Arduino bluetooth game

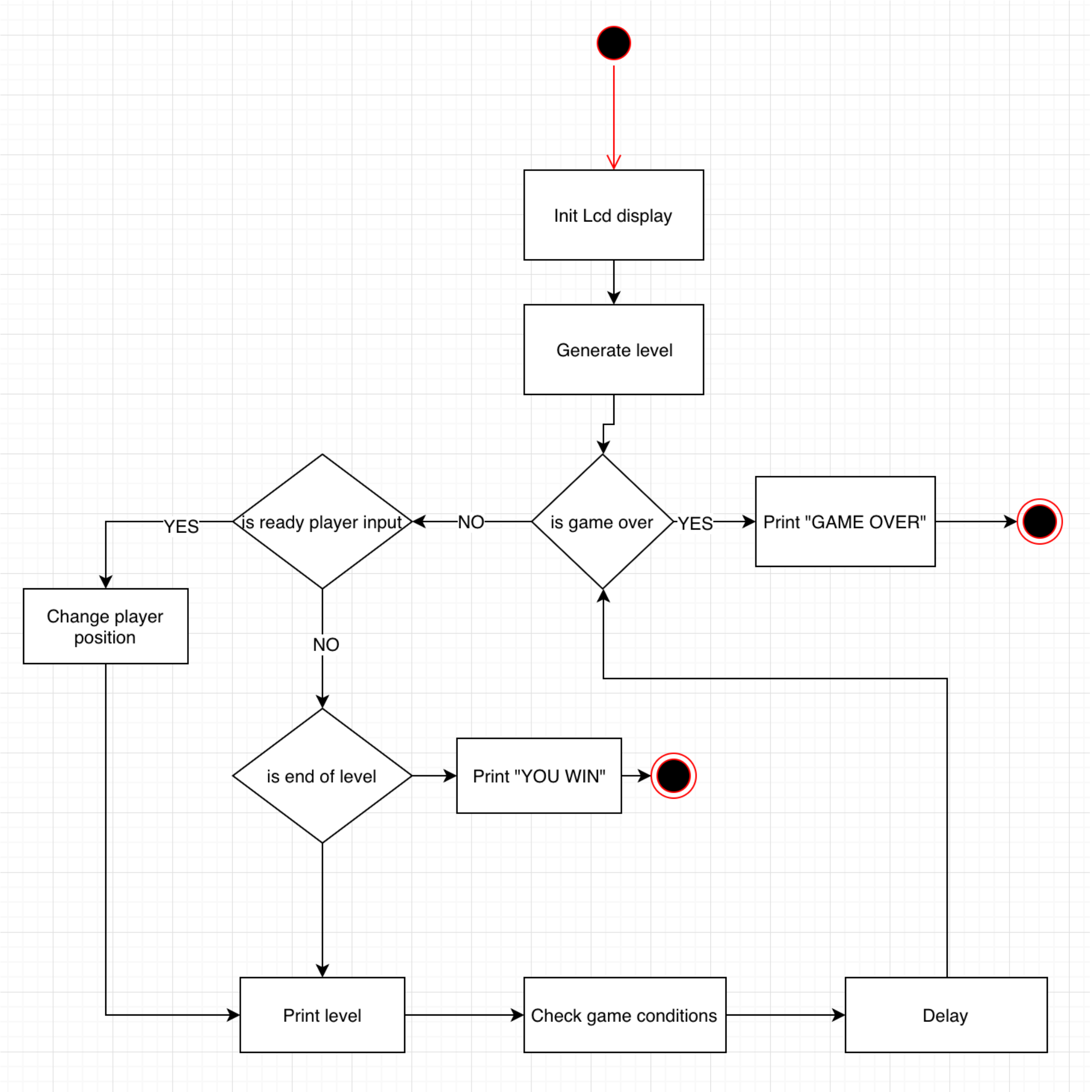
AUTORS: Jakub Tomczyk, Ihor Tkachuk, Oleksandr Samoiliuk, Roman Kostyshyn

Functional requirements of prototype

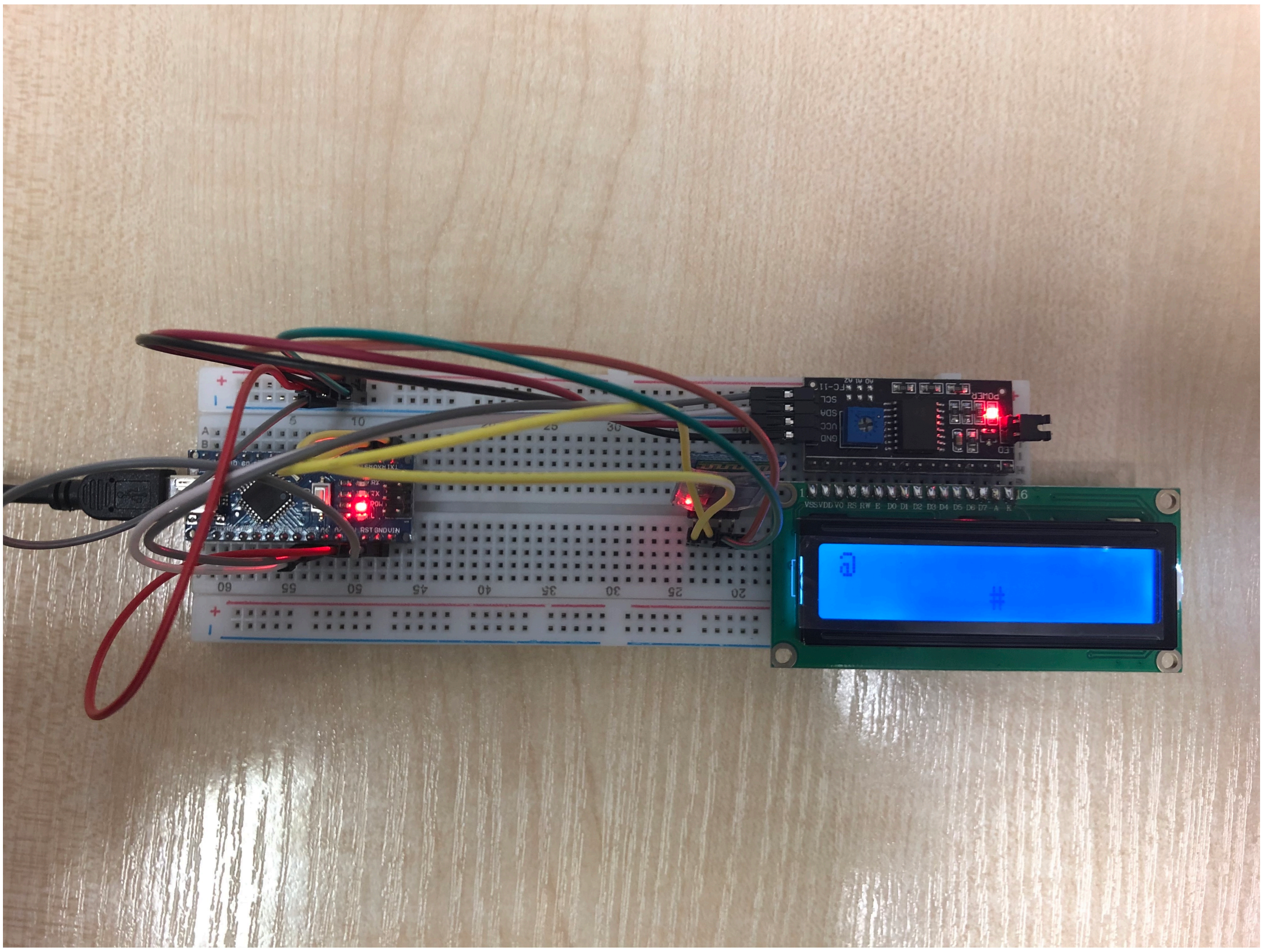
Project aims to be like tetris rally game.
Project's display shows two lanes, and player marked as @.
On lanes there are random obstacles shown as #.

Player should avoid all obstacles by changing lanes.
Player controls game by external PC client, which is simple console NodeJs application, which connects to Bluetooth serial port and sends '0' and '1' codes on key up/down clicks.

UML



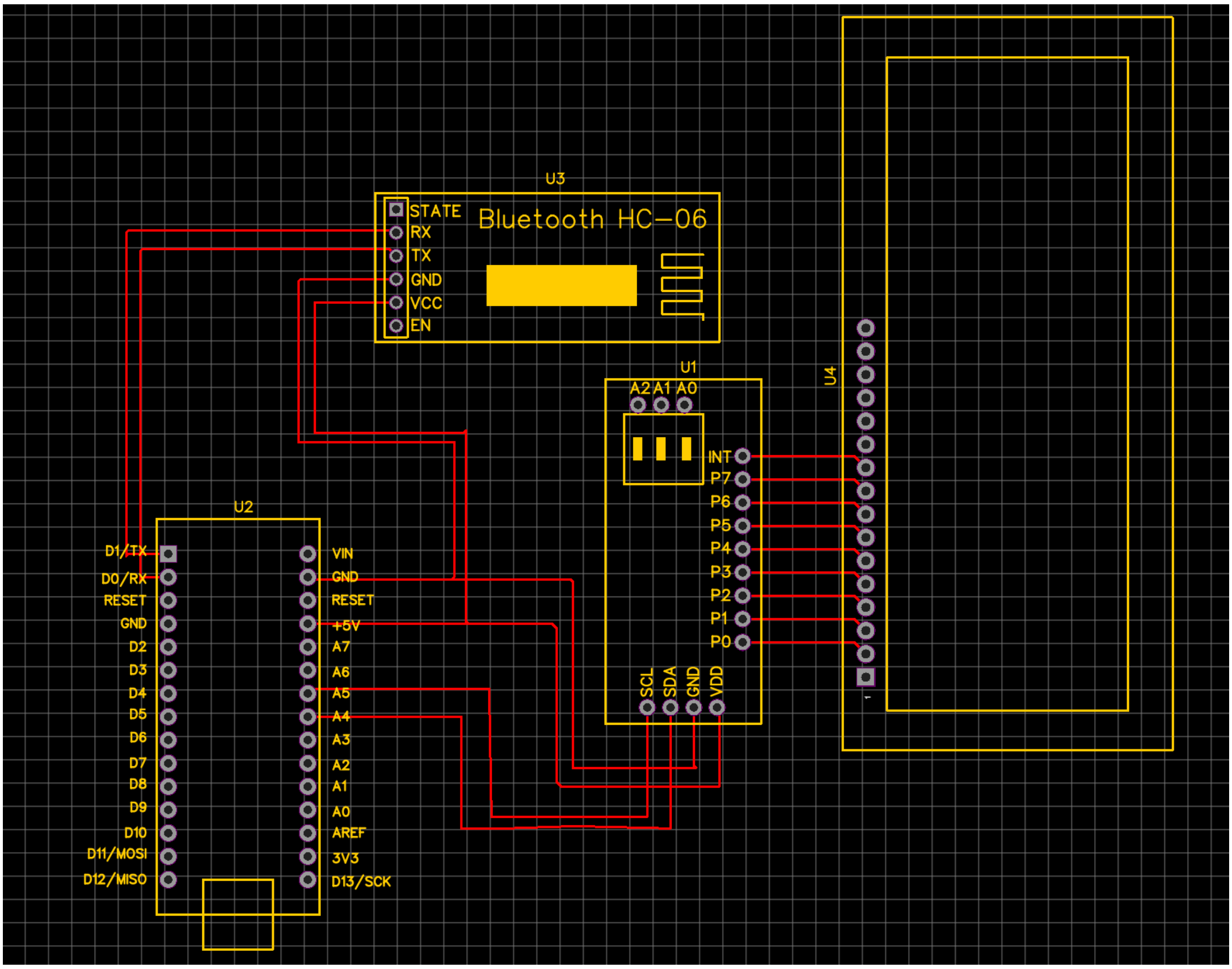
Prototype photo



Used sensors and devices

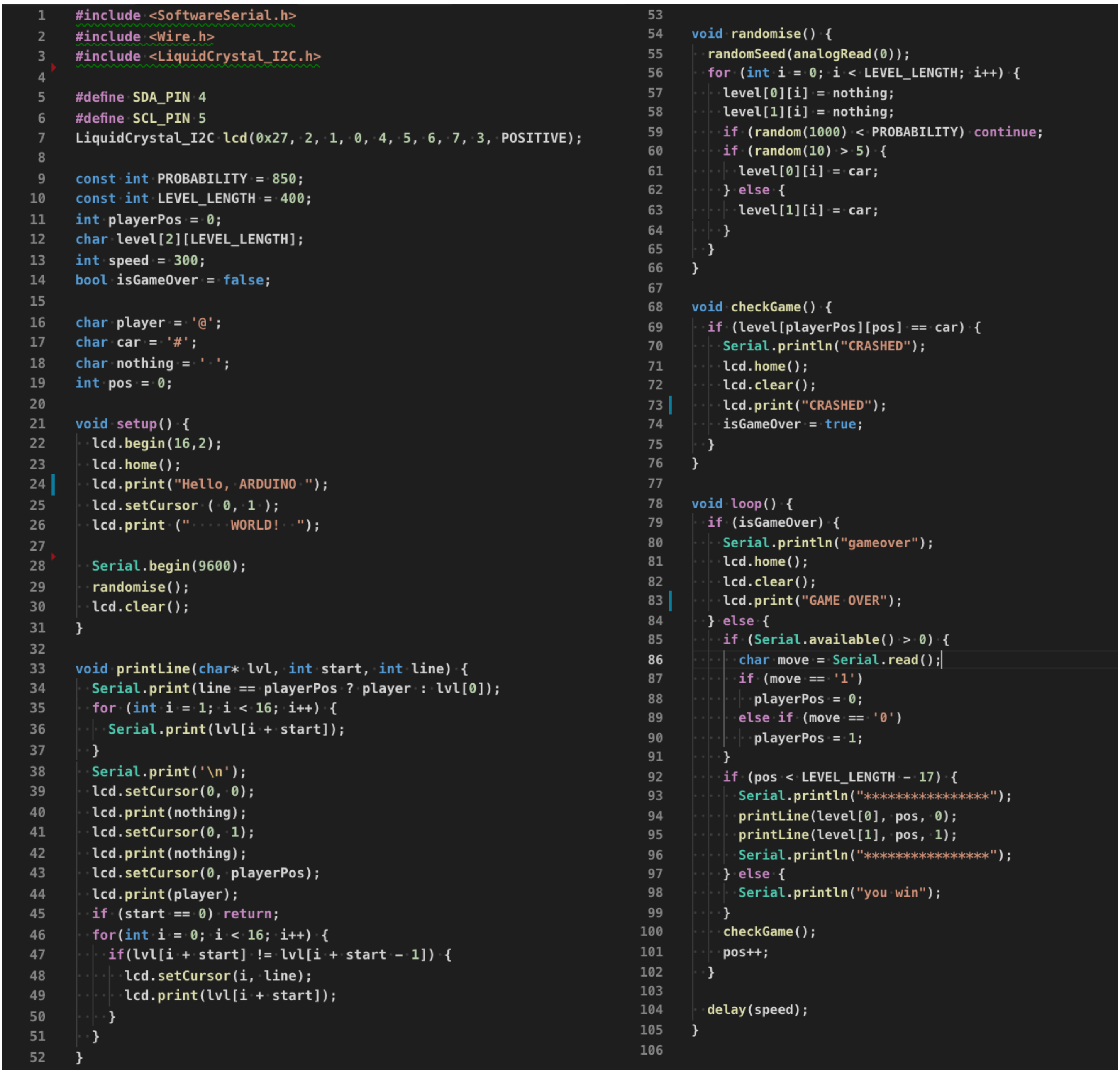
- **Bluetooth Slave HC06**
used to connect arduino with PC client. Connected to Arduino via RX, TX channels, so acts as Serial Input/Output.
- **Display LCD 1602 2x16 HD44780**
used to show game gui
- **I2C converter FC-11**
used to simplify arduino connection to LCD display

Project schema



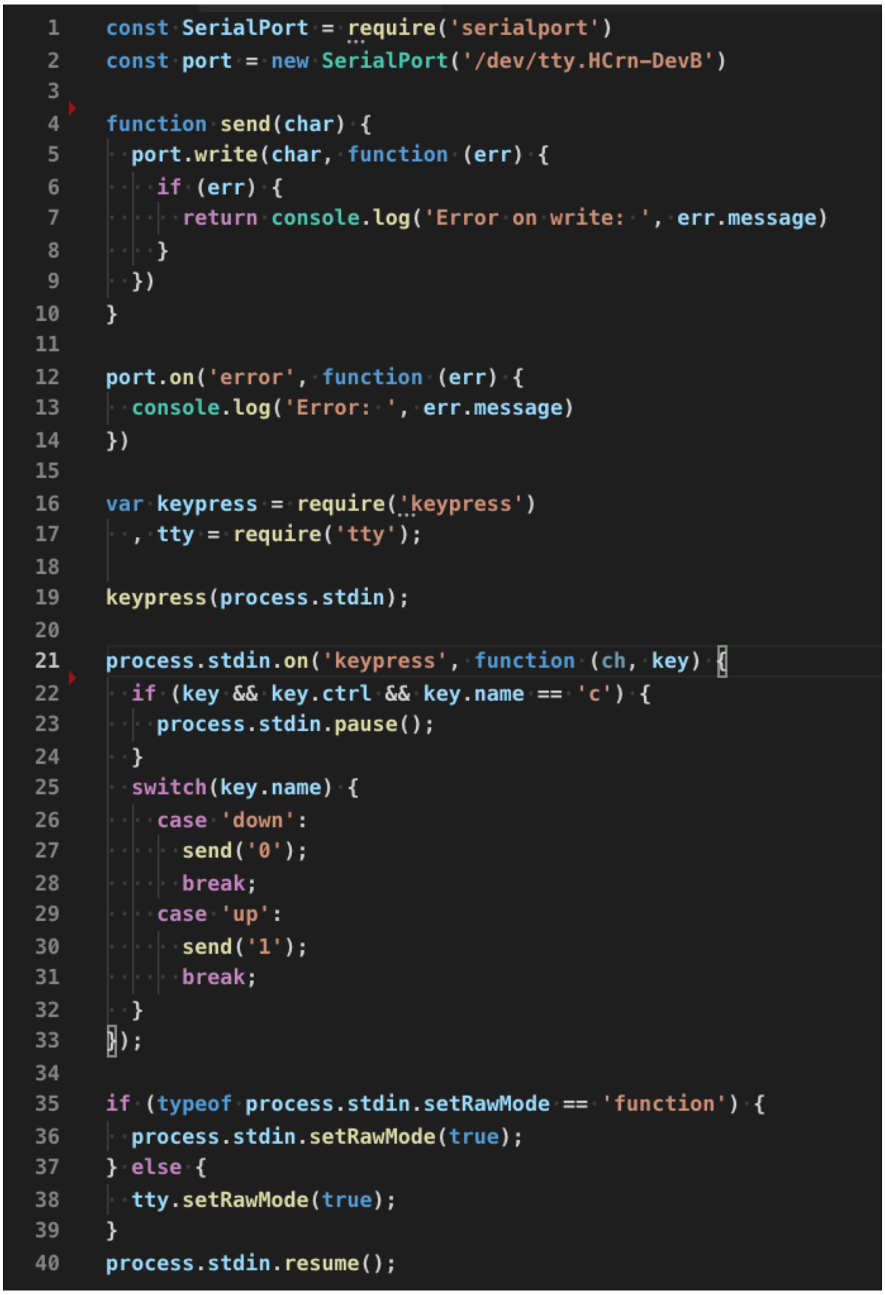
Program code

Arduino code



```
#include <SoftwareSerial.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define SDA_PIN 4
#define SCL_PIN 5
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);

const int PROBABILITY = 850;
const int LEVEL_LENGTH = 400;
int playerPos = 0;
char level[2][LEVEL_LENGTH];
int speed = 300;
bool isGameOver = false;

char player = '@';
char car = '#';
char nothing = ' ';
int pos = 0;

void setup() {
  lcd.begin(16,2);
  lcd.home();
  lcd.print("Hello, ARDUINO ");
  lcd.setCursor ( 0, 1 );
  lcd.print ("    WORLD!  ");

  Serial.begin(9600);
  randomise();
  lcd.clear();
}

void printLine(char* lvl, int start, int line) {
  Serial.print(line == playerPos ? player : lvl[0]);
  for (int i = 1; i < 16; i++) {
    Serial.print(lvl[i + start]);
  }
  Serial.print('\n');
  lcd.setCursor(0, 0);
  lcd.print(nothing);
  lcd.setCursor(0, 1);
  lcd.print(nothing);
  lcd.setCursor(0, playerPos);
  lcd.print(player);
  if (start == 0) return;
  for(int i = 0; i < 16; i++) {
    if(lvl[i + start] != lvl[i + start - 1]) {
      lcd.setCursor(i, line);
      lcd.print(lvl[i + start]);
    }
  }
}

void randomise() {
  randomSeed(analogRead(0));
  for (int i = 0; i < LEVEL_LENGTH; i++) {
    level[0][i] = nothing;
    level[1][i] = nothing;
    if (random(1000) < PROBABILITY) continue;
    if (random(10) > 5) {
      level[0][i] = car;
    } else {
      level[1][i] = car;
    }
  }
}

void checkGame() {
  if (level[playerPos][pos] == car) {
    Serial.println("CRASHED");
    lcd.home();
    lcd.clear();
    lcd.print("CRASHED");
    isGameOver = true;
  }
}

void loop() {
  if (isGameOver) {
    Serial.println("gameover");
    lcd.home();
    lcd.clear();
    lcd.print("GAME OVER");
  } else {
    if (Serial.available() > 0) {
      char move = Serial.read();
      if (move == '1')
        playerPos = 0;
      else if (move == '0')
        playerPos = 1;
    }
    if (pos < LEVEL_LENGTH - 17) {
      Serial.println("****************");
      printLine(level[0], pos, 0);
      printLine(level[1], pos, 1);
      Serial.println("****************");
    } else {
      Serial.println("you win");
    }
    checkGame();
    pos++;
  }

  delay(speed);
}
```

NodeJS code



```javascript
const SerialPort = require('serialport')
const port = new SerialPort('/dev/tty.HCrn-DevB')

function send(char) {
  port.write(char, function (err) {
    if (err) {
      return console.log('Error on write: ', err.message)
    }
  })
}

port.on('error', function (err) {
  console.log('Error: ', err.message)
})

var keypress = require('keypress')
  , tty = require('tty');

keypress(process.stdin);

process.stdin.on('keypress', function (ch, key) {
  if (key && key.ctrl && key.name == 'c') {
    process.stdin.pause();
  }
  switch(key.name) {
    case 'down':
      send('0');
      break;
    case 'up':
      send('1');
      break;
  }
});

if (typeof process.stdin.setRawMode == 'function') {
  process.stdin.setRawMode(true);
} else {
  tty.setRawMode(true);
}
process.stdin.resume();
```