## Arduino-1

```
#include <LiquidCrystal.h>

#define ADC_VREF_TYPE ((0<<REFS1) | (0<<REFS0) | (0<<ADLAR))
// for temp sensor

float stepADC = 0.0048828125;

LiquidCrystal lcd1(5, 4, 3, 13, 2, 6);
LiquidCrystal lcd2(12, 11, 10, 9, 8, 7);

char sNext[119];

// declared globally because in the case of the calculations in screen 2
// the memory is already written and another initialization should have been done
// with several variables, etc ..
static char *aValues[2];
static char *token;
static char *aValues1[9];
static char *token1;
static char *aValues2[15];
static char *token2;

int counter = 0;
bool screen = 0;
bool ButtonStatus = 0;
bool PreviousButtonStatus = 0;
//---------------------

unsigned int readADC(unsigned int adc_input)
```

```
{
  ADMUX = adc_input | ADC_VREF_TYPE;
  // ADMUX controls reference voltage
// delay required to set ADC for voltage
   // analog input
  delayMicroseconds(10);

// start conversion
  ADCSRA |= (1<<ADSC);
// Wait for conversion to complete
  while ((ADCSRA & (1<<ADIF))==0){}
  ADCSRA |= (1<<ADIF);

// return result on 16 bit
  return ADCW;
}

// code recognition / segmentation sequence
void segmentSequence()
{
// segment the main sequence
  token = strtok(sNext, "/");
  static int increment = 0;
  while (token != NULL)
  {
    aValues[increment++] = token;

    token = strtok(NULL, "/");
  }
```

```
// segmentation sequence 1
  token1 = strtok(aValues[0], ",");
  static int increment1 = 0;
  while (token1 != NULL)
  {
    aValues1[increment1++] = token1;

    token1 = strtok(NULL, ",");
  }

// segmentation sequence 2
  token2 = strtok(aValues[1], ",");
  static int increment2 = 0;
  while (token2 != NULL)
  {
    aValues2[increment2++] = token2;

    token2 = strtok(NULL, ",");
  }

// next segmental closing ----------
}

void ScreenOne()
{
// display speed
// atof- converts the string argument str to a floating-point number
  static float Speed = 0;
  if (strcmp(aValues1[0], "$GPVTG") == 0)
  {
```

```
    Speed = atof(aValues1[7]);
  }
// cursor setting: column 0, line 1
  lcd1.setCursor(0, 1);
  lcd1.print("Speed: ");
  lcd1.print(Speed);
  lcd1.print("km/h");

// time display
  static long int temp[4];
  static int iSatellites = 0;
  if (strcmp(aValues2[0], "$GPGGA") == 0)
  {
// storage string
          temp[0] = atof(aValues2[1]);
// second storage
          temp[3] = temp[0] % 100;
          temp[0] = temp[0] / 100;
// minute storage
          temp[2] = temp[0] % 100;
          temp[0] = temp[0] / 100;
// store hours
          temp[1] = temp[0];

    iSatellites = atoi(aValues2[7]);
  }
// atoi- converts the string argument str to an integer
// cursor setting: column 0, line 0
  lcd1.setCursor(0, 0);
  lcd1.print("time: ");
```

```
  lcd1.print(temp[1]);
  lcd1.print(":");
  lcd1.print(temp[2]);
  lcd1.print(":");
  lcd1.print(temp[3]);


// cursor setting: column 0, line 3
  lcd2.setCursor(0,0);
  lcd2.print("Nr. sat: ");
  lcd2.print(iSatellites);

// read thermometer from analog pin "0"
  unsigned int sensorValue = readADC(0);
 // calculate the temperature according to the values in the thermometer datasheet
  float fTemperature = (stepADC*sensorValue-0.5)*100;

// cursor setting: column 0, line 3
  lcd2.setCursor(0, 1);
  lcd2.print("Temp: ");
  lcd2.print(fTemperature, 2);
  lcd2.print(" C");
// end processing screen one --------
}

// display / select data on the second screen
void ScreenTwo()
{
// segment Sequence ();
   // calculate, display latitude
```

```
  static float fLatitude = 0;
  static float fLongitude = 0;
  static float fAltitude = 0;
  if (strcmp(aValues2[0], "$GPGGA") == 0)
  {
    fLatitude = atof(aValues2[2]);
    fLongitude = atof(aValues2[4]);
    fAltitude = atof(aValues2[9]) - atof(aValues2[11]);
  }
// display data
   // cursor setting: column 0, line 1
  lcd1.setCursor(0, 1);
  lcd1.print("Lat: ");
  lcd1.print((int)fLatitude/100+((int)(fLatitude)%100+(fLatitude-(int)(fLatitude)))/60);
  lcd1.print(aValues2[3]);
 // cursor setting: column 0, line 2
  lcd2.setCursor(0, 0);
  lcd2.print("Long: ");
  lcd2.print((int)fLongitude/100+((int)(fLongitude)%100+(fLongitude-
(int)(fLongitude)))/60);
  lcd2.print(aValues2[5]);
// cursor setting: column 0, line 3
  lcd2.setCursor(0, 1);
  lcd2.print("Alt: ");
  lcd2.print(fAltitude);
  lcd2.print("m");
// end screen processing two --------
}

//---------------------
// -------SETUP--------
```

```
//---------------------
void setup()
{
  Serial.begin(9600);
  lcd1.begin(2,16);
  lcd2.begin(2,16);
}
//---------------------
// -------LOOP---------
//---------------------
void loop()
{
  if(counter < 1)
  {
    Serial.readBytes(sNext, 119);
    Serial.println(sNext);
    segmentSequence();
    counter++;
  }
      ScreenOne();
    delay(1000);
    lcd1.clear();
            lcd2.clear();
 ScreenTwo();
    delay(1000);
    lcd1.clear();
            lcd2.clear();
}
```

## Arduino-2

```
//Tushar Maheshwari

#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int alarm = 6;char text1[] = "$GPVTG,054.7,T,034.4,M,005.5,N,010.2,K";
// input format from GPS device of the ground speed
char text2[] = "$GPGGA,134658.00,5106.9792,N,11402.3003,W,2,09,1.0,1048.47,M,-16.27,M,08,AAAA*60";
void setup()
{
   pinMode(alarm, OUTPUT);
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.

  Serial.begin(9600);
  Serial.begin(9600);
}

void loop()
{

 int heartRate;
  heartRate = random(40, 150);

  Serial.println(heartRate);
```

```
  lcd.print("Heart Rate: ");
  if(heartRate > 100 | heartRate < 60)
  {
    lcd.print(heartRate);

          digitalWrite(alarm, HIGH);
    delay(500);
    lcd.clear();
  }
  else
  {
    lcd.print(heartRate);

    digitalWrite(alarm, LOW);
    delay(500);
    lcd.clear();
  }
   delay(200);
  Serial.write(text1);
  Serial.write("/");
  delay(500);
  Serial.write(text2);
  delay(500);
}
```