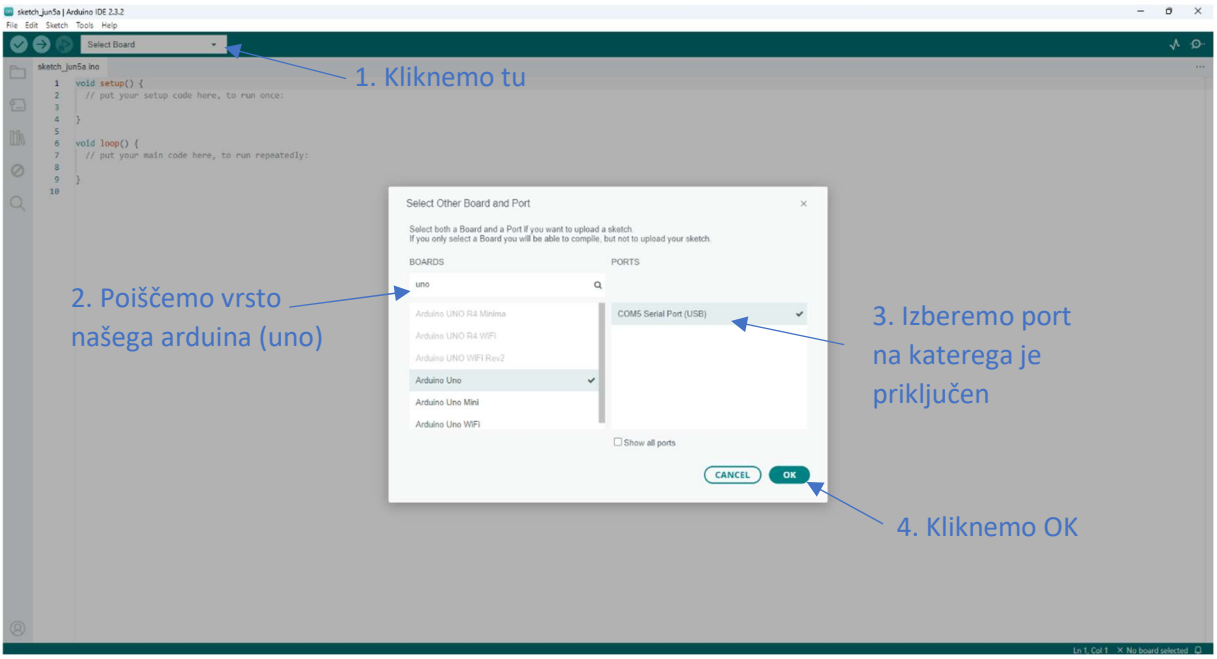
# Priročnik

1. **Ura:**
   a. Osnove ploščice arduino (Pregled lista z razloženo vsebino škatle)
      i. Digitalni vhodi/izhodi (2-13) – Napetost na njih je lahko ali 0 V ali 5 V (logična 0 ali 1). Pina 0 in 1 sta uporabljena za komunikacijo z računalnikom in se jima načeloma izogibamo. Torej ko bo neka komponenta priklopljena na digitalni pin in bo na tem pinu napetost 5 V potem bo ta komponenta delovala.
      ii. Analogni vhodi (A0-A5)– Z njimi lahko beremo »analogne vrednosti« napetosti (Napetosti med 0 V in med 5 V) . Arduino vrne te vrednosti v obliki števila med 0 in 1023 (0 = 0 V, 1023 = 5V). Dejanske vrednosti napetosti izračunamo po formuli (5/1023*izmerjena vrednost na analognem vhodu). Te pine pa lahko uporabljamo tudi kakor navadne digitalne pine.
      iii. USB priključek – Z njim povežemo arduino z računalnikom (preko njega nalagamo kodo in hkrati napajamo arduino)
      iv. Priključek za napajanje – Z njim lahko napajamo arduino tudi ko ni več priklopljen z USBjem na računalnik (kodo smo pred tem naložili na arduino, saj program ostane zapisan na pomnilniku)
      v. Gumb za ponovni zagon programa

   b. Osnovne funkcije v arduino
      i. Prenos ARDUINO IDE (Integrated Development Environment) okolja če še ni: https://www.arduino.cc/en/donate/newsletter
      (kliknite spodaj na: »JUST DOWNLOAD«)

      Uporabljamo ga za pisanje in nalaganje kode na arduino ploščico.

      ii. Priključitev ploščice na računalnik in izbira pravega COM porta in vrste ploščice.

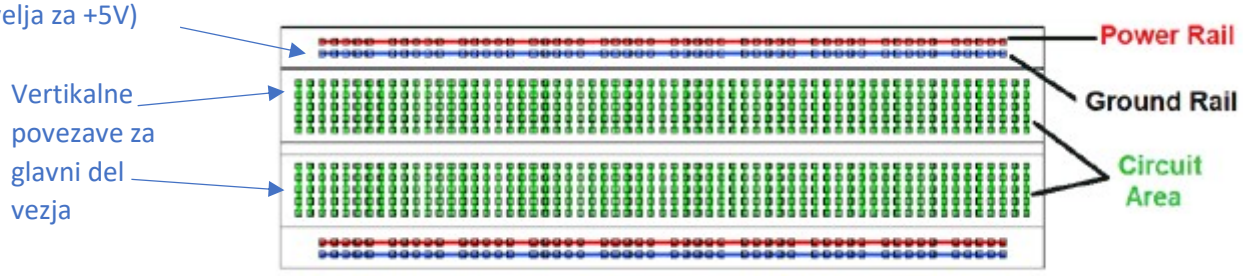


c. Vezava na protoboardu
    i. Na protoboard (glej sliko protoboarda) priključimo diodo (pazimo da je daljša žička na +5 V saj dioda prepušča tok samo v eno smer), upor (220 ohm – vrednost je napisana na pakiranju, barve na uporu nam povejo koliko ohmski je - resistor color code), žičko v digitalni port 10 in eno v GND.
        1. Kako deluje električni tokokrog?

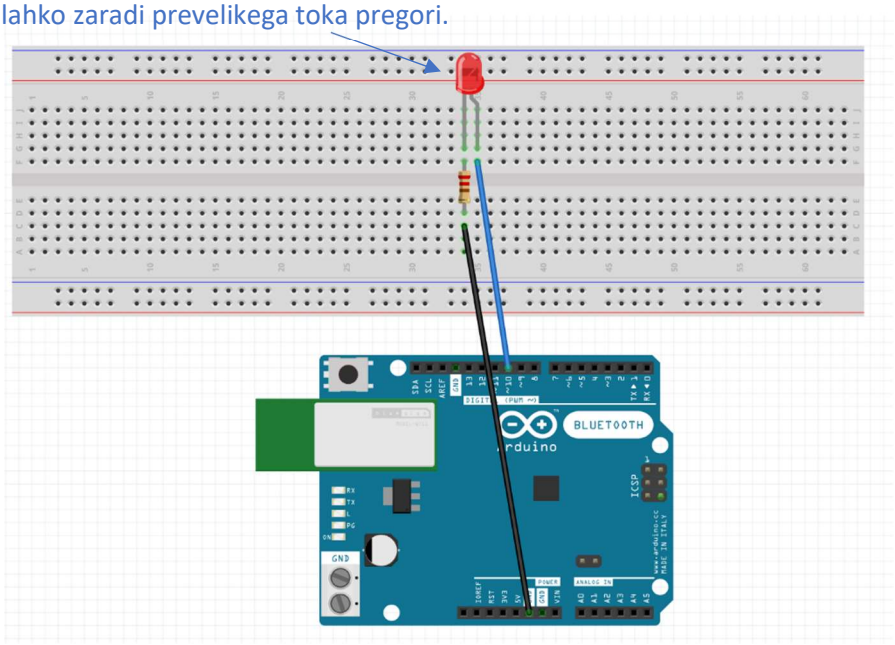
            Tokokrog je sestavljen iz različnih komponent, ki so povezane tako, da omogočajo kroženje električnega toka po poti, ki jo določajo te komponente. Glavne komponente električnega tokokroga so:

            - Vir napetosti (baterija, generator): Vir napetosti zagotavlja potrebno električno energijo za poganjanje toka skozi tokokrog. Primeri vključujejo baterije, akumulatorje in električne generatorje.
            - Prevodniki (žice): Prevodniki so materiali, običajno kovine, ki omogočajo pretok električnega toka zaradi svoje nizke električne upornosti. Uporabljajo se za povezovanje komponent v tokokrogu.
            - Električne komponente (upor, kondenzator, tuljava, diode, tranzistorji, itd.): To so elementi, ki imajo različne funkcije v tokokrogu. Na primer, upor omejuje tok, kondenzator shranjuje energijo v električnem polju, tuljava shranjuje energijo v magnetnem polju, diode omogočajo tok v eno smer, tranzistorji pa delujejo kot stikala ali ojačevalci.
            - Stikala: Stikala omogočajo ročno ali avtomatsko prekinitev ali vzpostavitev električnega tokokroga. S tem nadzorujemo, kdaj bo tok tekel skozi tokokrog.
            - Obremenitev (npr. žarnica, motor): To so naprave ali komponente, ki porabljajo električno energijo za opravljanje dela. Na primer, žarnica pretvarja električno energijo v svetlobo in toploto, motor pretvarja električno energijo v mehansko delo.
            - Električni tok teče od večje napetosti proti manjši. GND ima napetost 0 V, zato bo tok vedno tekel proti GND.

Ker ima arduino samo 1 +5V priključek in 3 GND priključke, pri večjih vezjih vežemo vse GND na modro vrstico in nato samo eno žičko od tu na arduino (enako velja za +5V)




LED dioda. Krajšo žičko vedno vežemo na GND, saj je dioda polprevodnik in prevaja samo v eno smer. Daljšo žičko vedno vežemo na višji potencial (tam kjer je napetost). Z diodo tudi vedno zaporedno (pred ali za njo) vežemo 220 ohmski upor, zato da omejimo tok, saj lahko zaradi prevelikega toka pregori.

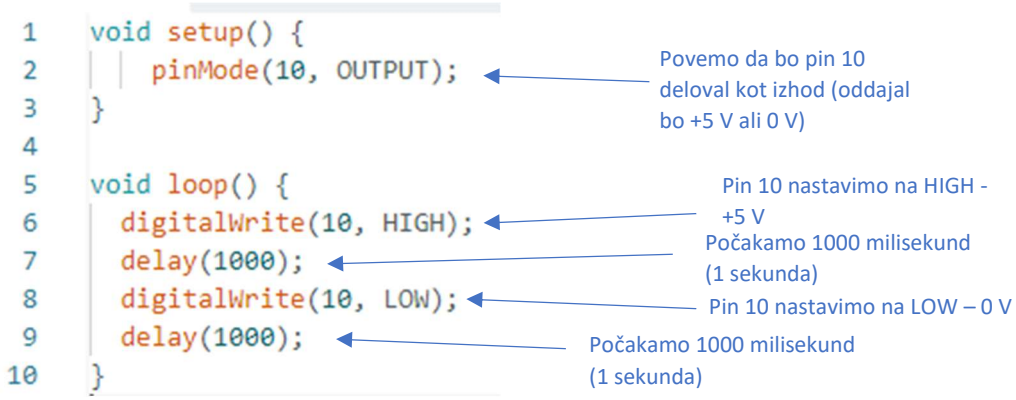


# Pisanje programa

ii. Razlaga void setup() in void loop()


iii. PRI PISANJU PROGRAMOV MORAMO PAZITI NA VELIKE IN MALE ČRKE!

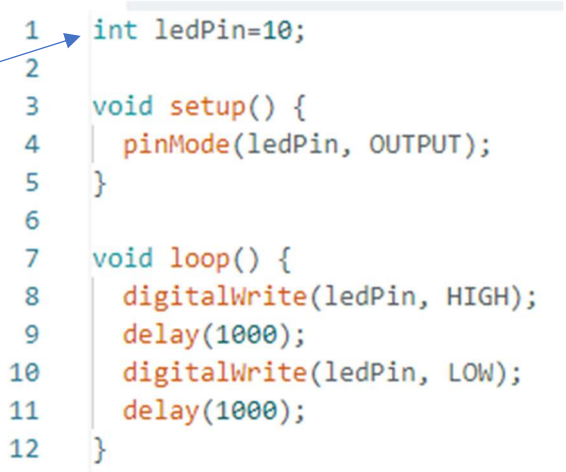
V IDE najprej nastavimo digitalni port na izhod (pinMode). Pazimo da na koncu vsakega ukaza zapišemo podpičje ( ; ) Nato ga nastavimo na HIGH (digitalWrite). Sedaj je dioda samo prižgana. Nato nastavimo diodo na LOW. Vidimo da se ne dogaja nič, saj se dioda samo zelo hitro vžiga in ugaša in naše oči tega ne morejo zaznati. Zato dodamo sekundni zamik po obeh digitalWrite funkcijah.




Led dioda se bo prižgala vsako sekundo in se ugasnila po eni sekundi

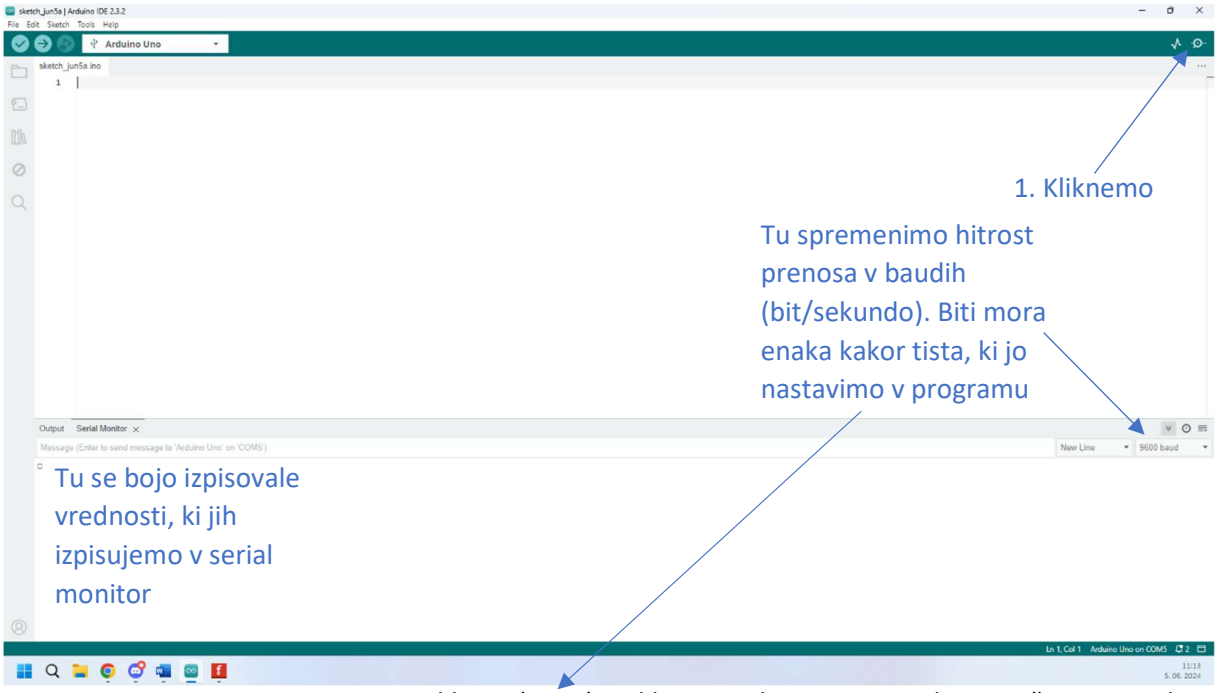
iv. Številke portov zamenjamo z spremenljivkami in razložimo spremenljivke

Integer, krajšamo z int. V spremenljivki tipa integer lahko shranjujemo celoštevilske vrednosti.



Namesto da uporabljamo številke, je boljši način, da čim več uporabljamo spremenljivke. Spremenljivke so v programiranju vrednosti, ki jih lahko spreminjamo in z njimi opravljamo logične in aritmetične operacije. Osnovno poznamo integer (int), floating point (float), character(char) in boolean (bool). Z spremenljivkami tipa integer hranimo celoštevilske vrednosti. Z floating pointom hranimo decimalne vrednosti, z characterjem hranimo znake (ASCII) in z booleanom hranimo logično 1 ali 0. Lahko jim določimo vrednost že takoj, ali pa jih samo ustvarimo in jim vrednost določimo kasneje.

2. **Ura:**
    a. Nadaljevanje osnov (serial monitor, if stavki, while in for zanka)
        i. Serial monitor
            1. Uporablja se za izpisovanje na zaslon računalnika (v IDE). Z njim lahko spremljamo potek programa




2. Serial.begin(9600); - vklopi serial monitor. Ta ukaz napišemo v void setup.
3. Serial.print(ime spremenljivke); – Izpiše vrednost spremenljivke v serial monitor
4. Serial.print("Arduino"); Na zaslon bo izpisalo besedo arduino
5. Serial.println(); Po izpisu se premakne v novo vrstico

ii. If, else in else if stavki so logični stavki, ki jih uporabljamo za reševanje problemov in logični potek kode.

1. If stavek se uporablja za izvajanje neke kode, če je nek zastavljen pogoj pravilen. Da začnemo if stavek, moramo zapisati ključno besedo »if« in nato v oklepaju določiti pogoj. Nato znotraj vijugastih oklepajev zapišemo kaj se zgodi če je pogoj pravilen.

2. Else if stavek se uporablja kot drugi if stavek po prvem if stavku. Če prvi pogoj ni pravilen, lahko v else if stavku zastavimo še en pogoj pod katerim se bo izvedla neka druga koda. Else if stavek zapišemo z ključno besedo »else if«

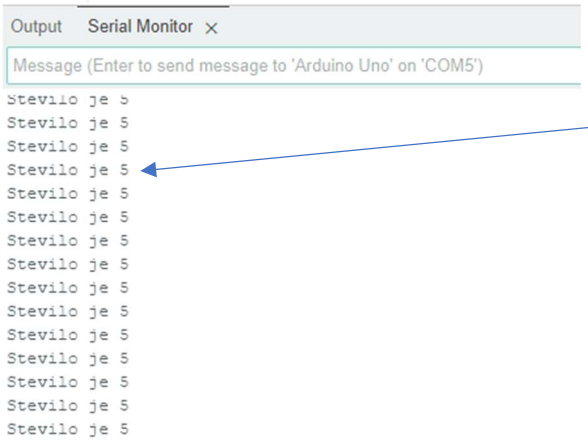
3. Else stavek se uporablja po if stavku. Else stavek izvede neko kodo če je bil pogoj if stavka napačen. Else stavek zapišemo z ključno besedo »else«. Tukaj ne določimo pogoja, saj se izvede če je bil pogoj if stavka napačen.

```
1  int stevilo=5;
2
3  void setup() {
4    Serial.begin(9600);
5  }
6
7  void loop() {
8    if(stevilo>5){
9      Serial.println("Stevilo je vecje od 5 ");
10   }
11   else if(stevilo==5){
12     Serial.println("Stevilo je 5 ");
13   }
14   else{
15     Serial.println("Stevilo je manjse od 5 ");
16   }
17
18 }
19
```

Preveri če je število večje od 5. Če je potem bo izpisalo »Stevilo je vecje od 5«

Če if stavek ni bil pravilen potem se preveri pogoj if stavka. Pazimo da uporabimo == in ne =, saj z == preverjamo vrednost, z = pa jo dodeljujemo

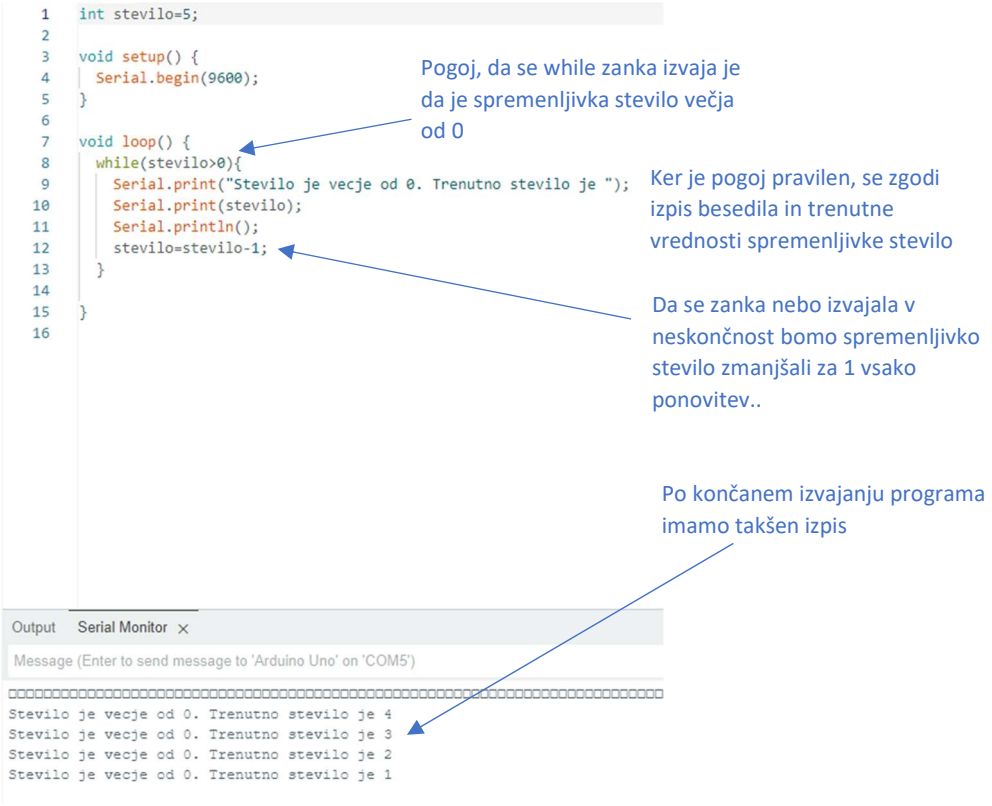
Če noben od if in else if stavkov ni bil pravilen, potem bo izpisalo da je število manjše od 5, saj je to še edina preostala možnost



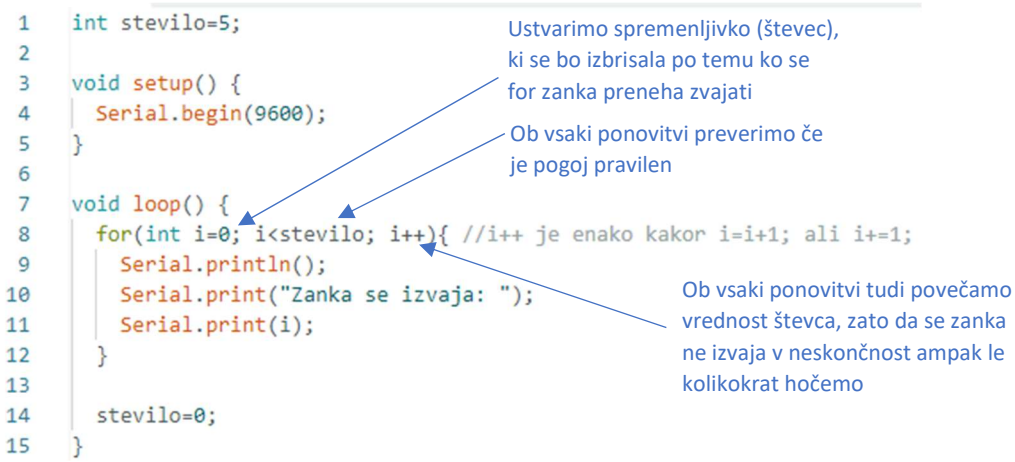

Ker je število enako 5, potem se vsakič izvede koda v else if stavku, kar je v našem primeru da izpiše »število je 5«

iii. Zanke (ang. loop) v programskih jezikih uporabljajmo za ponavljanje neke kode. V C++ poznamo tri različne vrste zank ki so while zanke, do while zanke in for zanke.

1. While zanka se izvaja če je pogoj resničen. Izvaja se dokler pogoj ni več resničen.




2. For zanko uporabljamo, ko vemo kolikokrat hočemo zanko ponoviti. Najprej določimo neko spremenljivko, nato se preveri pogoj in nato še našo vrednost povečamo, da se zanka ne izvaja v neskončnost.

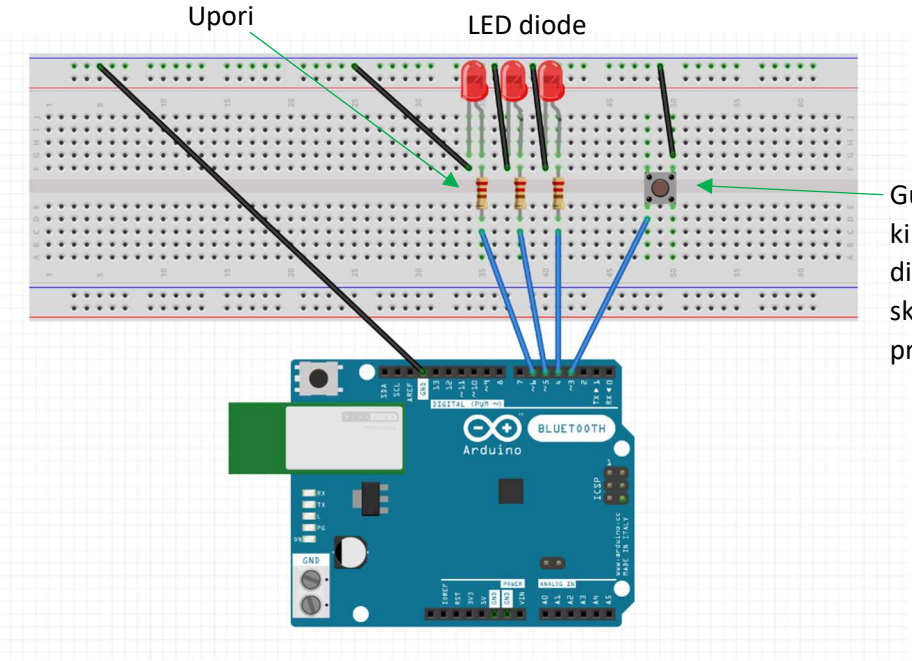



3. Poznamo tudi do while zanko, ki pa si je na delavnici ne bomo pogledali, po principu pa je zelo podobna while zanki in si jo lahko udeleženci pogledajo doma, če jih zanima (bistvo te zanke je da se najprej enkrat izvede koda v zanki, nato pa se šele preveri pogoj), to pomeni da se bo zanka izvedla vsaj enkrat tudi če pogoj ni pravilen).

4. Zanke lahko tudi ugnezdimo. To pomeni da imamo zanko, ki se izvaja znotraj neke druge zanke.

b. 3-strana kocka (3 diode) z ničlo (ob pritisku gumba, bo zasvetilo naključno število ledic, lahko tudi nobena).
   i. Sestavljanje vezja
      1. 3x Led dioda
      2. 3x 220 Ohm upor
      3. 1x Gumbek
      4. 9x M-M žička (moški priključek na obeh straneh)

      V shemah vezave so črne žičke povezave, ki vodijo v GND, rdeče žičke vodijo v +5V, modre žičke pa so uporabljene za povezavo pinov z elementi (zaradi preglednosti). Pri izvajanju se lahko uporabi katerakoli barva.




   ii. Pisanje programa

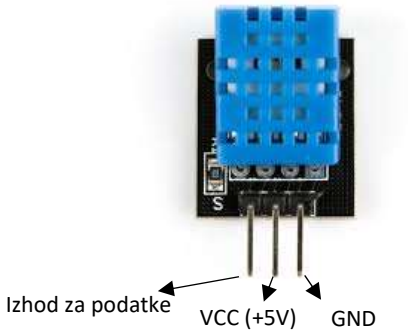
Najprej naredimo razmislek. Vprašamo se, kako bi začeli pisati ta program. Imena spremenljivk so lahko drugačna. Komentarji so v tej kodi samo v pomoč izvajalcu. Pri izvajanju delavnice jih ne prepisujemo, lahko pa uporabljamo svoje komentarje za pisanje opomb itd…

Za povezovanje gumbov uporabljamo »INPUT_PULLUP«. Ko nastavimo pin »but« kot »input_pullup«, uporabljamo vgrajen pull-up upor na Arduinu. To pomeni, da bo vrednost na tem pinu 5 V (HIGH), ko gumb ni pritisnjen, in 0 V (LOW), ko je gumb pritisnjen. S tem zagotovimo stabilno delovanje pina, saj brez pull-up upora lahko pin pobira električne motnje iz okolja, kar bi povzročilo napačna branja. Vgrajeni pull-up upor na Arduinu omogoča preprosto uporabo z nastavitvijo »input_pullup« brez potrebe po dodajanju zunanjih komponent.

KODA:

```
int led1 = 6; //definicija spremenljivk
int led2 = 5;
int led3 = 4;
int but = 3; //pin na katerem bo gumbek
int rng; //spremenljivka ki bo hranila naključno vrednost

void setup() {
  pinMode(led1, OUTPUT); //nastavitev pinov na katere si priklopljene ledice na izhod
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(but, INPUT_PULLUP); //nastavitev pina na katerem je gumb na vhod z pullup uporom
  Serial.begin(9600);  //vklop in nastavitev serial monitorja na 9600 baudov
}

void loop() {
    if (digitalRead(but) == LOW) {  //preveri če je stanje na pinu z gumbom LOW – gumb je pritisnjen

        rng = random(0, 4);  //Dobi naključno vrednost med 0 in 3 (random(vključno, izključno) in jo zapiše v spremenljivko rng. Nato jo izpiše v serial monitor (za preverjanje pravilnosti delovanja)
        Serial.println(rng);

    if (rng == 0) {             //Preveri če je vrednost rng 0
      digitalWrite(led1, LOW); //Ugasne vse lučke
      digitalWrite(led2, LOW);
      digitalWrite(led3, LOW);
    } else if (rng == 1) {      //Preveri če je vrednost rng 1
      digitalWrite(led1, HIGH); //Prižge 1. lučko
      digitalWrite(led2, LOW);
      digitalWrite(led3, LOW);
    } else if (rng == 2) {      //Preveri če je vrednost rng 2
      digitalWrite(led1, HIGH); //Prižge 1. in 2. lučko
      digitalWrite(led2, HIGH);
      digitalWrite(led3, LOW);
    } else if (rng == 3) {       //Preveri če je vrednost rng 3. Tu bi lahko uporabilo tudi else stavek saj je vrednost 3 še edina preostala možnost

      digitalWrite(led1, HIGH);
      digitalWrite(led2, HIGH); //Prižge vse 3 lučke
      digitalWrite(led3, HIGH);
    }

    while (digitalRead(but) == LOW) {  //Preveri ali je gumb še vedno pritisnjen
      delay(10);                       //Program ne dela nič – ne gre ponovno generirati naključne vrednosti dokler gumba ne spustimo in ga nato ponovno pritisnemo. 10 milisekundni zamik dodamo za razbremenitev procesorja
    }
  }
}
```

3. **Ura:**
   a. Razlaga DHT11 senzorja
      i. Uporablja se za merjenje vlage in temperature.

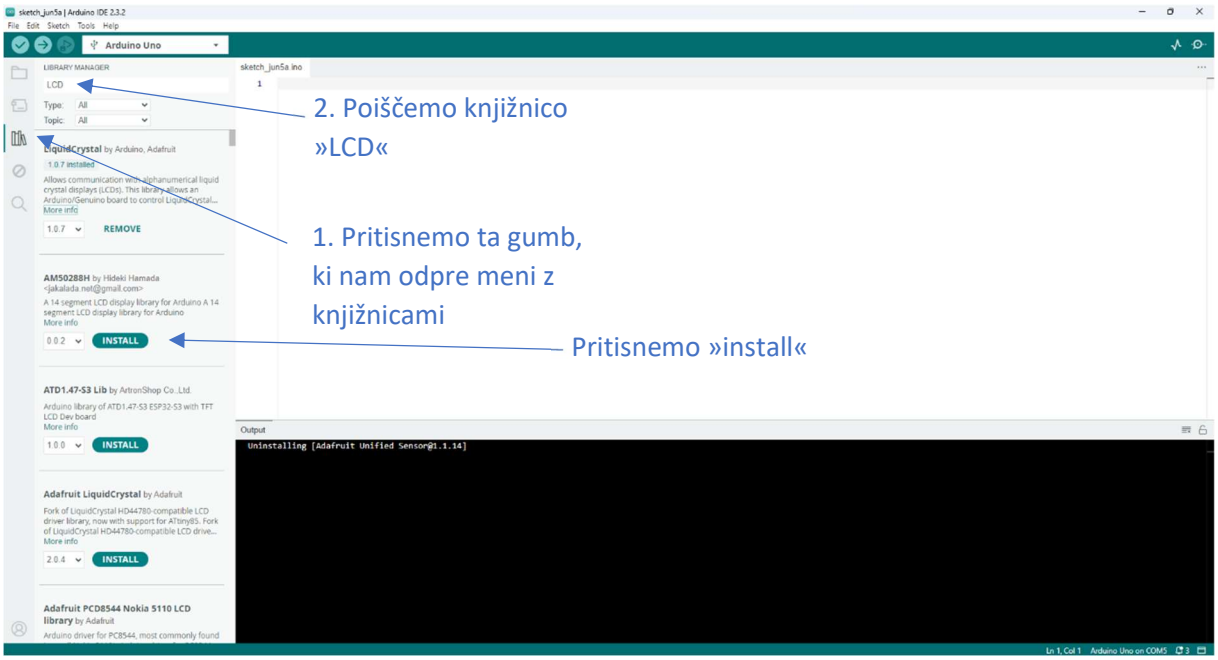



   b. Razlaga LCD (Liquid Crystal Display) zaslona
      i. Poiščejo ga v kompletih
      ii. Je vrsta zaslona, ki uporablja tekoče kristale za prikazovanje slik. Tekoči kristali se nahajajo med dvema polariziranima steklenima ploščama in spreminjajo svojo orientacijo pod vplivom električnega polja, kar omogoča prehajanje ali blokiranje svetlobe.

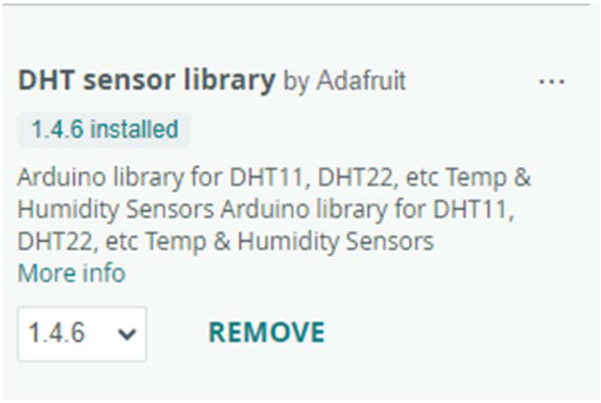

- VSS – GND
- VDD - +5V
- VO – Nastavljanje kontrasta (potenciometer)
- RS – Za nastavitev registra
- R/W – Nastavitev načina branja ali pisanja (Ponavadi vezan na GND saj ga LOW nastavi na WRITE način saj redko beremo iz zaslona
- E – Vklopi pisanje v register
- D0 do D7 – Pošiljanje podatkov na zaslon
- A – Anoda za osvetlitev ozadja (+5V vezan preko upora)
- K – Katoda za osvetlitev ozadja (GND)

c. Uporaba knjižnic
    i. Za lažje krmiljenje elektronskih kompleksnih elektronskih komponent in senzorjev uporabljamo ti. knjižnice. Knjižnice so zbirke funkcij, ki jih lahko naložimo in uporabimo v svojem programu in nam tako olajšajo delo.
    ii. Za nadzorovanje DHT11 senzorja in za krmiljenje LCD zaslona bomo uporabljali knjižnici.
    iii. Veliko knjižnic lahko najdemo že v arduino IDE. Tiste, katerih pa ne najdemo, pa lahko najdemo na spletni strani »GitHub«, ki je spletna stran za deljenje projektov in kode na različnih področjih računalništva.

d. Nalaganje knjižnic

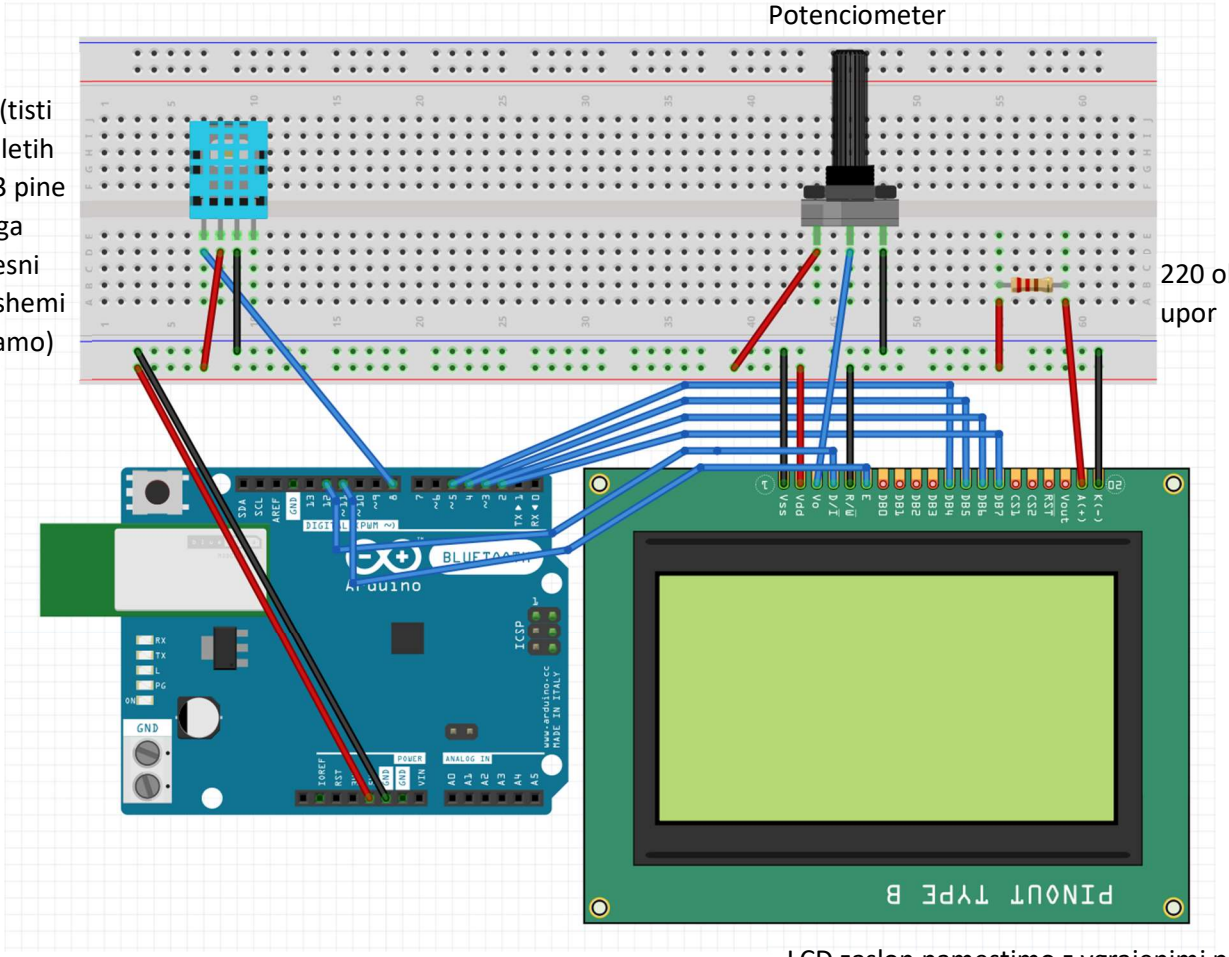



    i. Enako kakor je na sliki ponovimo še za DHT11 senzor. V iskalnik napišemo »DHT« in poiščemo »DHT sensor library by Adafruit«.
    ii. Če pritisnemo »more info« dobimo podatke o knjižnici in kako uporabljati vse njene funkcije.




e. Izdelava vremenske postaje
    i. 1x Potenciometer – nastavljiv upor. S tem ko ga vrtimo, spreminjamo vrednot upora v njem in tako čez spušča manjši tok. Uporabljali ga bomo za nastavljanje svetlosti LCD zaslona
    ii. 1x DHT11 senzor

iii. 8x M-M žička
iv. 12x M-F žička (moški in ženski priključek)
v. 1x 220 ohmski upor
vi. LCD zaslon



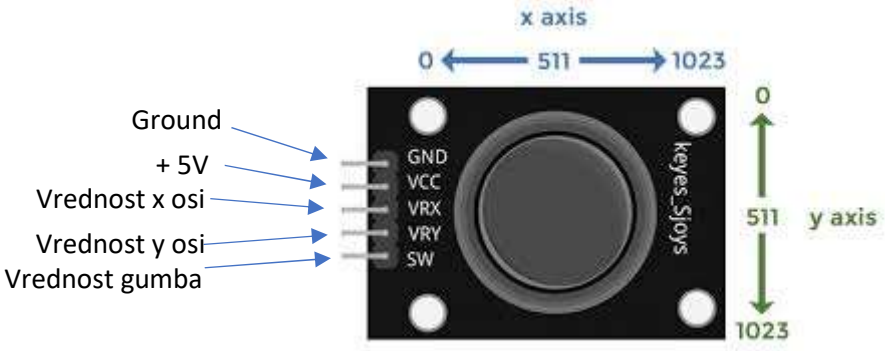

LCD zaslon namestimo z vgrajenimi pini na protoboard in nato naredimo povezave na protoboardu(na skici so narejene direktno na LCD zaslon zaradi razumljivosti)

f. Pisanje programa

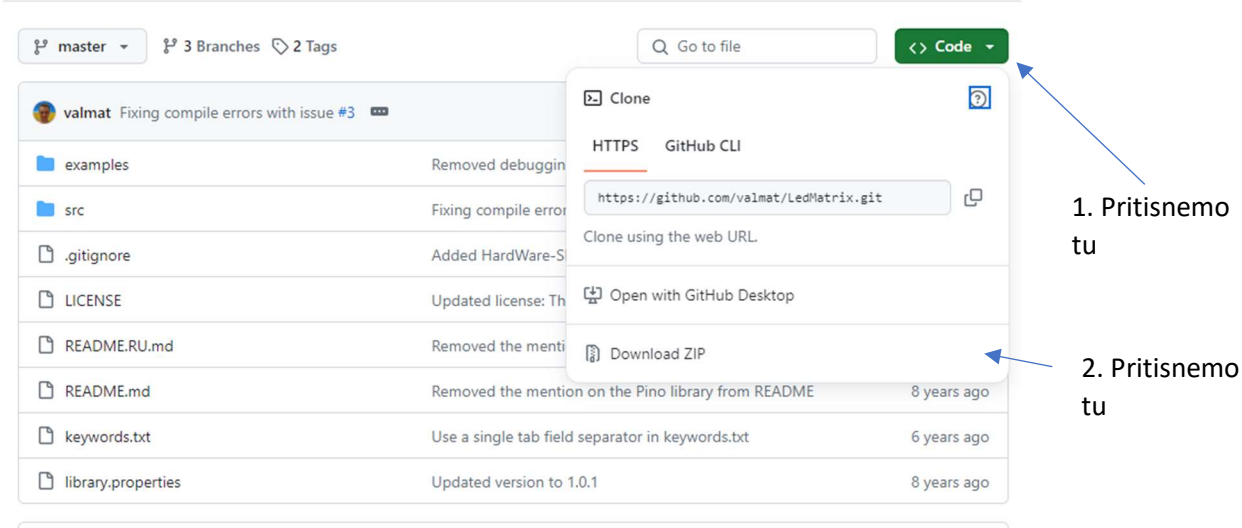
```
#include <LiquidCrystal.h> //uporabimo knjižnico za lcd

#include <DHT.h> //uporabimo knjižnico za DHT senzor

const int DHTTYPE = DHT11; //povemo katero vrsto DHT senzorja uporabljamo
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2; //povemo kateri pin na lcd je priključen na kateri pin na arduinu. Če spredaj napišemo »const« to pomeni da je ta vrednost konstanta in se nebo spreminjala. Namesto da pišemo vsako spremenljivko enakega tipa posebej, lahko uporabimo zgornji zapis.

LiquidCrystal lcd(rs, en, d4, d5, d6, d7); //definicija pinov za delovanje funkcije. Namesto dejanskih številk pinov smo uporabili konstante, ki smo jih prej določili.

DHT dht(8, DHT11); //definiramo na katerem pinu so podatki, ki jih pošilja DHT senzor in vrsto senzorja

void setup() {
  lcd.begin(16, 2); //Vklopimo naš LCD in definiramo koliko stolpcev in vrstic ima
  dht.begin(); //začne delovanje DHT  senzorja
  lcd.clear(); //pobriše zaslon
  //Pinov lcd zaslona in dht senzorja ni potrebno definirati, saj to stori knjižnica že sama
}

void loop() {
  lcd.setCursor(0, 0); //nastavi začetek pisanja na prvi stolpec in vrstico (začne šteti z 0)
  lcd.print("Temp: "); //Izpiše Temp: in presledek
  lcd.print(dht.readTemperature()); //izpiše temperaturo, ki jo je prebral DHT senzor
  lcd.print((char)223); //223 (ASCII vrednost za ° ) bo spremenilo v obliko znaka in ga izpisalo, saj ne prepozna če vnesemo notri samo znak. Nato izpiše še C
  lcd.print("C");

  lcd.setCursor(0, 1); //začne pisati v naslednji vrstici
  lcd.print("Vlaga: "); //izpiše Vlaga: in presledek
  lcd.print(dht.readHumidity()); //izpiše vlago, ki jo je prebral DHT senzor
  lcd.print("%"); //Izpiše znak %

  delay(5000); //5 sekundni zamik pred naslednjim branjem
}
```

g. Možnost razširitve
   i. Barometer (če ga dokupiš)
   ii. Senzor dežja (če ga dokupiš)
   iii. Senzor svetlobe (fotorezistor)
   iv. Sproženje alarma ob določeni temperaturi/vlagi…

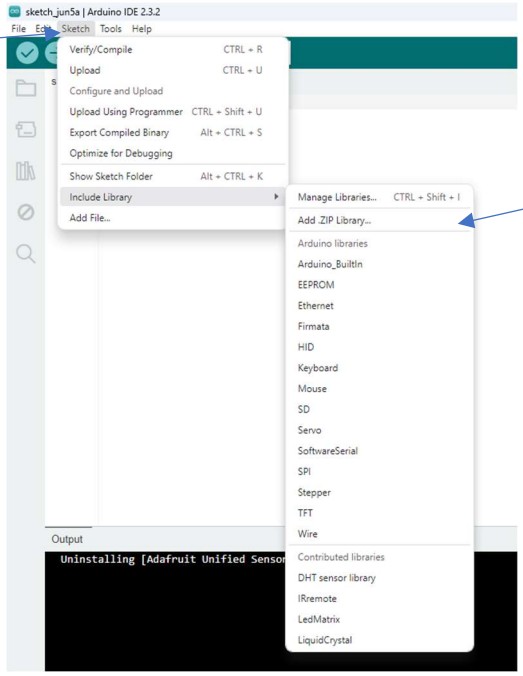
4. **Ura:**
    a. Razlaga 8x8 diodne matrike (shift register, priključki)
        i. 8x8 LED matrika je zaslon, sestavljen iz 64 LED diod, razporejenih v mrežo 8 vrstic in 8 stolpcev. Uporablja se za prikazovanje preprostih slik, znakov in animacij z nadzorom posameznih LED diod preko ustreznih električnih signalov.
        ii. Za nadzorovanje takšnega zaslona bi normalno potrebovali 16 žičk. 8 za vrstice in 8 za stolpce, zato so povezani skupaj z čipom, ki ga imenujemo premikalni register oz. shift register. Z njim lahko preko 3 priključkov nadzorujemo celotno matriko. Prepoznamo ga po vhodih DATA, CS/LATCH in CLK
    b. Razlaga joysticka
        i. Joystick ima v sebi 2 potenciometra. Eden se premika ko spreminjamo X os, drugi pa ko spreminjamo Y os. Glede na pozicijo potenciometra, bo na izhodu bila različna napetost. S pomočjo teh dveh napetosti lahko ugotovimo njegov položaj. Joystick lahko tudi pritisnemo in tako sprožimo gumbek.




    c. Nalaganje knjižnice za Led matriko: https://github.com/valmat/LedMatrix

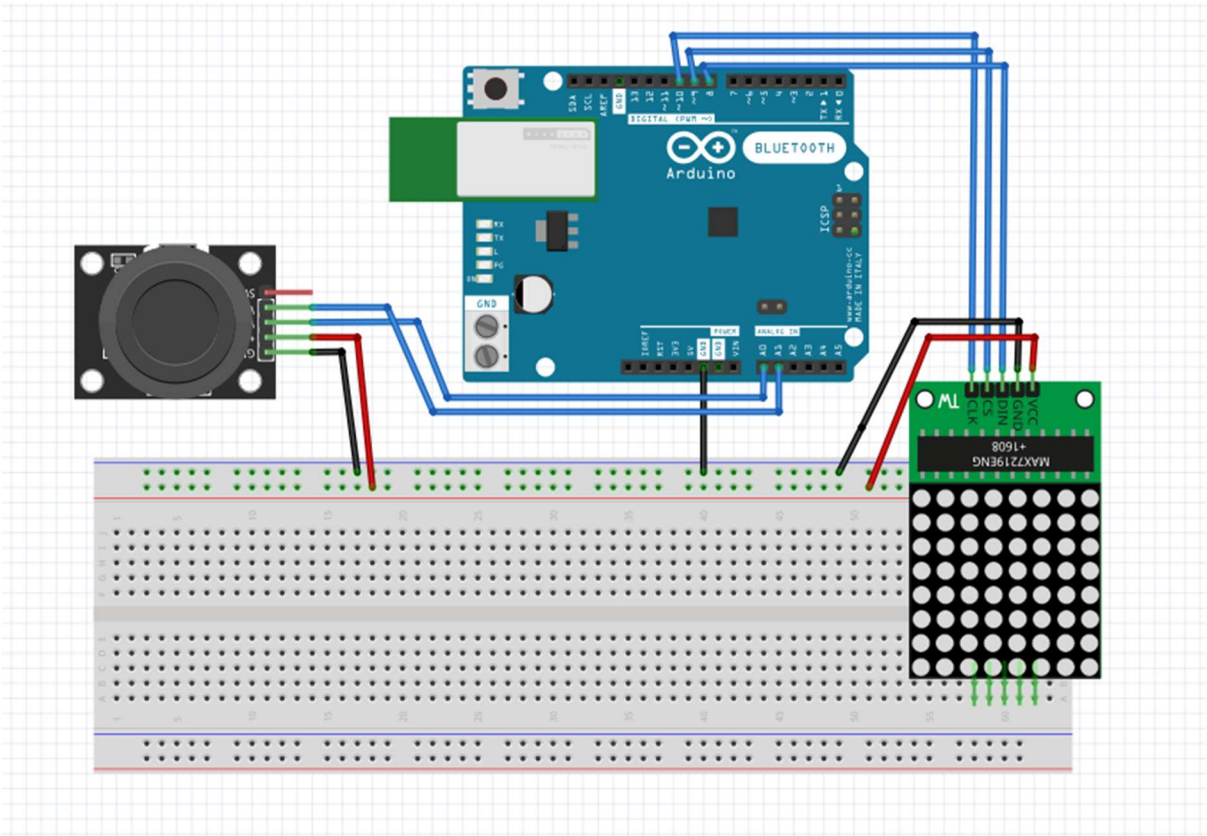


3. Pritisnemo tu




4. Pritisnemo tu in poiščem knjižnico, ki smo jo naložili (pod downloads/prenosi)

d. Sestavljanje vezja:
   i. 1x Joystick
   ii. 1x 8x8 Led matrika
   iii. 2x M:M žička
   iv. 9x M:F žička



Naš program bo spisan tako da bo joystick pravilno deloval če bodo komponente postavljene v enako smer kako na zgornji skici.

e. Pisanje programa

Pred začetkom pravega pisanja programa naj se samo napiše program, ki bi bral vrednosti iz joysticka in izpisoval vrednosti v Serial monitor.

```
const int xPin = A0, yPin = A1;
int yValue, xValue; //V tej spremenljivki bomo shranjevali vrednosti joysticka


void setup() {
  pinMode(xPin, INPUT);
  pinMode(yPin, INPUT);
  Serial.begin(9600);
}

void loop() {
  xValue=analogRead(A0);
  yValue=analogRead(A1);
  Serial.print("X: ");
  Serial.print(xValue);
  Serial.print(" ");
  Serial.print("Y: ");
  Serial.println(yValue);
  delay(1000);
}
```

Ugotovili bomo da so vrednosti na joysticku čudne oz. da sta X in Y smer zamenjani itd.. Zato bomo uporabili funckijo map(), ki vrednost prenese iz enega številskega spektra v nek nov spekter (prvotno vrne števila od 0 do 1023, mi pa bomo vse to premaknili za polovico proti negativno, tako da je sredina na 0).

```
const int xPin = A0, yPin = A1;
int yValue, xValue;

void setup() {
  pinMode(xPin, INPUT);
  pinMode(yPin, INPUT);
  Serial.begin(9600);
}

void loop() {
  xValue=map(analogRead(A0), 0, 1023, 511, -512);
  yValue=map(analogRead(A1), 0, 1023, -511, 512);
  Serial.print("X: ");
  Serial.print(xValue);
  Serial.print(" ");
  Serial.print("Y: ");
  Serial.println(yValue);
  delay(1000);
}
```

Minimalna in maximalna vrednost prvotnega spektra

Vrednost, ki jo hočemo premakniti v nov spekter

Minimalna in maximalna vrednost novega spektra

Pri testiranju joysticka smo ugotovili da je y os narobe obrnjena. Problem bomo rešili tako da bomo obrnili minimalno in maksimalno vrednost našega novega spektra

S tem novim programom bosta X in Y os pravilno naravnani, ko bo joystick postavljen na sredino bosta vrednosti X in Y približno 0 in možne bojo negativne vrednosti X in Y (kakor v koordinatnem sistemu, ko je joystick na sredini je to kakor da je v izhodišču).

Nato programu dodamo še konfiguracijo led matrike in funkcionalnost joysticka. Končna koda naj izgleda približno tako: !GLEJ TUDI NASLEDNO STRAN!

```
#include <LedMatrix.h> //Uporaba knjižnice za 8x8 led matrico


const int DIN = 8;
const int CS = 9;     //definicija pinov za led matriko
const int CLK = 10;
const int xPin = A0; //Za branje vrednosti iz joysticka bomo uporabili analogna vhoda A0 in
const int yPin = A1; A1 saj hočemo imeti vmesne vrednosti ne pa samo LOW in HIGH
int yValue;      //V te dve spremenljivki bomo shranjevali vrednost, ki
int xValue;      jo preberemo iz joysticka
int xPos = 0; //Nastavi začetno lokacijo igralca na levi spodnji
int yPos = 0; rob (prva vrstica in stolpec, saj začne šteti pri 0)
int xSad=random(1,8); //Nastavi lokacijo »sadja« na naključno lokacijo na matriki, ki na
int ySad=random(1,8); začetku nebo tista, na kateri začne igralec (0, 0) - Random(vključno,
izključno)
LedMatrix matrix(DIN, CLK, CS);  //Definicija pinov za led matriko
```

```
void setup() {
  pinMode(xPin, INPUT); //Nastavi pine za branje vrednosti joysticka na vhod
  pinMode(yPin, INPUT);
  matrix.wakeup(); //Vklopi matrico
  matrix.clear(); //Ugasne vse diode na matrici
  matrix.setIntensity(10); //Nastavi svetlost diod na 10 (1-15)
  matrix.setRotation(2); //Matriko rotiramo za 2*90 stopinj v smeri urinega kazalca, zato
                         da bo zaslon pravilno obrnjen za naš pogled (vrednost X:0 in Y:0
                         bo levo spodaj)
}

void loop() {
 if(xPos==xSad && yPos==ySad){ //Preveri če se nahajamo na enakem polju kot »sadje«
     ySad=random(0,8);         //Nastavi sadje na naključno lokacijo na matriki (med
     xSad=random(0,8);          vključno 0 in do vključno 7)
   }

 xValue=map(analogRead(A0), 0, 1023, 511, -512);  //Uporabimo analogRead, saj
 yValue=map(analogRead(A1), 0, 1023, -511, 512);  hočemo prebrati vrednosti med 0
                                                  in 1023 in ne LOW ali HIGH

 if(xValue>60 && xPos<7){          //Če joystick premaknemo za več kot 60 v
     xPos=xPos+1;                  desno in igralec ni ob desnem robu potem bo
   }                               premaknilo igralca za eno diodo v desno
 else if(xValue<-60 && xPos>0){    //Če joystick premaknemo za več kot 60 v
    xPos=xPos-1;                   levo in igralec ni ob levem robu potem bo
 }                                 premaknilo igralca za eno diodo v levo
 else if(yValue>60 && yPos<7){     //Če joystick premaknemo za več kot 60
    yPos=yPos+1;                   navzgor in igralec ni ob gornjem robu potem
 }                                 bo premaknilo igralca za eno diodo navzgor
 else if(yValue<-60 && yPos>0){      //Če joystick premaknemo za več kot navzdol
    yPos=yPos-1;                     in igralec ni ob spodnjem robu potem bo
 }                                   premaknilo igralca za eno diodo navzdol
  matrix.clear(); //Ugasne vse diode na matrici
  matrix.on(xSad, ySad); //Prižge diodo, kjer se nahaja sadje
  matrix.on(xPos, yPos); //Prižge diodo, kjer se nahaja igralec
  delay(100); //Počaka 100 milisekund
}
```

Če else if stavke spremenimo v navadne if stavke, bo to omogočilo premikanje diagonalno

5. **Ura in dalje**:
    a. Izposoja kompletov v knjižnici stvari in izdelovanje lastnega projekta
    b. IDEJE ZA PROJEKTE:
        i. Ključavnica z RFIG bralcem in alarmni sistem s senzorjem premikanja, senzorjem svetlobe…
        ii. Nadzorovanje pametnega domova s uporabo daljinca, ki oddaja infrardečo svetlobo. Vključitev raznih funkcionalnosti pametnega doma, kakor so spust in dvig rolet ob določeni svetlosti, nadzor glasnosti zvočnikov…
        iii. Zalivalni sistem
        iv. Igralna konzola z joystickom in gumbki – igra Snake!
        v. Udeležitev delavnice 3D tiska za izdelavo ohišij in nosilcev za projekte