# Description:

Program to calculate the distance between current location and predetermined location using the GPS module in the Iomatic Development Board.

# Source Code:

// include the library code:

#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins

LiquidCrystal lcd(11,12,14,15,16,17);

char Input[200];

char Longitude[100];

char Latitude[100];

String serialResponse = "";

String arr[22];

float Lon=79.047274;

float Lat=21.121621;

// In Kilometers

float radius\_of\_earth = 6378.1;

int cnt=0;

void setup()

{

//SIM808 wakeup connected on pin 13 in IomaTic board

pinMode(13,OUTPUT);

//Initialize the SIM808 Module

digitalWrite(13, HIGH);

delay(1000);

//Sending wake up signal to SIM808 Module

digitalWrite(13, LOW);

delay(1000);

//Keeping SIM808 in active/wakeup state

digitalWrite(13, HIGH);

delay(3000);

//Initialize the LCD in 16x2 mode

lcd.begin(16, 2);

delay(100);

//Set cursor at first character/coloumn of first line/row

lcd.setCursor(0,0);

//Print the message as metioned cursor location

lcd.print(" IomaTic ");

//Set cursor at first character/coloumn of first line/row

lcd.setCursor(0,1);

//Print the message as metioned cursor location

lcd.print("GPS Location.........");

//Initialize a serial communication with baud rate 9600

Serial.begin(9600);

delay(500);

Serial.print("AT+CGNSPWR=1\r\n");

delay(2000);

// Serial.print("AT+CGNSTST=1\r\n");

// delay(2000);

Serial.print("AT+CGNSSEQ=\"RMC\"\r\n");

delay(2000);

}

void loop()

{

GetLocation();

delay(1000);

}

void GetLocation()

{

// OPERATOR (GPS Location)

Serial.print("AT+CGNSINF\r\n");

// find operator name between two double quotes

if (Serial.find("+CGNSINF: "))

{

serialResponse = Serial.readStringUntil('\r\n');

cnt=0;

char buf[sizeof(Input)];

serialResponse.toCharArray(buf, sizeof(buf));

char \*p = buf;

char \*str;

// delimiter is the semicolon

while ((str = strtok\_r(p, ",", &p)) != NULL)

{

Serial.println(str);

arr[cnt]=str;

cnt++;

}

lcd.clear();

//Set cursor at first character/coloumn of first line/row

lcd.setCursor(0,0);

lcd.print("Lon:");

//Print the message as metioned cursor location

lcd.print(arr[3]);

//Set cursor at first character/coloumn of first line/row

lcd.setCursor(0,1);

lcd.print("Lat:");

//Print the message as metioned cursor location

lcd.print(arr[4]);

delay(2000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print(distance\_in\_km(Lat, Lon, arr[3].toFloat(), arr[4].toFloat()));

delay(2000);

}

}

double distance\_in\_km(float start\_lat, float start\_long, float end\_lat, float end\_long)

{

// / 180 / PI converts degrees to radians

start\_lat/= 180 / PI;

start\_long/= 180 / PI;

end\_lat/= 180 / PI;

end\_long/= 180 / PI;

// haversine formula - based on implementation at http://www.movable-type.co.uk/scripts/latlong.html

float a = pow(sin((end\_lat-start\_lat)/2), 2) + cos(start\_lat) \* cos(end\_lat) \* pow(sin((end\_long-start\_long)/2), 2);

float answer = radius\_of\_earth \* 2 \* atan2(sqrt(a), sqrt(1-a));

return double(answer \* 1000);

}

# Libraries:

No additional libraries required.

# Functions:

*double distance\_in\_km(float start\_lat, float start\_long, float end\_lat, float end\_long)* *double distance\_in\_km(float start\_lat, float start\_long, float end\_lat, float end\_long)*

This function takes the input from the program, it takes two pairs of gps coordinates the first is the present located gps coordinates and the second is the predetermined gps coordinates. IT uses the haversine formula to calculate the distance between the two pairs of coordinates and returns the answer.