111100, B00100100, B00100100, B00011000, B00000000, // p
4, 8, B00011000, B00100100, B00100100, B11111100, B00000000, // q
4, 8, B11111000, B00010000, B00001000, B00001000, B00000000, // r
4, 8, B10010000, B10101000, B10101000, B01001000, B00000000, // s
3, 8, B00001000, B01111110, B10001000, B00000000, B00000000, // t
4, 8, B01111000, B10000000, B10000000, B11111000, B00000000, // u
5, 8, B00111000, B01000000, B10000000, B01000000, B00111000, // v
5, 8, B01111000, B10000000, B01111000, B10000000, B01111000, // w
5, 8, B10001000, B01010000, B00100000, B01010000, B10001000, // x
4, 8, B10011100, B10100000, B10100000, B01111100, B00000000, // y
3, 8, B11001000, B10101000, B10011000, B10001000, B00000000, // z
3, 8, B00001000, B00110110, B01000001, B00000000, B00000000, // {
1, 8, B01111111, B00000000, B00000000, B00000000, B00000000, // |
3, 8, B01000001, B00110110, B00001000, B00000000, B00000000, // }
4, 8, B00001000, B00000100, B00001000, B00000100, B00000000, // ~
};

int data = 12;    // DIN pin of MAX7219 module
int load = 10;    // CS pin of MAX7219 module
int clock = 11;  // CLK pin of MAX7219 module



int maxInUse = 5;  //how many MAX7219 are connected

MaxMatrix m(data, load, clock, maxInUse); // define Library
byte buffer[10];

char string1[] = " HARAP TENANG  ";  // Scrolling Text
```

```
}

void setup(){
  m.init(); // module MAX7219
  m.setIntensity(5); // LED Intensity 0-15

}

void loop(){


  byte c;
  delay(150);
  m.shiftLeft(false, true);
  printStringWithShift(string1, 150);  // Send scrolling Text
}
// Put extracted character on Display
void printCharWithShift(char c, int shift_speed){
  if (c < 32) return;
  c -= 32;
  memcpy_P(buffer, CH + 7*c, 7);
  m.writeSprite(maxInUse*8, 0, buffer);
  m.setColumn(maxInUse*8 + buffer[0], 0);

  for (int i=0; i<buffer[0]+1; i++)
  {
    delay(shift_speed);
    m.shiftLeft(false, false);
  }
}
// Extract characters from Scrolling text
```

```
void printStringWithShift(char* s, int shift_speed){
  while (*s != 0){
    printCharWithShift(*s, shift_speed);
    s++;
  }
}
```