

Georgi Babouchkine, Heikki Ketoharju, Oskari Kuusela

Viherkasvin monitorointilaite Happy Plant

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikka

Laiteraportti

10.10.2018

Sisällys

[1 Johdanto 1](#_Toc526865709)

[2 Projektin kuvaus 1](#_Toc526865710)

[3 Laitteisto 1](#_Toc526865711)

[3.1 Arduino-ympäristön historiaa 1](#_Toc526865712)

[3.2 Komponentit 1](#_Toc526865713)

[4 Työnkulku 2](#_Toc526865714)

[5 Osien toteutus 2](#_Toc526865715)

[5.1 Musiikin soittaminen 2](#_Toc526865716)

[5.2 8x8 LED-matriisin toteutus 3](#_Toc526865717)

[5.3 Datan esittäminen OLED-näytöllä 4](#_Toc526865718)

[5.4 Muistiin tallentaminen 5](#_Toc526865719)

[6 Haasteet 5](#_Toc526865720)

[6.1 8x8 ledin matriisimoduuli 5](#_Toc526865721)

[6.2 Haasteita ohjelmointikielen kanssa 6](#_Toc526865722)

[6.3 Muistin käyttö ylärajoilla 7](#_Toc526865723)

[7 Johtopäätökset 7](#_Toc526865724)

[8 Lähdeluettelo 7](#_Toc526865725)

# Johdanto

Projektin tavoitteena oli opetella laiteläheistä ohjelmointia avointa Arduino-mikrokontrolleria ja C++-ohjelmointikieltä käyttäen.

Aiheeksi valittiin kasvin mullankosteuden ja ilman lämpötilan seuranta. Projektin aihevalintaan haettiin inspiraatiota muun muassa Arduinon Playground -sivustolta ja harrastajien blogeista. Osa ryhmän jäsenistä harrastaa chilien ja muiden viherkasvien kasvatusta, joten viherkasvin hyvinvoinnin seurantalaite oli sopiva valinta projektin aiheeksi.

# Projektin kuvaus

Projekti Happy Plant on laite, joka mittaa mullankosteutta ja ympäristön lämpötilaa sekä tallentaa luettuja arvoja ja näyttää niitä OLED-näytöllä sekä LED-matriisimoduulin avulla.

# Laitteisto

## Arduino-ympäristön historiaa

Arduino on avoimeen laitteistoon perustuva elektroniikka-alusta ja ohjelmointiympäristö, joka on kehitetty Italiassa vuonna 2003. Arduino eroaa muista elektroniikka-alustoista siten, että se on julkaistu avoimella lisenssillä. Arduinoa valmistava yritys ei pidä hallussaan patentteja tai suljettua lähdekoodia. Sekä lauta että ohjelmointiympäristö kirjastoineen ovat avointa lähdekoodia.

Avoin ympäristö on innostanut harrastajia ja yrityksiä ympäri maailman kehittämään työkaluja, ohjelmointikirjastoja ja komponentteja ympäristöön. Tämän vuoksi Arduino-pohjalle voi rakentaa lähes mitä hyvänsä. Internetistä löytyy paljon ohjeita, kytkentäkaavioita ja koodiesimerkkejä erilaisiin projekteihin.

## Komponentit

Projektissa käytettiin Arduino Uno -mikrokontrolleria, kapasitiivista mullankosteusanturia, lämpöanturia, kahta painonappia ja OLED-näyttöä sekä 8x8 LED-matriisin ohjausmoduulia. Useimmat osat löytyivät valmiiksi koululta, mutta projektia varten päädyttiin tilaamaan laadukkaampi ja luotettavampi kosteusanturi Amazon.de -verkkokaupasta. LED-matriisin ohjausmoduuli hankittiin elektroniikkaliike Partcosta.

# Työnkulku

Projektin aluksi sovittiin työnjako ja projektin toteutusaikataulu. Alkuperäinen suunnitelma oli, että projekti jaetaan kolmeen eri osaan: näytölle piirtäminen, musiikin soittaminen ja anturien lukeminen. Jokainen ryhmän jäsen otti yhden osan suorittaakseen.

Alkuperäisessä suunnitelmassa laitteeseen kuului LED-matriisi ja mullankosteusanturi. Projektin edetessä päätettiin lisätä laitteeseen vielä OLED-näyttö mitattujen arvojen esittämiseen ja tallentaa mittaustulokset Arduino Uno -alustassa olevaan EEPROM-muistiin (Electronically Erasable Programmable Read-Only Memory).

# Osien toteutus

## Mullankosteuden mittaus

Mullan kosteutta mitattiin kapasitiivisella mullankosteusanturilla. Anturi mittaa kosteuden muutoksesta aiheutuvaa kapasitanssin muutosta, kun vesimolekyylit imeytyvät ohueen eristekalvoon. Kosteuden arvot anturilla olivat noin 250-600, pienimmän arvon ollessa märkä ja suurimman kuiva. [1.] Anturi liitettiin mikrokontrollerialustan analogiseen lähtöpinniin A1 ja virta- sekä maadoituspinneihin.

Kapasitiivista kosteusanturia päädyttiin käyttämään koululta löytyvän resistiivisen anturin sijaan, koska kapasitiivinen kosteusanturi ei ole suoraan kosketuksissa kosteuden kanssa ja näin sen käyttöikä on huomattavasti pidempi. Resistiivisen kosteusanturin suurin vika oli veden kanssa tapahtuva korroosio, mikä olisi voinut vahingoittaa kasvia.

## Musiikin soittaminen

Projektin tavoitteena oli rakentaa laite, joka antaa paitsi visuaalista palautetta, myös soittaisi musiikkia, kun kasvin tila muuttuu. Päädyimme ratkaisuun, jossa kasvi ”laulaa”, kun sitä kastellaan.

Musiikin soittaminen toteutettiin kytkemällä laitteistoon Piezo-kaiutin, jota ohjattiin sähköjännitteellä Arduino-laudan pinnistä 9.

Kaiuttimesta lähtevän äänen aloittavat ja lopettavat käskyt on annettava juuri oikealla hetkellä. Mikroprosessori kuitenkin lukee koodia ylhäältä alas ja suorittaa käskyjä järjestyksessä. Laitteessa oli useita eri toimintoja ja jotta musiikkia soittavat käskyt saatiin ajastettua täsmällisesti, käytettiin mikrokontrollerin sisäänrakennettua laitteistokeskeytystä. Arduinossa on ajastin, jolle voidaan asettaa määrätty väliaika, jonka jälkeen ajastin lähettää keskeytyksen. Ohjelmakoodiin voidaan määritellä käskyt, jotka suoritetaan keskeytyksen aikana. [2.]

Laitteeseen ohjelmoitiin neljän millisekunnin välein suoritettava keskeytys, joka tarkistaa, pitääkö soittaa seuraava nuotti vai sammuttaa edellinen.

Ohjelmaan on mahdollista määritellä muistin sallimissa rajoissa rajaton määrä melodioita, joita voidaan soittaa vaihtelevilla tempoilla. Melodiat tallennetaan Arduinon dynaamiseen muistiin tavutaulukoina. Sävelet on kuvattu musiikissa yleisesti käytetyn MIDI-standardin nuottiarvoina (Musical Instrument Digital Interface, musiikkisoittimen digitaalinen rajapinta).

Arduino-ohjelma tekee muunnoksen MIDI-arvosta värähtelytaajuuteen ja sävelet soitetaan Arduinon sisäänrakennettua Tone-käskyä käyttäen.

## 8x8 LED-matriisin toteutus

Alun perin käytössä ollut LED-matriisinäyttö tarvitsi käyttöönsä kaikki Arduinon Unon 14 digitaalista kytkentäpinniä. Tarvittavaa pinnien määrää saatiin vähennettyä ottamalla käyttöön kaksi siirtorekisteriä, josta ensimmäinen on kytketty suoraan ja toinen vastuksien kautta LED-matriisiin. Siirtorekisterien kytkentä on mahdollista tehdä itse, mutta kytkentöjen selvyyden vuoksi päätettiin hankkia LED-matriisinäyttö, joka sisältää valmiiksi tarvittavat komponentit. Näin tarvittavien pinnipaikkojen määrä saatiin vähennettyä yhteensä viiteen.

MAX7219 8x8 LED-matriisikomponentti koostuu itse LED-matriisista ja siihen kolvatusta levystä, joka lukee matriisin rivejä ja sarakkeita. Tieto kulkee siirtorekisteriin (Shift Register) ja sieltä kolmeen pinniin Arduinossa. Projektissamme LED-matriisi käyttää pinnejä 10, 11 ja 12 [3.]

LED-matriisille piirtävä rutiini ottaa syötteeksi binaarimuotoisia kuvia ja piirtää niitä drawScreen()-funktion avulla.

**const** byte HAPPY[] = {  
  
 B00000000,  
 B00100100,  
 B00100100,  
 B00100100,  
 B00000000,  
 B01000010,  
 B01000010,  
 B00111100  
};

1. Yksi neljästä hymiöstä binaarimuodossa.

Koodista voidaan lukea, miten kuva piirretään LED-matriisiin. Jokainen ykkönen binaariluvussa vastaa palavaa valodiodia. Binaariluvut tallennetaan vakiotaulukkoon koodisilmukan ulkopuolelle.

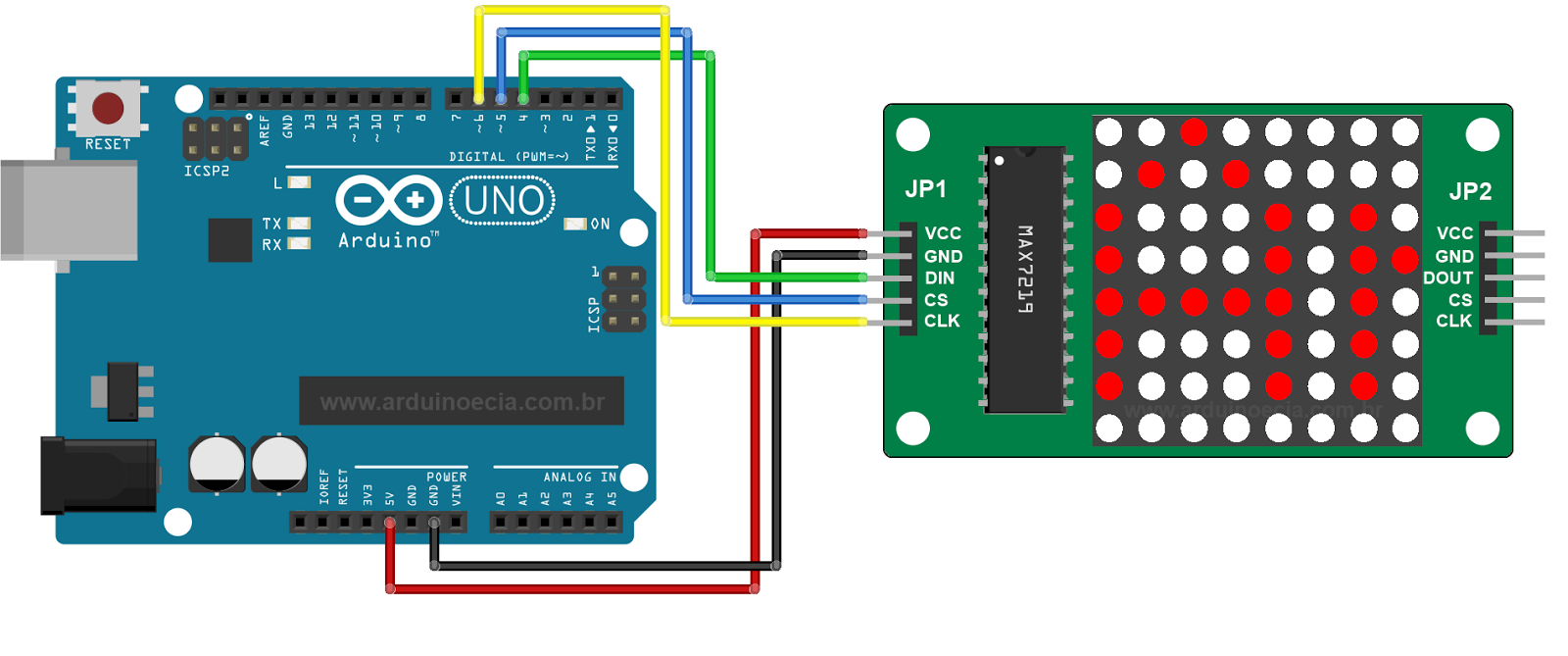
// function that draws smileys  
**void**  drawScreen(byte buffer2[ ])  
{  
  // Turn on each row in series  
  **for** (byte i = 0; i < 8; i++)        // count next row  
  {  
    **for** (byte a = 0; a < 8; a++)    // count next row  
    {  
      display.setLed(0, i, a, bitRead(buffer2[i], 7 - a));  
    }

1. DrawScreen-funktio käyttää LedMatrix-kirjaston setLed-komentoa ledien sytyttämiseen ja sammuttamiseen.

Koodisilmukasta kutsutaan piirtofunktiota, kun ehdot tietyn kuvan näyttämiseksi täyttyvät. Eli kun kosteusanturi saa koodissa määritellyn hyväksytyn arvon, LED-näyttö näyttää hymynaaman.

/\* Led Screen \*/  
  
  **if** (soilSensor() < MaxMoistureEmoji)  
  {  
    drawScreen(DED);  
  }  
  **else** **if** (soilSensor() >= MaxMoistureEmoji && soilSensor() < OptimalHighEmoji)  
  {  
    drawScreen(NEUTRAL);  
  }

1. Tarkistetaan, mikä on mullankosteusarvo ja sen perusteella mikä tulostettava hymiö valitaan.



1. LED-matriisi, joka on kytketty kolmeen digitaaliseen tulopinniin (4, 5, 6) ja virta- ja maadoituspinneihin (5V, GND).

/\* PINS for 8x8 Matrix \*/  
**const** **int** DIN\_PIN = 12;  
**const** **int** CLK\_PIN = 11;  
**const** **int** CS\_PIN = 10;

1. Pinnipaikkojen asetus koodissa LED-matriisia varten.

Itse idea kuvan tallentamiseen binaarimuodossa saatiin LED Matrix Editor -työkalusta [4]. Työkalun tarkoitus on yksinkertaistaa ja havainnollistaa kuvien tekoa LED-matriisille. Työkalu myös tarjoaa esimerkkikoodia ja ohjeet animaatioiden teolle.

## Datan esittäminen OLED-näytöllä

Lämpötila- ja kosteusanturin tuottamat arvot päätettiin esittää yhden tuuman OLED-näytöllä. Näytön tarkkuus oli 128x32 pikseliä, joten näytöllä pystyi esittämään tekstin lisäksi yksinkertaisia kuvioita. Näytölle tulostettiin reaaliaikaisesti senhetkinen huoneen lämpötila ja mullankosteusarvo. Lämpötila esitettiin celsiusasteina ja kosteusarvo pisarasymboleina. Laitteeseen lisättiin painonappi, jota painamalla näytölle tulostettiin muistiin tallennettujen lämpö- ja kosteusarvojen vuorokauden ja viikon keskiarvot.

Näyttö tarvitsi toimiakseen yksivärisille OLED-näytöille tehdyn u8g2-kirjaston, joka mahdollisti monien eri fonttien ja symbolien käytön. U8g2-kirjasto on yksi monista Arduinon avoimen lähdekoodin kirjastoista, joka on vapaasti ladattavissa Arduino IDE:n (Arduinon ohjelmointiympäristö) kautta.

## Muistiin tallentaminen

Mitatut arvot päätettiin tallentaa laitteen EEPROM-muistiin. Arduino Unossa on sisäänrakennettuna 1024 tavun kokoinen muisti, jonka sisältö säilyy, vaikka laitteesta katkaistaisiin virta. [5.]

Suurin haaste muistiin tallentamisessa oli tallennettavan tiedon muuntaminen tavumuotoon. Tavumuotoinen muuttuja pystyy tallentamaan numeroarvot vain väliltä 0-255, joten sekä lämpötilamittaukset että kosteusmittarin tulokset piti muuttaa tähän asteikkoon sopiviksi. Muunnosprosessissa anturien tuottaman datan tarkkuudesta osa hävisi, mutta tällä ei ollut projektin kannalta merkitystä.

Laitteeseen ohjelmoitiin keskiarvotoiminto, joka tulostaa näytölle päivittäiset ja viikoittaiset keskiarvot kosteudesta sekä lämpötilasta. Laitteeseen tehtiin myös painonappi, jota painamalla kaiken tallennetun datan sai poistettua ja tallentamisen aloitettua uudestaan tyhjään muistiin [6.]

# Haasteet

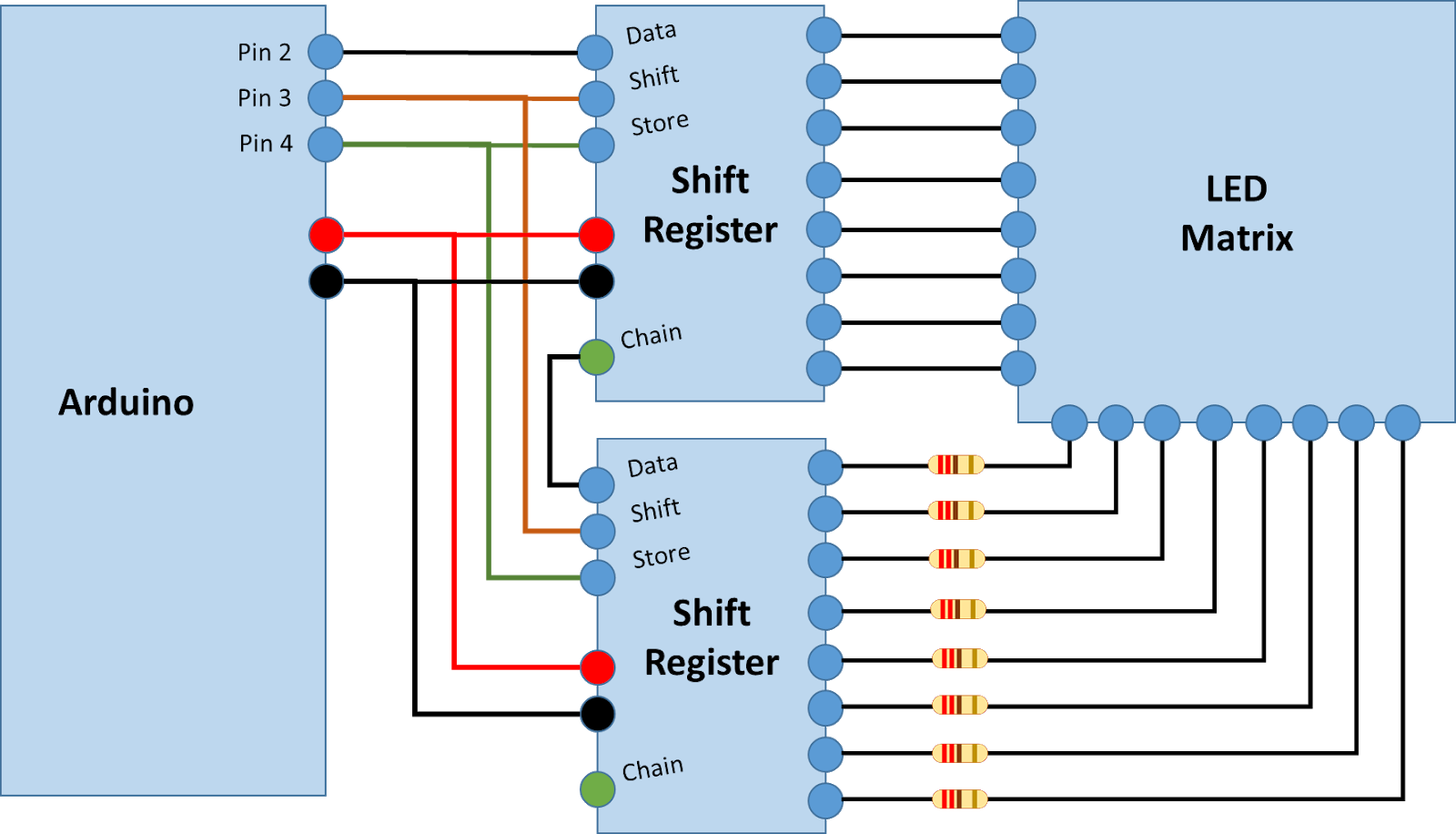
Haasteeksi osoittautui ohjelman kirjoittaminen siten, että jokainen komponentti pystyy toimimaan oikea-aikaisesti, eikä mikään järjestelmän osa käytä kaikkea suoritusaikaa. Esimerkiksi alkuperäisessä ratkaisussa lämpötila- ja kosteusarvojen tulostaminen näytölle keskeytyi, kun kaiutin antoi äänimerkin.

Ongelma ratkaistiin käyttämällä Arduinon sisäisiä ajastimia ja jakamalla näin suoritusaikaa vuorotellen eri komponenteille.

Tämä toteutustapa inspiroi myös toteuttamaan OLED-näytölle suppean tilakoneen, jonka avulla näytön sisältöä pystyy vaihtamaan joustavasti, esimerkiksi tietyn aikarajan myötä, tai nappia painamalla.

## 8x8 ledin matriisimoduuli

LED-matriisin ajurin rakentaminen osoittautui haasteelliseksi. Itse rakennetut kytkennät saivat LED-näytölle valodiodeja satunnaisesti päälle, vaikka koodissa oli määritelty kaikki valodiodit sytytettäviksi. Huomattiin, että ongelma oli johdotuksissa. Vaikka käytettävien pinnien määrä oli saatu vähennettyä kolmeen, vaati siirtorekisterien itse kytkeminen huomattavasti enemmän johtoja, kuin mitä alkuperäinen suunnitelma olisi tarvinnut.



1. NERDfirst-blogista otettu suunnitelma Led Matriisille.

Kuvassa on yksinkertaistettu ohje kytkennälle. Alkuperäinen kytkentäkaavio oli huomattavasti monimutkaisempi. Oman haasteensa toi myös huonosti paikallaan pysyvät johdot, joten jokainen johto lisäsi ongelma- ja häiriöriskiä. Lopulta päätettiin, että on parempi käyttää LED-matriisia, jossa on siirtorekisterit valmiiksi liitettynä ja käyttää säästetty aika itse ohjelmiston kirjoittamiseen.

## Haasteita ohjelmointikielen kanssa

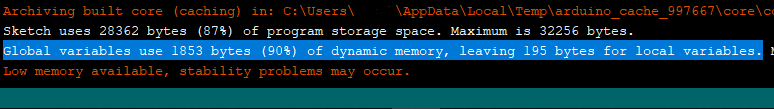
Projekti kirjoitettiin käyttäen Arduinon C++-kieleen perustuvaa ohjelmointikieltä. Lähes kaikille ryhmän jäsenille kieli oli entuudestaan täysin tuntematon ja projektin tekeminen sisälsi paljon itse kielen opiskelua. Koska C++ on matalan tason ohjelmointikieli, jotkut sen ominaisuudet osoittautuivat haastaviksi käyttää. Kieli on vahvasti tyypitelty ja sen muuttujat ovat suurimmaksi osaksi numeromuotoisia. Korkeamman tason ohjelmointikielistä tutut joustavat merkkijonojen käsittelytavat ja automaattiset tyyppimuunnokset puuttuvat täysin.

Myös taulukoiden käyttö vaatii ohjelmoijalta tarkkuutta, sillä C++ antaa lukea muistia taulukoiden ulkopuolellekin. Ohjelmalle pitää myös erikseen kertoa, jos aliohjelmalle halutaan toimittaa pelkkä viittaus muuttujaan varsinaisen muuttujan sijasta.

Projektin aikana jouduttiin harjoittelemaan muuttujien tyyppimuunnoksia. Lisäksi huomattiin, että C++:ssa funktiot pitää määritellä ennen niiden käyttöä. Yleinen tapa onkin lisätä funktioiden yhden rivin mittainen esittely koodin alkuun #define-komennolla, ja määritellä varsinainen aliohjelma vasta pääohjelman jälkeen.

## Muistin käyttö ylärajoilla

Arduino Uno piirilevyssä on 2048 tavua dynaamista muistia ja 32256 tavua sisäistä muistia. Projektin kuluessa havaittiin, että ulkopuoliset kirjastot vievät paljon tilaa dynaamisesta muistista. Laitteen toiminnan vakaus laskee sitä mukaa, mitä vähemmän dynaamista muistia on jäljellä. Ulkopuolisten kirjastojen käyttöä pyrittiin pitämään minimissä. Lisäksi musiikkijärjestelmää optimoitiin niin, että laskutoimituksia tehdään lennossa pysyvän muistin varaamisen sijasta.



1. Arduino iloisesti ilmoittaa viimeisellä rivillä, että ongelmia voi tulla.

# Yhteenveto

Arduinolla toteutetun Happy Plant -laitteen rakentaminen ja ohjelmoiminen opetti paljon C++-kielen toiminnasta, laitteiden rakentamisesta ja eri komponenttien toiminnasta. Projektia tehdessä opittiin myös ryhmätyötaitoja ja projektisuunnitelman tekoa sekä aikataulutusta. Jokaisen ryhmänjäsenen mielestä sekä projektin että kurssin aihealue oli mielenkiintoinen ja osa päätyikin ostamaan omaksi Arduino aloituspakkauksen. Valmis laite täytti asetetut vaatimukset ja saatiin toimintakuntoon määräaikaan mennessä. Kurssista jäi käteen hyvin suunniteltu ja toteutettu laite sekä uutta hyödyllistä osaamista.

Lähdeluettelo

Capacitive Soil Moisture Sensor SKU:SEN0193. 2017. Verkkoaineisto. DFRobot. <https://www.dfrobot.com/wiki/index.php/Capacitive\_Soil\_Moisture\_Sensor\_SKU:SEN0193#FAQ>. 25.5.2017. Luettu 17.9.2018

Arduino timer interrupts. 20xx. Verkkoaineisto. Amanda Ghassaei. <https://www.instructables.com/id/Arduino-Timer-Interrupts/>. Luettu 27.9.2018.

Shift Registers with LED Matrix. 2017. Verkkoaineisto. NERDfirst. <www.csc.fi/lehdet/atcs/atcsc98/html>. 6.9.2017. Luettu 5.10.2018.

LED Matrix Editor. 2017. Verkkoaineisto. Xantorohara. <https://xantorohara.github.io/led-matrix-editor/>. 9.9.2017. Luettu 9.10.2018.

Using EEPROM to Store Data on the Arduino. 2017. Verkkoaineisto. Mads Aasvik. <https://www.norwegiancreations.com/2017/02/using-eeprom-to-store-data-on-the-arduino/>. 17.2.2017. Luettu 1.10.2018.

Tutorial: Arduino and Buttons. Verkkoaineisto. Mads Aasvik. <https://www.norwegiancreations.com/2015/11/tutorial-arduino-and-buttons/>. 12.10.2015. Luettu 1.10.2018.