```
/*
-------------------------------------------------------------------------------------------------------------------------------
   Ima gomila stimunga koji treba tek da se klasifikuje i upakuje kako treba- bude li potrebe. :P
   Definisane vrednosti za note su cesto tweakovane da bi se postigao neki efekat u nekim deonicama- ima
   dosta odstupanja od zadatih vrednosti. Osnova za navedene zadate vrednosti je u tabeli ispod:

    vr. note I vr. u f-ji
    ---------------------
      cela   I    4
      pola   I    2
    cetvrtinaI    1
     osmina  I   0.5
      1/16   I   0.25
      1/32   I   0.125
      1/64   I   0.0625

-------------------------------------------------------------------------------------------------------------------------------
*/


int coila1 = 4;
int coila2 = 5;
int coilb1 = 6;
int coilb2 = 7;
int sensor = 11;

int LEDb = 13;
int LEDc = 10;
float tempo = 99.00;                        // Tempo, bpm.
float t = (60.00/tempo)/8;                  // Trajanje cele note u sekundama/8
int o = 1; // Za podizanje oktave.
float H0 = o*30.87;
float C1 = o*32.70;
float C1s = o*34.65;
float D1 = o*36.71;
float D1s = o*38.89;
float E1 = o*41.20;
float F1 = o*43.65;
float F1s = o*46.25;
float G1 = o*49.00;
float G1s = o*51.91;
float lA1 = o*55.00;                      // Zbog postojećih promenljivih za analogne pinove.
float A1s = o*58.27;
float H1 = o*61.74;
float C2 = o*65.41;
float C2s = o*69.30;
float D2 = o*73.42;
float D2s = o*77.80;
float E2 = o*82.41;
float F2 = o*87.31;
float F2s = o*92.50;
float G2 = o*98.00;
float G2s = o*103.83;
float lA2 = o*110.00;                      // Zbog postojećih promenljivih za analogne pinove.
float A2s = o*116.54;
float H2 = o*123.47;
float C3 = o*130.81;
float C3s = o*138.59;
float D3 = o*146.83;
float D3s = o*155.56;
float E3 = o*164.81;
float F3 = o*179.61;
float F3s = o*185.00;
float G3 = o*199.9999;
float G3s = o*207.65;
float lA3 = o*220.00;                      // Zbog postojećih promenljivih za analogne pinove.
float A3s = o*233.08;
float H3 = o*246.94;
float C4 = o*261.63;


void setup() {

pinMode(coila1, OUTPUT);
pinMode(coila2, OUTPUT);
pinMode(coilb1, OUTPUT);
pinMode(coilb2, OUTPUT);
pinMode(sensor, INPUT);
pinMode(LEDb, OUTPUT);
pinMode(LEDc, OUTPUT);


Serial.begin(115200);

}

void loop() {

 float cela = 4;
 float pola = 2;
 float cetvrtina = 1;
 float osmina = 0.5;
 float sesnaestina = 0.25;
 float tripola = 0.125;
 float sestcetvrt = 0.0625;
 float dvanaestosam = 0.03125;

 float p = (t*1000.0)/8;                     // Vrednost cele pauze, t je trajanje cele note u sekundama/8, za delay funkciju nam trebaju milisekunde.
 int sensor_state;

 sensor_state = digitalRead(sensor);

 if(sensor_state == LOW) {                   // Kako bismo vratili šetača na početak pre svake ponovljene sekvence.

   play(C4, 0.01, 0);

 }

 else {

   //-------------------------------------------------------------------------------------------------------------------------------------------------

 delay(900);
```

```
play(C3, 1, 1);
delay(p*0.75);
play(C3, 0.5, 1);
play(C3, 0.165, 1);
play(C3, 0.165, 0);
play(C3, 0.165, 1);
play(C3, 0.5, 0);
play(C3, 0.165, 0);
play(C3, 0.165, 1);
play(C3, 0.165, 0);
play(G2s, 0.165, 1);
play(G2s, 0.165, 0);
play(G2s, 0.165, 1);
play(G2s, 0.5, 1);
delay(p);

play(C3, 1, 1);
delay(p*0.75);
play(C3, 0.5, 1);
play(C3, 0.165, 1);
play(C3, 0.165, 0);
play(C3, 0.165, 1);
play(C3, 0.5, 0);
play(C3, 0.165, 0);
play(C3, 0.165, 0);
play(C3, 0.165, 0);
play(G2s, 0.165, 1);
play(G2s, 0.165, 0);
play(G2s, 0.165, 1);
play(G2s, 0.5, 1);
delay(p);

play(C3, 1, 1);
delay(p*0.75);
play(C3, 0.5, 0);
play(C3, 0.165, 1);
play(C3, 0.165, 0);
play(C3, 0.165, 1);
play(C3, 0.5, 0);
play(C3, 0.165, 0);
play(C3, 0.165, 0);
play(C3, 0.165, 0);
play(G2s, 0.165, 1);
play(G2s, 0.165, 1);
play(G2s, 0.165, 1);
play(G2s, 0.5, 1);
delay(p);

play(C2, 1.1, 0);
play(C2, 1.1, 0);
play(C2, 1.1, 0);
play(G1s, 0.75, 1);
play(D2s, 0.33, 0);
play(C2, 1.1, 1);
play(G1s, 0.75, 1);
play(D2s, 0.33, 0);
play(C2, 2, 0);
play(G2, 1.1, 0);
play(G2, 1.1, 1);
play(G2, 1.1, 0);
play(G2s, 0.75, 1);
play(D2s, 0.33, 0);
play(C2, 1.1, 1);
play(G1s, 0.75, 1);
play(D2s, 0.33, 0);
play(C2, 2, 0);

play(C3, 1.1, 1);
play(C2, 0.75, 0);
play(C2, 0.25, 1);
play(C3, 1.1, 1);
play(H2, 0.75, 0);
play(A2s, 0.25, 1);
play(lA2, 0.25, 0);
play(G2s, 0.25, 1);
play(lA2, 0.5, 0);
delay(p*32);
play(C2s, 0.45, 1);
delay(p*0.25);
play(F2s, 1.1, 0);
play(F2, 0.75, 1);
play(E2, 0.25, 0);
play(D2s, 0.25, 1);
play(D2, 0.25, 0);
play(D2s, 0.5, 1);
delay(p*32);
play(G1s, 0.45, 1);
play(H1, 1.1, 1);
play(G1s, 0.75, 1);
delay(p*0.25);
play(H1, 0.25, 0);
play(D2s, 1.1, 0);
play(C2, 0.75, 1);
delay(p*0.25);
play(D2s, 0.25, 0);
play(G2, 2.2, 0);

play(C3, 1.1, 1);
play(C2, 0.75, 0);
play(C2, 0.25, 1);
play(C3, 1.1, 1);
play(H2, 0.75, 0);
play(A2s, 0.25, 1);
play(lA2, 0.25, 0);
play(G2s, 0.25, 1);
play(lA2, 0.5, 0);
delay(p*32);
play(C2s, 0.45, 1);
delay(p*0.25);
play(F2s, 1.1, 0);
play(F2, 0.75, 1);
play(E2, 0.25, 0);
```

```
 play(D2s, 0.25, 1);
 play(D2, 0.25, 0);
 play(D2s, 0.5, 1);
 delay(p*32);
 play(G1s, 0.45, 1);
 play(H1, 1.1, 1);
 play(G1s, 0.75, 1);
 delay(p*0.25);
 play(H1, 0.25, 0);
 play(D2s, 1.1, 0);
 play(G1s, 0.75, 1);
 delay(p*0.25);
 play(C2, 0.25, 0);
 play(C2, 2.5, 0);
 delay(p*0.5);

 play(C3, 1, 1);
 delay(p*0.5);
 play(C3, 0.5, 1);
 play(C3, 0.165, 1);
 play(C3, 0.165, 0);
 play(C3, 0.165, 1);
 play(C3, 0.5, 0);
 play(C3, 0.165, 0);
 play(C3, 0.165, 1);
 play(C3, 0.165, 0);
 play(G2s, 0.165, 1);
 play(G2s, 0.165, 0);
 play(G2s, 0.165, 1);
 play(G2s, 0.5, 1);
 delay(p);

 play(C3, 1, 1);
 delay(p*0.75);
 play(C3, 0.5, 1);
 play(C3, 0.165, 1);
 play(C3, 0.165, 0);
 play(C3, 0.165, 1);
 play(C3, 0.5, 0);
 play(C3, 0.165, 0);
 play(C3, 0.165, 1);
 play(C3, 0.165, 0);
 play(G2s, 0.165, 1);
 play(G2s, 0.165, 0);
 play(G2s, 0.165, 1);
 play(G2s, 0.5, 1);
 delay(p);

 play(C3, 1, 1);
 delay(p*0.75);
 play(C3, 0.5, 1);
 play(C3, 0.165, 0);
 play(C3, 0.165, 0);
 play(C3, 0.165, 1);
 play(C3, 0.5, 0);
 play(C3, 0.165, 0);
 play(C3, 0.165, 1);
 play(C3, 0.165, 0);
 play(G2s, 0.165, 1);
 play(G2s, 0.165, 0);
 play(G2s, 0.165, 1);
 play(G2s, 0.5, 0);

 play(G2, 0.5, 0);
 play(G2, 0.165, 0);
 play(G2, 0.165, 0);
 play(G2, 0.165, 0);

 play(G2s, 0.165, 0);
 play(G2s, 0.165, 0);
 play(G2s, 0.165, 0);
 play(A2s, 0.5, 0);

 play(C3, 0.5, 0);
 play(C3, 0.165, 0);
 play(C3, 0.165, 0);
 play(C3, 0.165, 0);
 play(C3s, 0.165, 1);
 play(C3s, 0.165, 0);
 play(C3s, 0.165, 1);
 play(F3, 0.5, 0);

 delay(p*256);

/*
 play(D2, 2.2, 1);
 delay(p*2.2);
 play(D2, 2.2, 1);
 delay(p*2.2);
 play(D2, 2.2, 1);
 delay(p*2.2);
 play(D2, 2.2, 1);
 delay(p*2.2);
 play(D2, 2.2, 0);
 delay(p*2.2);
 play(D2, 2.2, 0);
 delay(p*2.2);
 play(D2, 2.2, 0);
 delay(p*2.2);
 play(D2, 2.2, 0);
 delay(p*2.2);
*/
//------------------------------------------------------------------------------------------------------------------------------------------------------------------

 play(C2, 1.5, 1);
 play(C2, 0.25, 1);
 play(C2, 0.25, 0);
 play(C2, 1.5, 1);
 play(C2, 0.25, 0);
 play(C2, 0.25, 1);
 play(C2, 1, 1);
 play(C2, 1, 1);
 play(C2, 0.5, 0);
play(C2, 0.5, 1);
play(C2, 0.33, 0);
play(C2, 0.33, 1);
play(C2, 0.33, 0);

play(E2, 1.5, 1);
play(C2, 0.25, 1);
play(C2, 0.25, 0);
play(E2, 1.5, 0);
play(C2, 0.25, 1);
play(C2, 0.25, 0);
play(E2, 1, 1);
play(E2, 1, 0);
play(E2, 0.5, 1);
play(E2, 0.5, 0);
play(E2, 0.33, 1);
play(C2, 0.33, 0);
play(E2, 0.33, 1);

play(G2, 1.5, 1);
play(E2, 0.25, 1);
play(E2, 0.25, 0);
play(G2, 1.5, 0);
play(E2, 0.25, 1);
play(E2, 0.25, 0);
play(G2, 1, 1);
play(G2, 1, 0);
play(G2, 0.5, 1);
play(G2, 0.5, 0);
play(G2, 0.33, 1);
play(E2, 0.33, 0);
play(G2, 0.33, 1);

play(lA2, 2.5, 0);
delay(p*0.5);
play(lA2, 0.33, 1);
play(lA2, 0.33, 0);
play(lA2, 0.33, 1);
play(H2, 1.5, 1);
play(C3, 0.1875, 0);
play(H3, 0.1875, 0);
play(lA3, 0.1875, 0);
play(G3, 0.1875, 0);
play(C3, 0.1875, 1);
play(H3, 0.1875, 1);
play(lA3, 0.1875, 1);
play(G3, 0.1875, 1);
play(C3, 1, 0);
play(C3, 0.33, 1);
play(C3, 0.33, 0);
play(C3, 0.33, 1);
play(C3, 0.5, 0);
delay(p*32);

play(C2, 0.75, 1);
play(C2, 0.25, 1);
play(F2, 2, 1);
play(G2, 1.35, 1);
play(G2s, 0.33, 0);
play(A2s, 0.33, 1);
play(G2s, 2, 0);
play(C2, 1.1, 1);
delay(p*0.5);

play(C2, 0.5, 1);
play(C2, 0.5, 0);
play(F2, 1.7, 1);
play(G2, 0.5, 1);
play(G2s, 0.5, 0);
play(C2, 0.5, 1);
play(lA2, 0.33, 0);
play(F2, 0.33, 1);
play(C3, 0.33, 0);
play(A2s, 3, 0);

play(C2, 0.75, 1);
play(C2, 0.25, 1);
play(F2, 1.75, 0);
play(G2, 0.25, 1);
play(G2s, 0.75, 1);
play(F2, 0.25, 1);
play(C3, 0.75, 1);
play(lA2, 0.25, 1);
play(F3, 2, 0);
play(F2, 1, 0);

play(G2s, 0.33, 1);
play(G2, 0.33, 0);
play(F2, 0.33, 1);
play(C3, 1.33, 1);
play(G2s, 0.33, 0);
play(F2, 0.33, 0);
play(C2, 1, 1);
play(C2, 0.75, 0);
play(C2, 0.25, 1);
play(F2, 2.5, 1);
delay(p*0.5);

play(C2, 0.75, 1);
play(C2, 0.25, 1);
play(F2, 2, 0);
play(G2, 1.35, 1);
play(G2s, 0.33, 0);
play(A2s, 0.33, 1);
play(G2s, 2, 0);
play(C2, 1.1, 1);
delay(p*0.5);

play(C2, 0.5, 1);
play(C2, 0.5, 1);
play(F2, 1.7, 0);
play(G2, 0.5, 1);
play(G2s, 0.5, 0);
```

```
 play(C2, 0.5, 1);
 play(lA2, 0.33, 0);
 play(F2, 0.33, 1);
 play(C3, 0.33, 0);
 play(A2s, 3, 1);

 play(C2, 0.75, 1);
 play(C2, 0.25, 1);
 play(F2, 1.75, 0);
 play(G2, 0.25, 1);
 play(G2s, 0.75, 0);
 play(F2, 0.25, 1);
 play(C3, 0.75, 0);
 play(lA2, 0.25, 1);
 play(F3, 2, 0);
 play(F2, 1, 1);

 play(G2s, 0.33, 1);
 play(G2, 0.33, 0);
 play(F2, 0.33, 1);
 play(C3, 1.33, 1);
 play(G2s, 0.33, 0);
 play(F2, 0.33, 0);
 play(C2, 1, 1);
 play(C2, 0.75, 0);
 play(C2, 0.25, 1);
 play(F2, 2.5, 0);
 delay(p*0.41);

 play(F2, 1.5, 1);
 play(F2, 0.25, 1);
 play(F2, 0.25, 0);
 play(F2, 1.5, 1);
 play(F2, 0.25, 0);
 play(F2, 0.25, 0);
 play(F2, 1, 1);
 play(F2, 1, 1);
 play(F2, 0.5, 0);
 play(F2, 0.5, 1);
 play(F2, 0.33, 0);
 play(F2, 0.33, 1);
 play(F2, 0.33, 0);

 play(lA2, 1.5, 1);
 play(lA2, 0.25, 0);
 play(lA2, 0.25, 0);
 play(lA2, 1.5, 1);
 play(lA2, 0.25, 0);
 play(lA2, 0.25, 1);
 play(D3, 1, 0);
 play(D3, 1, 1);
 play(D3, 0.5, 0);
 play(D3, 0.5, 1);
 play(D3, 0.33, 0);
 play(D3, 0.33, 1);
 play(D3, 0.33, 0);

 play(E3, 1.5, 0);
 delay(p*0.25);
 play(C3, 0.1875, 1);
 play(H3, 0.1875, 0);
 play(lA3, 0.1875, 1);
 play(G3, 0.1875, 1);
 play(C3, 0.1875, 0);
 play(H3, 0.1875, 1);
 play(lA3, 0.1875, 0);
 play(G3, 0.1875, 1);
 play(C3, 1, 1);
 play(C3, 0.33, 0);
 play(C3, 0.33, 1);
 play(C3, 0.33, 0);
 play(C3, 1, 1);

   //------------------------------------------------------------------------------------------------------------------------------------------------------

 delay(900);

}

}

void play(float f, float v, int s) {       // (frekvencija, vrednost, smer (bilo sta vece od 0 (koristim 1 uglavnom) je napred, 0 nazad))

   int i;
   float h;
   float d;                                // duration
   float p = (t*1000.0)/8;                 // vrednost cele pauze, t je trajanje cele note u sekundama/8, za delay funkciju nam trebaju milisekunde.

   h = (1000.0)/f;                         // koliko puta u 1000ms je frekvencija f, iz toga sledi h- period.
   d = (v*f*t);                            // vrednost note ponovljene frekvencija puta (to je za trajanje od jedne sekunde) pomnožena sa t.
                                           // Za t=1 imam 60bpm- 60 nota u minuti.

   if (s>0) {

    for (i = 0; i <= d; i++) {

      digitalWrite(LEDb, HIGH);

      digitalWrite(coila1, HIGH);
      digitalWrite(coila2, LOW);
      digitalWrite(coilb1, LOW);
      digitalWrite(coilb2, HIGH);
      delay(h);

      digitalWrite(coila1, HIGH);
      digitalWrite(coila2, LOW);
      digitalWrite(coilb1, HIGH);
      digitalWrite(coilb2, LOW);
      delay(h);

      digitalWrite(coila1, LOW);
      digitalWrite(coila2, HIGH);
      digitalWrite(coilb1, HIGH);
      digitalWrite(coilb2, LOW);
      delay(h);

      digitalWrite(coila1, LOW);
      digitalWrite(coila2, HIGH);
      digitalWrite(coilb1, LOW);
      digitalWrite(coilb2, HIGH);
      delay(h);

      digitalWrite(LEDb, LOW);

    }

   }

   else {

     for (i = 0; i <= d; i++) {

      digitalWrite(LEDc, HIGH);

      digitalWrite(coila1, LOW);            // Sekvenca u nazad- imajući u vidu koji je bio poslednji korak ranije, eksperimentisati sa ovim.
      digitalWrite(coila2, HIGH);
      digitalWrite(coilb1, HIGH);
      digitalWrite(coilb2, LOW);
      delay(h);

      digitalWrite(coila1, HIGH);
      digitalWrite(coila2, LOW);
      digitalWrite(coilb1, HIGH);
      digitalWrite(coilb2, LOW);
      delay(h);

      digitalWrite(coila1, HIGH);
      digitalWrite(coila2, LOW);
      digitalWrite(coilb1, LOW);
      digitalWrite(coilb2, HIGH);
      delay(h);

      digitalWrite(coila1, LOW);
      digitalWrite(coila2, HIGH);
      digitalWrite(coilb1, LOW);
      digitalWrite(coilb2, HIGH);
      delay(h);

      digitalWrite(LEDc, LOW);

    }

  }

  delay(p*v*21);                              // Pauza nakon svake note, definisana trajanjem note. Pomaze u postizanju staccato osecaja i uopste boljoj
definiciji note.

Serial.print("Frekvencija: ");
Serial.print(f);
Serial.print("  _____  Period: ");
Serial.print(h);
Serial.print("  _____  Broj ponavljanja: ");
Serial.println(d);

}
```