Day 1 Programs

1.Multiple LED Blink

```
int led1 =4;
// initialise a variable named led1 (i.e. for the LED connected to GPIO 4)
int led2 =2;
// initialise a variable named led2 (i.e. for the LED connected to GPIO 2)
void setup() {
pinMode(led1,OUTPUT); // intitialise GPIO as output
pinMode(led2,OUTPUT); // intitialise GPIO as output
}
void loop() {
digitalWrite(led1,HIGH); // turn ON first LED
digitalWrite(led2,LOW); // turn OFF second LED
delay(1000); // wait for a second
digitalWrite(led1,LOW); // turn OFF first LED
digitalWrite(led2,HIGH); // turn ON second LED
delay(1000); // wait for a second
}
```

2.Blinking LED using a switch (push button)

```
int buttonpin=4; // the number of the pushbutton pin
int ledpin=2; // the number of the LED pin
int buttonnew; // variable to store present status of pushbutton
int buttonold=1; // variable to store previous status of pushbutton
int ledstate=0; // initial status of led
void setup() {
 // initialize the LED pin as an output:
 pinMode(ledpin,OUTPUT);
 // initialize the pushbutton pin as an input:
 pinMode(buttonpin,INPUT);
}
void loop() {
 // read the state of the pushbutton value:
 buttonnew=digitalRead(buttonpin);
 // check whether the pushbutton is pressed and released
if(buttonold==0 && buttonnew==1){
 // check whether ledstate is LOW
 if(ledstate==0){
 // turn LED on:
 digitalWrite(ledpin,HIGH);
 // update ledstate
 ledstate=1;
 }
 else
 {
 // turn LED off:
 digitalWrite(ledpin,LOW);
 // update ledstate
 ledstate=0;
 }
 }
 // update buttonold state with buttonnew state
 buttonold=buttonnew;
 delay(100);
}
```

3.Interfacing with a Pushbutton while making an LED blink independently

```
const int buttonPin=4; // the number of the pushbutton pin
const int ledPin= 2; // the number of the LED pin
int led =15; // initialise a variable named led
// variables will change:
int buttonState=0; // variable for reading the pushbutton status
void setup() {
// initialize the LED pin as an output:
pinMode(ledPin,OUTPUT);
// initialize the pushbutton pin as an input:
pinMode(buttonPin,INPUT);
pinMode(led,OUTPUT); //configure the pin as an output
}
void loop() {
// read the state of the pushbutton value:
buttonState=digitalRead(buttonPin);
// check if the pushbutton is pressed. If it is, the buttonState is HIGH:
if (buttonState==HIGH) {
 // turn LED on:
 digitalWrite(ledPin,HIGH);
} else {
 // turn LED off:
 digitalWrite(ledPin,LOW);
}
digitalWrite(led,HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(led,LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}
```

4. Interfacing a Pushbutton while making an LED blink independently without using the ‘delay’ function

```
const int ledPin= 2;// the number of the LED pin
const int buttonPin=4; // the number of the pushbutton pin
const int BledPin= 15; // the number of the Blinking LED pin
// Variables will change:
int ledState=LOW; // ledState used to set the LED
bool buttonState=0;
// Generally, you should use "unsigned long" for variables that hold time
// The value will quickly become too large for an int to store
unsigned long previousMillis=0; // will store last time LED was updated
// constants won't change:
const long interval =1000; // interval at which to blink (milliseconds)
void setup() {
// set the digital pin as output:
pinMode(ledPin,OUTPUT);
pinMode(BledPin,OUTPUT);
pinMode(buttonPin,INPUT);
}
void loop() {
// check to see if it's time to blink the LED; that is, if the difference
// between the current time and last time you blinked the LED is bigger than
// the interval at which you want to blink the LED.
unsigned long currentMillis=millis();
if (currentMillis-previousMillis>= interval) {
 // save the last time you blinked the LED
 previousMillis=currentMillis;
 // if the LED is off turn it on and vice-versa:
 if (ledState==LOW) {
 ledState=HIGH;
 } else {
 ledState=LOW;
 }
 // set the LED with the ledState of the variable:
 digitalWrite(BledPin,ledState);
}
// read the state of the pushbutton value:
buttonState=digitalRead(buttonPin);
// check if the pushbutton is pressed. If it is, the buttonState is HIGH:
if (buttonState==HIGH) {
 // turn LED on:
 digitalWrite(ledPin,HIGH);
} else {
 // turn LED off:
 digitalWrite(ledPin,LOW);
}
}
```

USING SERIAL MONITOR

1. Turning ON LED using serial monitor

```
int LED = 2;
void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(9600); // Open the serial port at 9600 bps
}
void loop() {
  if (Serial.available() > 0) {
    // Read the incoming string:
    String userInput = Serial.readString();
    userInput.trim(); // Remove any leading/trailing whitespace
if (userInput == "ON") {
      digitalWrite(LED, HIGH);
    }
    else if (userInput == "OFF") { // Fixed 'else if' syntax
      digitalWrite(LED, LOW);
    }
    else {
      Serial.println("Invalid Input");
    }
  }
}
```

ANALOG PROGRAMS

1.Night Light Using LDR

```
int led = 2;
int ADCvalue;
int ADCpin= 4;
void setup() {
pinMode(ADCpin,INPUT);
pinMode(led,OUTPUT);
}
void loop() {
ADCvalue=analogRead(ADCpin);
if(ADCvalue> 1600) {
 digitalWrite(led,HIGH);
}
else {
 digitalWrite(led,LOW);
}
}
```

2. LED Fade / Breathing LED

```
const int ledPin = 4; // LED connected to GPIO 4

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // Increase brightness
  for (int brightness = 0; brightness <= 255; brightness++) {
    analogWrite(ledPin, brightness);
    delay(10);
  }

  // Decrease brightness
  for (int brightness = 255; brightness >= 0; brightness--) {
    analogWrite(ledPin, brightness);
    delay(10);
  }
}
```

Interfacing with Sensors and Displays

1. Interfacing with an LCD Display

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// Initialize the LCD (I2C address 0x27, 16 columns, 2 rows)
LiquidCrystal_I2C lcd(0x27, 16, 2);
int x = 0;
void setup() {
 lcd.init(); // Initialize the LCD
 lcd.backlight(); // Turn on the LCD backlight
 lcd.print("Hello!");
}
void loop() {
 lcd.setCursor(0, 1);
 lcd.print(x);
 x++;
 delay(1000);
}
```

2. Measuring Voltages Higher than 3.3V

```
int ADCvalue;
int ADCpin= 4;
int R1 = 100000;
int R2 = 10000;
float Vout;
float Vin;
void setup() {
pinMode(ADCpin,INPUT);
Serial.begin(9600); //open the serial port at 9600 bps
}
void loop() {
ADCvalue=analogRead(ADCpin);
Vout=ADCvalue* (3.3 / 4095.0);
Vin = ((R1+R2)*(Vout))/R2;
Serial.println(Vin);
}
```

3. Interfacing with ACS712

```
int ADCvalue;
int ADCpin= 4;
int R1 = 100000;
int R2 = 150000;
int Sensitivity = 185;
float Vout;
float Vin;
float Current;
float Offset = 0;
void setup() {
pinMode(ADCpin,INPUT);
Serial.begin(9600); //open the serial port at 9600 bps
}
void loop() {
ADCvalue=analogRead(ADCpin);
Vout=ADCvalue* (3300 / 4095.0);
Vin = ((R1 + R2) * (Vout)) / R2;
Current = (Vin - Offset) / Sensitivity;
Serial.println(Current);
}
```

4. Interfacing with IR Sensor

```
int ir_pin=4;
int led_pin=2;
void setup(){
 pinMode(ir_pin,INPUT);
 pinMode(led_pin,OUTPUT);
 Serial.begin(9600); //open the serial port at 9600 bps
}
void loop(){
 // print status of OUTPUT pin of IR sensor
 Serial.println(digitalRead(ir_pin));
 // check for presence of object
 if(digitalRead(ir_pin)==0){
 // if present turn on led
 digitalWrite(led_pin,HIGH);
 }
 else
 digitalWrite(led_pin,LOW);
}
```

5. Using IR sensor as Object Counter

```
#include<Wire.h>
#include<LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
int ir_pin=4;
int objectout;
int objectin=1;
int x=0;
void setup(){
pinMode(ir_pin,INPUT);
lcd.init(); // initializing the LCD
lcd.backlight(); // Enable or Turn On the backlight
lcd.print(" Object Counter ");
}
void loop(){
 // copy the status of OUTPUT pin of IR
 objectout=digitalRead(ir_pin);
 // wait until object is passwd through IR sensor
if(objectin==0 && objectout==1){
 x++; // increment count by 1 unit
lcd.setCursor(0,1);
lcd.print(x);
}
 // copy status of objectout to objectin
 objectin=objectout;
delay(10);
}
```