Riprendiamo ora lo sketch del sensore ad ultrasuoni e modifichiamolo in questo modo per tenere conto della temperatura rilevata

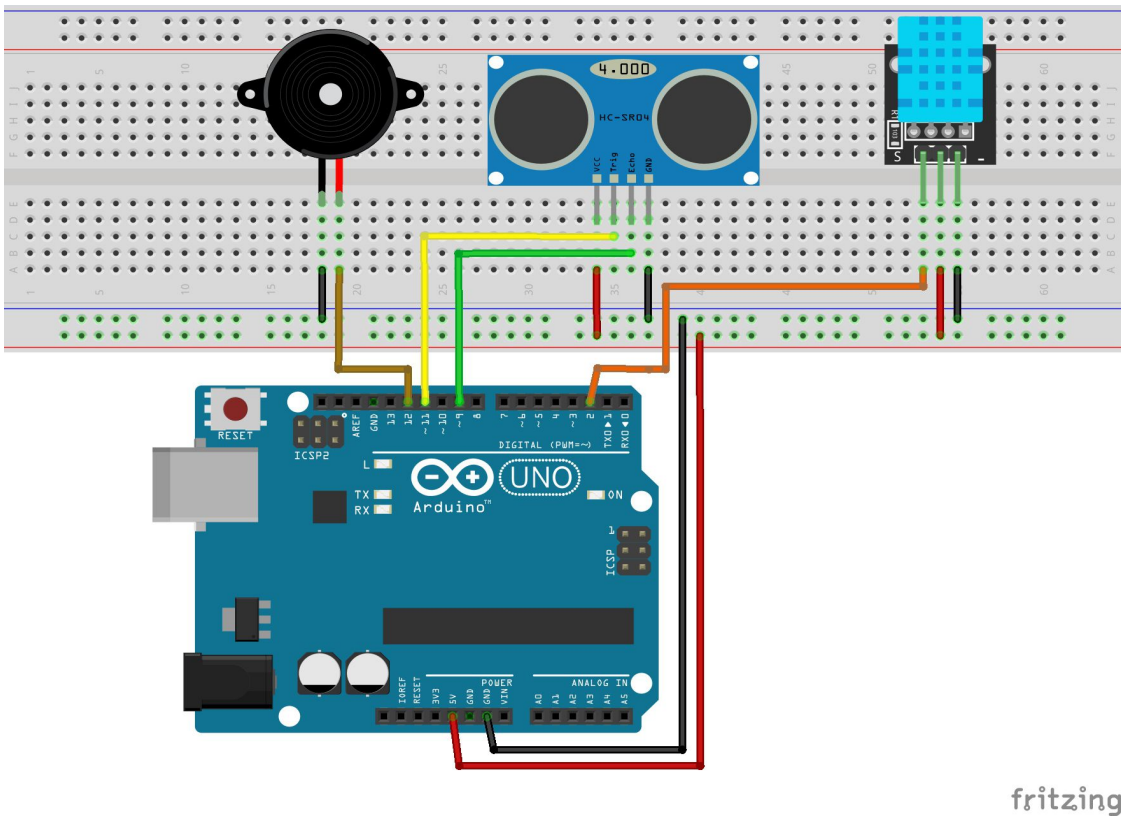
```
/*  sensore ultrasuoni HC-SR04
 *  programma di test funzionamento
 *  distanza= (tempo segnale echo high/2) *velocità del suono ((331,45 + (0,62 * T)))
 *  echo collegato al pin 9
 *  trig collegato al pin 11
 *  Aggiunto sensore DHT11 - temperatura ed umidità
 *  Signal collegato al pin 2
 *  Per usare questo sensore in modo semplice dobbiamo includere
 *  nel nostro programma una libreria con il codice preparato da altri per questo scopo
 *  il nome di questa libreria è dht.h
*/
#include <dht.h>
// ora creiamo l'oggetto DHT cioè un oggetto che eredita tutte le
// proprietà (metodi e funzioni) scritte nella libreria
dht DHT;

//definiamo i pin a cui siamo collegati
const int pindht = 2;
const int trigPin = 11;
const int echoPin = 9;

//===== spazio variabili ========
// variabili usate per durata impulso echo e calcolo centimetri
float durata, cm;
// definisco una variabile per memorizzare il passare del tempo
unsigned long tempo;
//definisco l'intervallo minimo tra due letture del sensore di temperatura
int intervallo = 10000;
//variabile temperatura e umidità
float miaTemp = 0.0;
float miaUmid = 0.0;

void setup() {
    Serial.begin(9600);
    pinMode(trigPin,OUTPUT);
    pinMode(echoPin,INPUT);
    pinMode(pindht, INPUT);
    // tempo è uguale ad adesso!
    tempo = millis();
    Serial.println("HT-SR04 + DHT11 \t Inizializzazione");
    delay(2000);
}

void loop() {
    //controllo se è ora di interrogare il sensore temperatura
```

```
46    if (( millis() - tempo )>= intervallo) {
47         int chk = DHT.read11(pindht);
48         switch (chk){
49         case DHTLIB_OK:
50            break;
51         case DHTLIB_ERROR_CHECKSUM:
52            Serial.print("Checksum error,\t");
53            break;
54         case DHTLIB_ERROR_TIMEOUT:
55            Serial.print("Time out error,\t");
56            break;
57         default:
58            Serial.print("Unknown error,\t");
59            break;
60        }
61        miaTemp = DHT.temperature;
62        miaUmid = DHT.humidity;
63        Serial.println("");
64        Serial.print("umidità = ");
65        Serial.print(miaUmid);
66        Serial.print("%\t");
67        Serial.print("temperatura = ");
68        Serial.print(miaTemp);
69        Serial.println(" °C");
70        Serial.println("");
71        // aggiorno il tempo con adesso!
72        tempo = millis();
73    }
74    //richiesta invio impulsi al sensore
75    digitalWrite(trigPin,LOW);
76    delayMicroseconds(2);
77    digitalWrite(trigPin,HIGH);
78    delayMicroseconds(10);
79    digitalWrite(trigPin,LOW);
80
81    //restituisce il tempo di echo
82    durata=pulseIn(echoPin,HIGH);
83
84    //trasformo in cm
85    cm = durata / 20000.0 *(331.45 + (0.62 * miaTemp));
86    //scrivo sul monitor seriale
87    Serial.print("Ostacolo rilevato a: ");
88    Serial.print(cm);
89    Serial.println(" cm di distanza");
90    delay(1000);
91 }
```

Nella prima riga dello sketch c'è il comando: ***#include <libreria.h>*** che serve appunto per includere nel programma che stiamo scrivendo le funzioni presenti nella libreria indicata, cos'è una libreria? Si tratta di una raccolta di funzioni, scritte per rendere più semplice la programmazione o l'interfaccia di oggetti esterni. In pratica in questo caso non ci interessa come deve essere fatta la richiesta di lettura della temperatura e la decodifica delle informazioni che il sensore dht11 ci passa, noi scriviamo semplicemente: ***DHT.read11(pin_signal)***, e otterremo direttamente la temperatura e l'umidità rilevate nelle due variabili ***DHT.temperature*** e ***DHT.humidity***, pronte per l'utilizzo. L'utilizzo delle librerie non è obbligatorio, potremmo metterci a studiare il datasheet del componente e creare per conto nostro tutto il software relativo al suo funzionamento ma così è molto più semplice.
Prima di continuare è importante sapere che non tutte le librerie sono automaticamente disponibili nell'ide di Arduino. Le librerie che non sono presenti vanno scaricate da internet in formato ".ZIP" e poi installate usando il menù Sketch-> #include libreria ->Aggiungi libreria da file .zip. A questo punto si apre una

finestra dove indichiamo dove si trova sul nostro computer la libreria da installare, la selezioniamo e clicchiamo su apri per completare l'importazione. Da questo momento le funzioni della libreria saranno disponibili per i nostri sketch.

Le raccomandazioni per l'uso del sensore temperatura dicono di attendere almeno 2 secondi tra una lettura e l'altra, se usiamo un tempo più breve di campionamento, rischiamo di rileggere il vecchio valore che resta memorizzato fino alla nuova lettura. Per la nostra applicazione non abbiamo esigenze particolari, una lettura ogni 10 secondi sarà più che sufficiente, ovviamente non possiamo usare il comando delay(10000), altrimenti tutto il programma rimarrebbe in attesa, quindi definiamo una costante con il tempo di attesa e con l'istruzione millis() che ci da il tempo trascorso in millisecondi dall'ultimo reset/accensione, controlliamo se è passato tempo a sufficienza per eseguire una nuova lettura, nel frattempo continuiamo con le letture del sonar ogni secondo.
Se ora colleghiamo un buzzer attivo tra massa e pin 12 e quando il bersaglio è a meno di un metro comincia a suonare ad intermittenza e facciamo aumentare la frequenza del suono ogni 10 cm di avvicinamento.... abbiamo simulato un sensore di parcheggio
Ovviamente con tutti i limiti dovuti alla tipologia dei componenti (velocità di campionamento, precisione...)
Ecco la breadboard finale



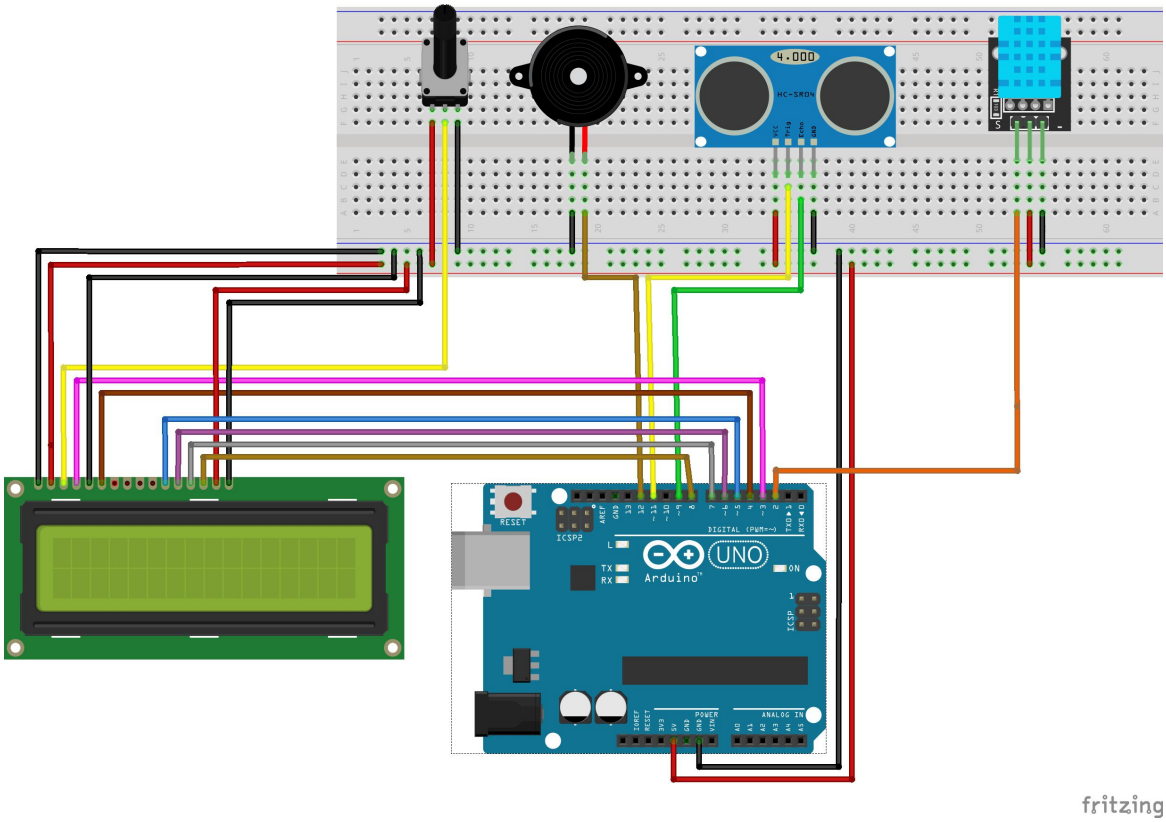
e qui lo sketch di prima con l'aggiunta di una funzione per far suonare il nostro buzzer con frequenze diversa e una serie di istruzioni if nidificate per valutare quale frequenza applicare
alcune delle funzioni già viste prima le ho compresse per non allungare troppo il listato.

```
/*  sensore ultrasuoni HC-SR04
#include <dht.h>
// ora creiamo l'oggetto DHT cioè un oggetto che eredita tutte le
// proprietà (metodi e funzioni) scritte nella libreria
dht DHT;

//definiamo i pin a cui siamo collegati
const int pindht = 2;
const int trigPin = 11;
const int echoPin = 9;
const int buzzer = 12;
//===== spazio variabili ========
int buzzerOn = 0;           // tempo di suono buzzer
int buzzerOff = 0;          // tempo di pausa buzzer

float durata;               // variabile per durata impulso echo
float cm = 200.0;           // variabile per calcolo centimetri
unsigned long tempoD;       // tempo e intervallo per il sensore distanza
int intervalloD = 1000;

unsigned long tempoT;       // definisco una variabile per memorizzare il passare del tempo
int intervalloT = 10000;    //definisco l'intervallo minimo tra due letture del sensore di temperatura
float miaTemp = 0.0;        //variabili temperatura e umidità
float miaUmid = 0.0;

// Creo una funzione che mi permette di far suonare con frequenze diverse il buzzer
// suonaBuzzer(pin, tempo_on, tempo_off)
void suonaBuzzer (int myPin, int myTOn, int myTOff) {
    digitalWrite(myPin,HIGH);
    delay(myTOn);
    digitalWrite(myPin, LOW);
    delay(myTOff);
}

void setup() {
    Serial.begin(9600);
    pinMode(trigPin,OUTPUT);
    pinMode(echoPin,INPUT);
    pinMode(pindht, INPUT);
    pinMode(buzzer, OUTPUT);

    // tempo è uguale ad adesso!
    tempoT = millis();
    tempoD = millis();
    Serial.println("HT-SR04 + DHT11 \t Inizializzazione");
```

```
    delay(2000);
}

void loop() {
    //controllo se è ora di interrogare il sensore temperatura
    if (( millis() - tempoT )>= intervalloT) {
    //controllo se è ora di interrogare il sensore distanza
    if ((millis() - tempoD) >= intervalloD) {
        //richiesta invio impulsi al sensore
        digitalWrite(trigPin,LOW);
        delayMicroseconds(2);
        digitalWrite(trigPin,HIGH);
        delayMicroseconds(10);
        digitalWrite(trigPin,LOW);

        //restituisce il tempo di echo
        durata=pulseIn(echoPin,HIGH);

        //trasformo in cm
        cm = durata / 20000.0 *(331.45 + (0.62 * miaTemp));
        //scrivo sul monitor seriale
        Serial.print("Ostacolo rilevato a: ");
        Serial.print(cm);
        Serial.println(" cm di distanza");
         // aggiorno il tempo con adesso!
        tempoD = millis();
    }
    if (cm < 100 && cm >90) {
        buzzerOn = 200;
        buzzerOff = 200;
    }
    else if (cm < 91 && cm > 80) {
        buzzerOn = 180;
        buzzerOff = 180;
    }
    else if (cm < 81 && cm > 70) {
    else if (cm < 71 && cm > 60) {
    else if (cm < 61 && cm > 50) {
    else if (cm < 51 && cm > 40) {
    else if (cm < 41 && cm > 30) {
    else if (cm < 31 && cm > 20) {
    else if (cm < 21 && cm > 10) {
        buzzerOn = 10;
        buzzerOff = 10;
    }
    else if (cm < 11 ) {
        buzzerOn = 5;
        buzzerOff = 5;
    }
    if (cm < 101) {
        suonaBuzzer (buzzer, buzzerOn, buzzerOff);
    }
}
```

Ora giusto per completare l'opera, aggiungiamo un display lcd da 2 righe al nostro progettino, così non dovremo usare il monitor seriale per visualizzare le informazioni.
I display di questo tipo prevedono numerose connessioni per poter funzionare. Possiamo vedere sul datasheet “lcd1602A.pdf” l'elenco completo della piedinatura, dovremo collegare i pin da 1 a 6 e da 11 a 16, sei dei quali andranno ad altrettanti pin di Arduino.

Questa è la nostra breadboard



e questo lo sketch completo che anche in questo caso usa una libreria esterna per semplificarne la scrittura, si tratta della libreria “LiquidCrystal.h” specifica per la gestione di questi lcd, ho inserito come commento alla riga di creazione dell'oggetto lcd l'elenco dei pin utilizzati, con il loro nome sull'lcd. Notare che i pin che sono stati assegnati all'lcd non hanno bisogno di altre dichiarazioni nel setup, la loro gestione (input, output...) è eseguita totalmente dalla libreria:

```
/*  sensore ultrasuoni HC-SR04
//aggiungiamo le librerie che ci servono
#include <dht.h>
#include <LiquidCrystal.h>

//creiamo l'oggetto lcd e gli assegnamo i pin collegati
//                BS E  D4 D5 D6 D7
LiquidCrystal lcd(3, 4, 5, 6, 7, 8);

// ora creiamo l'oggetto DHT cioè un oggetto che eredita tutte le
// proprietà (metodi e funzioni) scritte nella libreria
dht DHT;

//definiamo i pin a cui siamo collegati
const int pindht = 2;
const int trigPin = 11;
const int echoPin = 9;
const int buzzer = 12;
//===== spazio variabili ========
int buzzerOn = 0;           // tempo di suono buzzer
int buzzerOff = 0;          // tempo di pausa buzzer

float durata;              // variabile per durata impulso echo
float cm = 200.0;          // variabile per calcolo centimetri
unsigned long tempoD;      // tempo e intervallo per il sensore distanza
int intervalloD = 1000;

unsigned long tempoT;      // definisco una variabile per memorizzare il passare del tempo
int intervalloT = 10000;   //definisco l'intervallo minimo tra due letture del sensore di temperatura
float miaTemp = 0.0;       //variabili temperatura e umidità
float miaUmid = 0.0;

// Creo una funzione che mi permette di far suonare con frequenze diverse il buzzer
// suonaBuzzer(pin, tempo_on, tempo_off)
void suonaBuzzer (int myPin, int myTOn, int myTOff) {

void checkTemp() {
   if (( millis() - tempoT )>= intervalloT) {
      int chk = DHT.read11(pindht);

      miaTemp = DHT.temperature;
      miaUmid = DHT.humidity;
      //scrivo i valori sulla prima riga dell'lcd
      lcd.setCursor(0, 0);
      lcd.print("T:    C  U:    %");
      lcd.setCursor(2, 0);
      lcd.print(miaTemp,1);
      lcd.setCursor(11, 0);
      lcd.print(miaUmid,1);

      tempoT = millis();                        //aggiorno il tempo con adesso!
   }
}
```

```
70  void checkDist() {
71     if ((millis() - tempoD) >= intervalloD) {
72        digitalWrite(trigPin,LOW);                 //richiesta invio impulsi al sensore
73        delayMicroseconds(2);
74        digitalWrite(trigPin,HIGH);
75        delayMicroseconds(10);
76        digitalWrite(trigPin,LOW);
77        durata=pulseIn(echoPin,HIGH);              //restituisce il tempo di echo
78
79        cm = durata / 20000.0 *(331.45 + (0.62 * miaTemp));    //trasformo in cm
80
81        lcd.setCursor(0, 1);                    //scrivo il valore sulla seconda riga dell'lcd
82        lcd.print("Dist.:       cm");
83        lcd.setCursor(6, 1);
84        lcd.print(cm);
85
86        tempoD = millis();                       // aggiorno il tempo con adesso!
87     }
88  }
89  void checkBuzzer() {
134
135 void setup() {
136     //inializzazione lcd
137     lcd.begin(16, 2);
138     pinMode(trigPin,OUTPUT);
139     pinMode(echoPin,INPUT);
140     pinMode(pindht, INPUT);
141     pinMode(buzzer, OUTPUT);
142
143     // tempo è uguale ad adesso!
144     tempoT = millis();
145     tempoD = millis();
146     delay(2000);
147 }
148
149 void loop() {
150     checkTemp();              //controllo se è ora di interrogare il sensore temperatura
151     checkDist();              //controllo se è ora di interrogare il sensore distanza
152     checkBuzzer();            //controllo la frequenza del buzzer
153 }
```