4)HeartRate Sensor

```
#define USE_ARDUINO_INTERRUPTS true
#include <PulseSensorPlayground.h>
const int PULSE_SENSOR_PIN = A0;
const int LED_PIN = 13;
const int THRESHOLD = 550;
PulseSensorPlayground pulseSensor;
void setup() {
 Serial.begin(9600);
 pulseSensor.analogInput(PULSE_SENSOR_PIN);
 pulseSensor.blinkOnPulse(LED_PIN);
 pulseSensor.setThreshold(THRESHOLD);
 if (pulseSensor.begin()) {
  Serial.println("PulseSensor initialized successfully!");
 } else {
  Serial.println("PulseSensor initialization failed!");}}
void loop() {
 int currentBPM = pulseSensor.getBeatsPerMinute();
 if (pulseSensor.sawStartOfBeat()) {
  Serial.println("❤ Heartbeat detected!");
  Serial.print("BPM: ");
  Serial.println(currentBPM);
 }
 delay(20);
}
```

5)Temperature,humidity (Software)

```
int tmp = A0;
int pot = A1;
double vol = 0, tc = 0, tf = 0;
void setup() {
  pinMode(pot, INPUT);
  Serial.begin(9600);}
void loop() {
  int rv = analogRead(tmp);
  vol = (rv / 1023.0) * 5000;
  tc = (vol - 500) * 0.1;
  tf = (tc * 1.8) + 32;
  Serial.println("--- Temperature Sensor Readings ---");
  Serial.print("Voltage: "); Serial.println(vol);
  Serial.print("Temperature (Celsius): "); Serial.println(tc);
  Serial.print("Temperature (Fahrenheit): "); Serial.println(tf);
  Serial.println();
  int hu = analogRead(pot);
  int humidity = map(hu, 0, 1023, 10, 70);
  Serial.println("--- Humidity Sensor Readings ---");
  Serial.print("Raw Value: "); Serial.println(hu);
  Serial.print("Humidity (%): "); Serial.println(humidity);
  Serial.println("--------------------------------");
  delay(2000);
}
```

6)Ultrasonic sensor

```
int trig = 6;
int echo = 7;
int led = 13;
void setup(){
  pinMode(led, OUTPUT);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);}
void loop()
{
  digitalWrite(trig,LOW);
  delayMicroseconds(2);
  digitalWrite(trig,HIGH);
  delayMicroseconds(10);
  digitalWrite(trig,LOW);
  float a = pulseIn(echo,HIGH);
  float v = a*0.034/2;
  if(v<50){
    digitalWrite(led,HIGH);
  }
  if(v>50){
    digitalWrite(led,LOW);
  }
        delay(5000);
}
```

2 and 7) IR SENSOR and PIR sensor

```
int pir = 7;
int led = 12;
void setup()
{
 pinMode(led, OUTPUT);
 pinMode(pir, INPUT);
 Serial.begin(9600);
}
void loop()
{
 int v = digitalRead(pir);
 if(v==HIGH){
  digitalWrite(led,HIGH);
  Serial.println("Motion is Detected");
 }
 else{
  digitalWrite(led,LOW);
  Serial.println("Motion Is Not detected");
 }
 delay(3000);
}
```

8)Gas detection

```
const int gasSensorPin = A0;
const int buzzerPin = 9;
const int threshold = 300;
void setup() {
 pinMode(buzzerPin, OUTPUT);
 pinMode(gasSensorPin, INPUT);
 digitalWrite(buzzerPin, LOW);
 Serial.begin(9600);
}
void loop() {
 int sensorValue = analogRead(gasSensorPin);
 Serial.print("Gas Sensor Value: ");
 Serial.println(sensorValue);
 if(sensorValue > threshold) {
  digitalWrite(buzzerPin, HIGH);
 }
 else {
  digitalWrite(buzzerPin, LOW);
 }
 delay(500);
}
```

1.a) GSM Module

```
#include <SoftwareSerial.h>
SoftwareSerial sim(10, 11);
void setup() {
 Serial.begin(9600);
 sim.begin(9600);
 Serial.println("SIM900A Ready. Type 's' to send, 'r' to receive.");}
void loop() {
 if (Serial.available()) {
  char cmd = Serial.read();
  if (cmd == 's') sendMessage();
  if (cmd == 'r') receiveMessage();}
 if (sim.available()) Serial.write(sim.read());}
void sendMessage() {
 Serial.println("Sending Message...");
 sim.println("AT+CMGF=1");
 delay(100);
 sim.println("AT+CMGS=\"+917904329949\"");
 delay(100);
 sim.println("Good morning!");
 sim.write(26);
 Serial.println("Message Sent!");
}
void receiveMessage() {
 sim.println("AT+CNMI=2,2,0,0,0");
 Serial.println("Waiting for messages...");
}
```

1.b) Bluetooth

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial(3, 2);
void setup() {
  Serial.begin(9600);
  mySerial.begin(9600);
  Serial.println("Initializing...");
  Serial.println("The device started, now you can pair it with Bluetooth!");
}
void loop() {
  if (Serial.available()) {
    mySerial.write(Serial.read());
  }
  if (mySerial.available()) {
    Serial.write(mySerial.read());
  }
  delay(20);
}
```

3. RaspberryPi

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
def RCtime(RCpin):
    reading = 0
    GPIO.setup(RCpin, GPIO.OUT)
    GPIO.output(RCpin, GPIO.LOW)
    time.sleep(2)
    GPIO.setup(RCpin, GPIO.IN)
    while GPIO.input(RCpin) == GPIO.LOW:
        reading += 1
    return reading
try:
    while True:
        print(RCtime(12))
        time.sleep(1)
except KeyboardInterrupt:
    GPIO.cleanup()
    print("Exiting...")
```