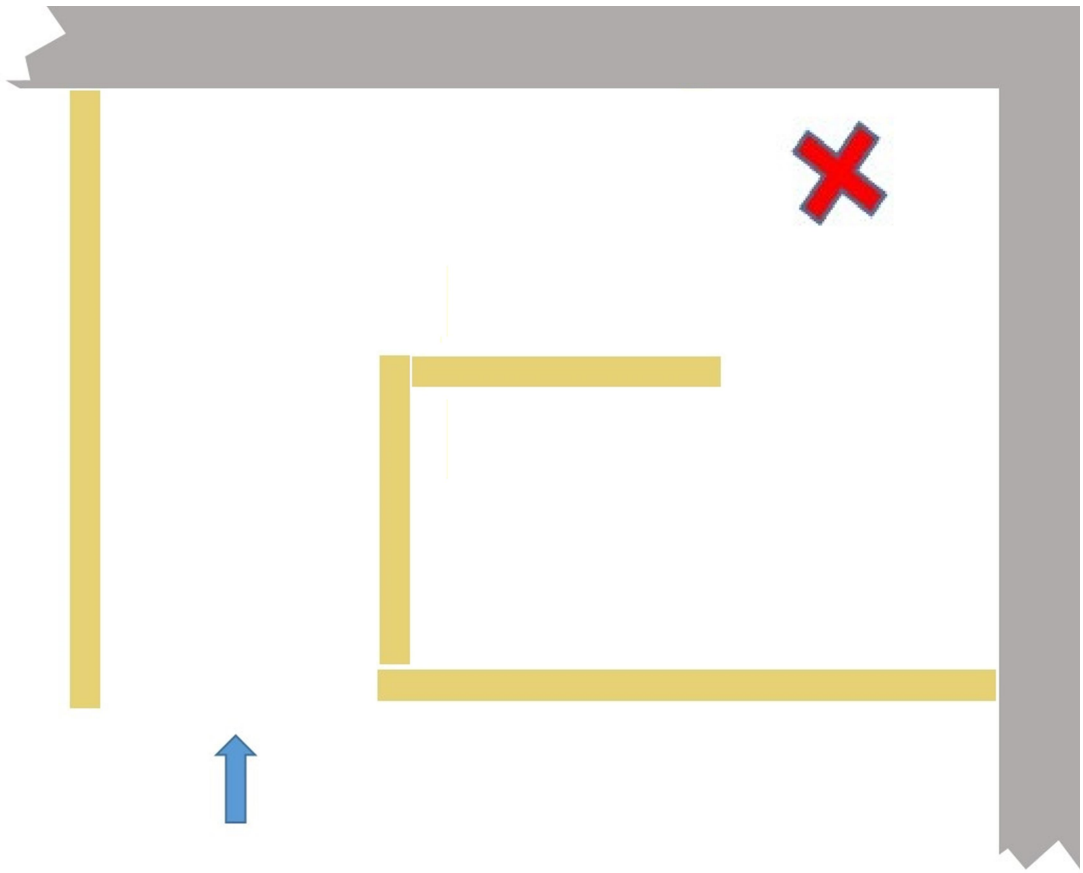
Tehtävä 23. Ohjelmoidaan robotti kulkemaan labyrintissa ja etsimään sieltä kohde

Tutustutaan opetustilaan rakennettuun labyrinttiin. Tehtävänä on ohjelmoida robotti löytämään merkitty kohde, kertomaan siitä äänimerkillä, sekä löytämään tiensä takaisin ulos labyrintista. Aikaa ei ole rajoitettu. Tehtävään käytetään aikaisemmista harjoituksista tuttua perusrobottia.



(kuva 5/5) Labyrintti, kohde voi sijaita missä vain ja sen paikkaa voidaan vaihtaa..

Taktiikka

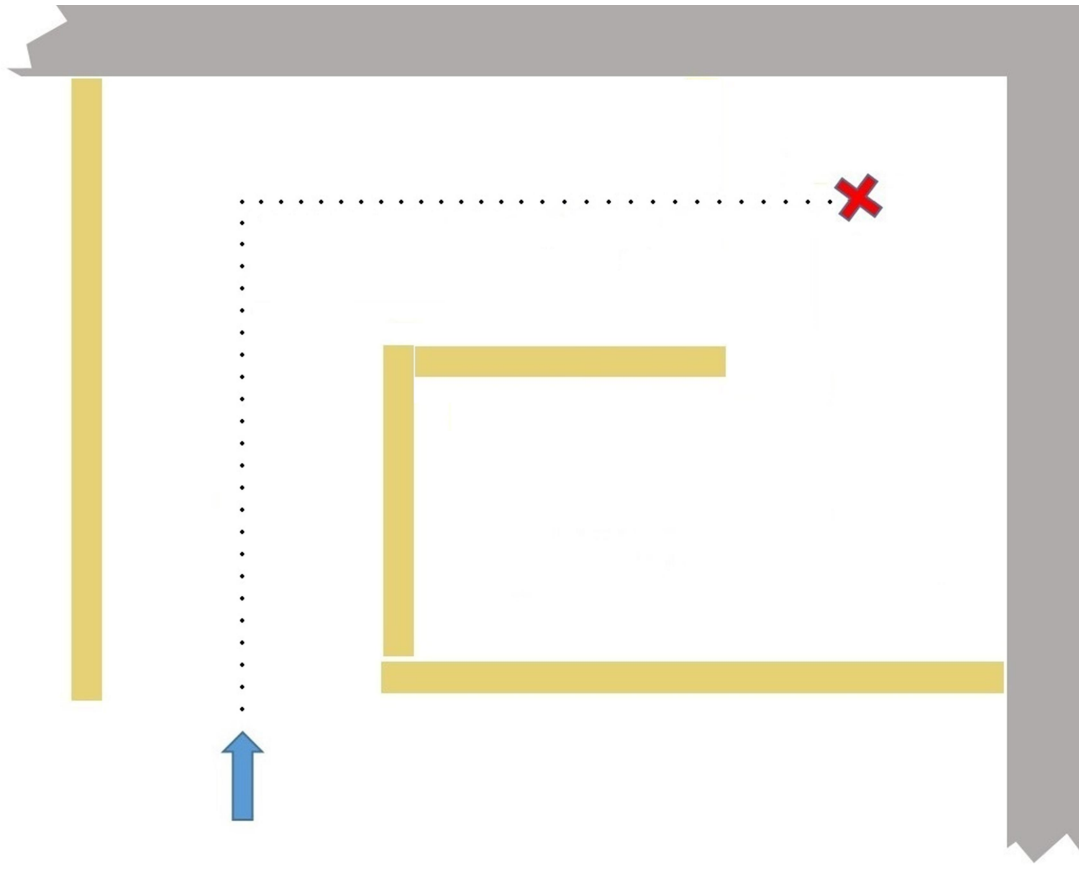
Tässä tehtävässä kuvaan astuu taktiikka. Etsittävä kohde on punaisella teipillä merkitty, halkaisjaltaan noin 15cm suuruinen X-merkki. Merkki saadaan tunnistettua valokennolla, mutta se pitää ensin löytää.

Vaihtoehtoja labyrintissa suunnistamiseen on monia, esim.

1. Ennalta ohjelmoitu reitti, näitä olemme jo harjoitelleet. Tämän heikkoutena on sen herkkyys pienimmillekin ohjelmoinnin yms. virheille, mukaanlukien radan poikkeamat. Sitä kannattaa silti kokeilla.

   Periaate on ajaa ensin eteenpäin, kääntyä, ajaa, kääntyä jne. kunnes on päästy perille. Ja sitten samaa reittiä peruuttaen takaisin.

   Robotin liikkumista ohjaavan ohjelman osan rinnalle haaroitetaan (polkua käyttäen) itsenäinen ohjelman haara, joka tarkkailee sattuisiko valokennon eteen punaista kohdetta ja antaa äänimerkin niin sattuessa.



(kuva 6/5) Vaihtoehto 1, Ennalta määritelty reitti

(kuva 7/5) Pysäköintitehtävän ”sivulle katsova” robotti soveltuu labyrintti-tehtäviin. Huomaa miten robotin pyörähtäessä ympäri aivan paikallaan, tämän pyörähdyksen keskipiste on käynnistyspainikkeen kohdalla. Takaosa on lyhyt, mikä helpottaa seinän kähellä kääntymistä. Etäämmällä edessä olevilla antureillaan robotti saa tarkastettua ison alueen tekemällä pienen käännöksen.

Kuvitellaan lisäksi, että näemme ovelta vain käytävään ja emme voi tietää mihin suuntaan ja miten pitkälle käytävä jatkuu kulman takana.

2. Seuraavana vaihtoehtona on lähteä kulkemaan seuraamalla seinää. Ensin kuljetaan lähellä seinää, ja jos ei tärppää ensimmäisellä yrittämällä, kuljetaan toisella kertaa hiukan etäämmällä. Tämä on varmatoimisempi kuin ensimmäinen vaihtoehto, mutta silti nopea.

Lähdetään tekemään vaihtoehdon 2 –mukaista ohjelmaa. Mietitään ensin mitä robotin tulisi tehdä, jotta se löytäisi labyrintin perukoille ja takaisin. Kuten oikeassa ohjelmointityössä, tässäkin tehtävä kannattaa ensin purkaa pieniin osiin, joita on sitten helpompi ratkoa yksitellen.

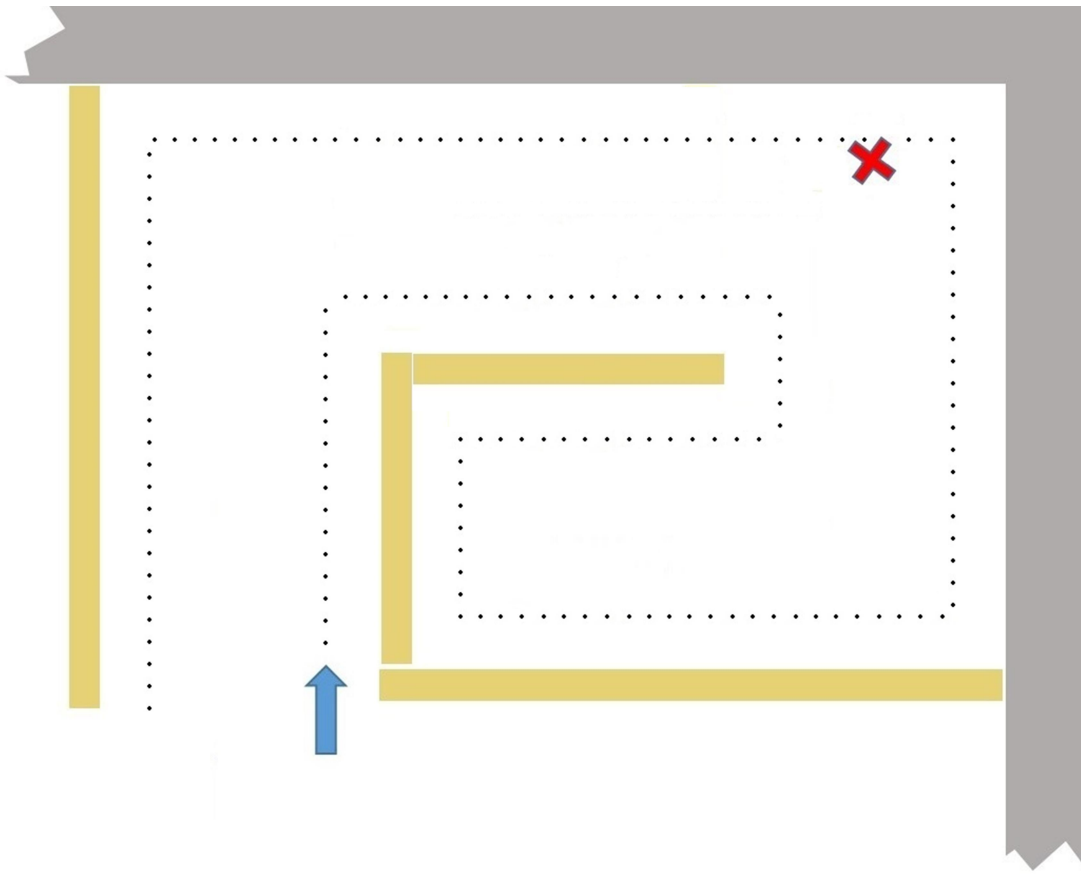
Alla on kuva reitistä, jota lähdetään nyt yrittämään. Ultraäänianturin tuottaman tiedon avulla voidaan seurata seinää halutulla etäisyydellä, ohjelmoidaan kuten viivanseuranta.

Robotti kulkee ensimmäisen puoli metriä, kunnes se sivuuttaa laudan pään. Tässä on ensimmäinen ongelmakohta (ulkonurkka), miten tietää kuinka kauas voi ajaa, missä kohdassa pitää kääntyä, miten paljon (90°/180°) ja mitä tehdään seuraavaksi.

Seuraava pulma onkin kääntyminen vasemmalle (sisänurkka). Sitäkin varten meidän pitäisi tietää etukäten kuinka pitkälle voidaan vielä ajaa. Siis aina, ennen kuin robotti ajaa ainuttakaan suoraa osuutta, sen pitää ensin mitata paljonko sillä on edessään tilaa edetä.

Montako ulko- ja sisänurkkaa tässä labyrintissa on?

Miten robotti voisi mitata paljonko sillä on edessään tilaa kulkea?



(kuva 8/5) Labyrintti, kohde voi sijaita missä vain ja sen paikkaa voidaan vaihtaa.

Tehtävä on yksinkertainen, mutta se ei tee siitä helppoa.

Piirtäminen auttaa usein ohjelmointiin liittyvien pulmien ratkomisessa. Vielä tehokkaammaksi on osoittautunut käytännön kenttätyö. Katsotaan millainen se kone, laite tai robotti on ihan oikeasti ja miten se istuu sen oikeaan käyttöympäristöön. Eli, viedään robotti labyrinttiin ja pyöritellään sitä sielä ihan käsin.

> Apuna kannattaa käyttää edellisessä harjoituksessa tehtyä ohjelmaa, joka näyttää ultraäänianturin mittaaman etäisyyden ohjaimen näyttöruudulla.

- Etsitään yhdessä kokeilemalla paikkoja missä edessä olevaa vapaata tilaa pitää ja voi mitata.

a. Mietitään yhdessä miten robotti voisi toimia sisänurkissa, mistä ja miten se voi tietää milloin sen pitää pysähtyä.

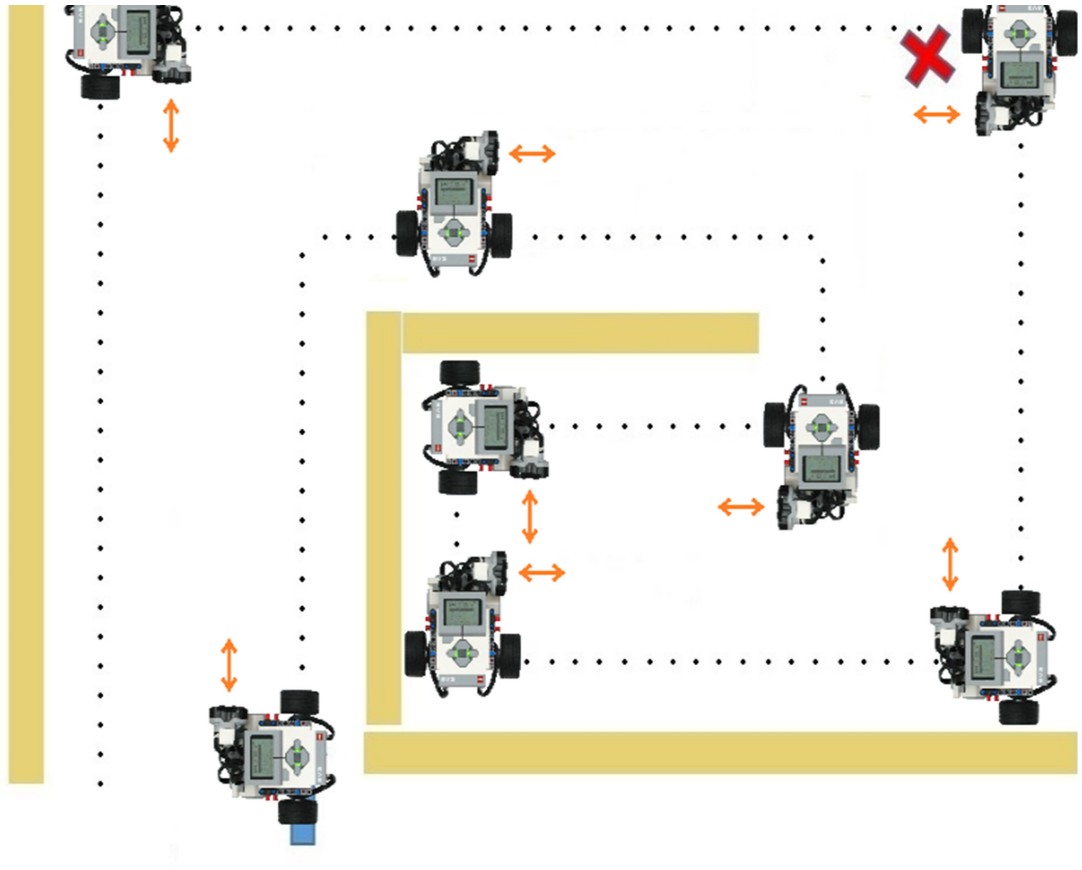
b. Entä ulkonurkissa?

a. Mistä ja miten robotti voi päätellä, mihin suuntaan sen tulee seuraavaksi kääntyä?

b. Miten robotin tulee toimia seinän päätyä kiertäessään?

Samalla pidetään mielessä, että ohjelmoinnin ja ohjelman toiminnan kannalta on edullista, jos ohjelman osia voidaan käyttää sellaisenaan mahdollisimman monessa paikassa.

Luonnos niistä kohdista, joissa robotin tulisi mitata paljonko sillä on edessään tilaa kulkea.



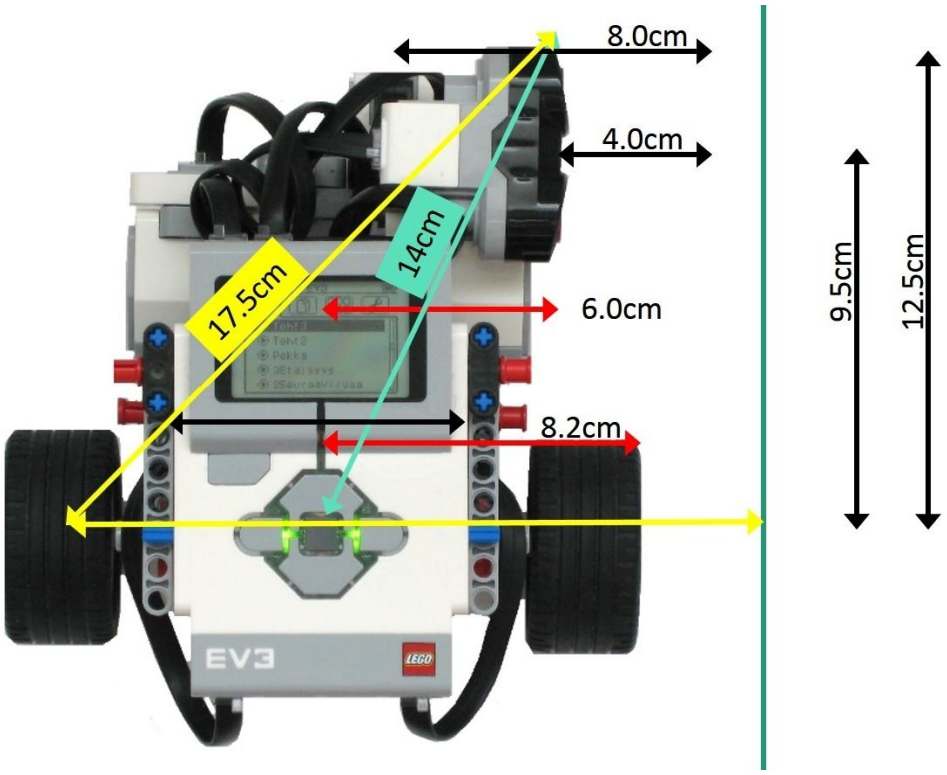
(kuva 9/5)

Harjoituksen labyrinttiin sisältyy

- käännös oikealle,
- tuplakäännös oikealle,
- tuplakäännös vasemmalle,
- kolme tavallista käännöstä vasemmalle
- ja eri mittaisia suoraan ajettavia osuuksia.

Ohjelmoijan pitää osata matematiikkaa

Hyvin pian voi huomata, että tämän tehtävän ratkominen edellyttää muutamien robotin perus mittojen tuntemista, sekä pientä ynnäilyä.



(kuva 10/5) Joitakin robotin mittoja, joita saatat tarvita

Lähtökohdaksi on otettu etsittävän kohteen koko, 10cm. Tältä pohjalta robotin tulisi kyetä kulkemaan alkaen 5cm etäisyydellä seinästä.

Luokkaan voidaan tehdä samanlainen, mutta isompi labyrintti pöydistä. Kokeilkaa kävelemällä, löytäisikö robotti oikean reitin näillä ohjeilla

a. Jos oikealla puolellasi on labyrintin ”seinä”, käänny vasemmalle.
b. Arvioi, paljonko oikealla puolellasi on nyt matkaa lähimpään ”seinään”.
c. Käänny oikealle.
d. Nollaa matkamittari (laskuri) ja lähde seuraamaan ”seinää”.
e. Jos kuljettava matka tulee täyteen, pysähdy ja käänny vasemmalle, tai
f. jos seurattava seinä loppuu, käänny oikealle ja kulje vielä pieni matka.
g. aloita uudestaan kohdasta a.

Oikean robotin ohjelman toimintakuvaus poikkeaa tästä vain vähän. Perus periaate jolla robotti toimii on aivan sama. Koneille pitää kuitenkin kertoa joitakin asioita vähän tarkemmin, kuin meille ihmisille. Kun esimerkiksi joku pyytää meitä kääntymään vasemmalle, me oletamme pyynnön tarkoittavan 90 asteen käännöstä paikallaan. Koneelle meidän pitää kertoa kaikki sen tekemisiin vaikuttavat tiedot pienimpiä yksityiskohtia myöten ja vieläpä hyvin tarkasti.

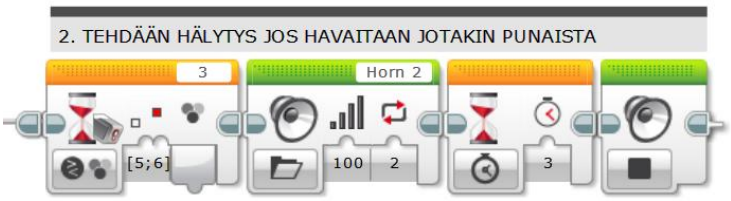
Ihmiset osaavat myös arvioida omien havaintojensa luotettavuutta. Koneet, kuten robotit, sen sijaan joutuvat luottamaan anturiensa tuottamaan tietoon sokeasti, ellei tähän ole puututtu ohjelmoinnin keinoin.

Labyrintti-robotin toimintakuvaus

1. *Alkurutiinit*, ohjelmoitiinpa mitä tahansa, ylensä ohjelman alkuun joudutaan kirjoittamaan jotakin, että ohjelma lähtee jouhevasti liikkeelle. Muuttujille määritellään arvoja, antureita ja laskureita nollataan. Samalla voidaan tyhjentää näyttöruutu ja kirjoittaa sille sellaiset tekstit, joita ei tulla muuttamaan.

(kuva 11/5)

2. Seuraavaksi voidaan kirjoittaa etsittävän kohteen tunnistava ja tästä hälyttävä ohjelman pätkä.

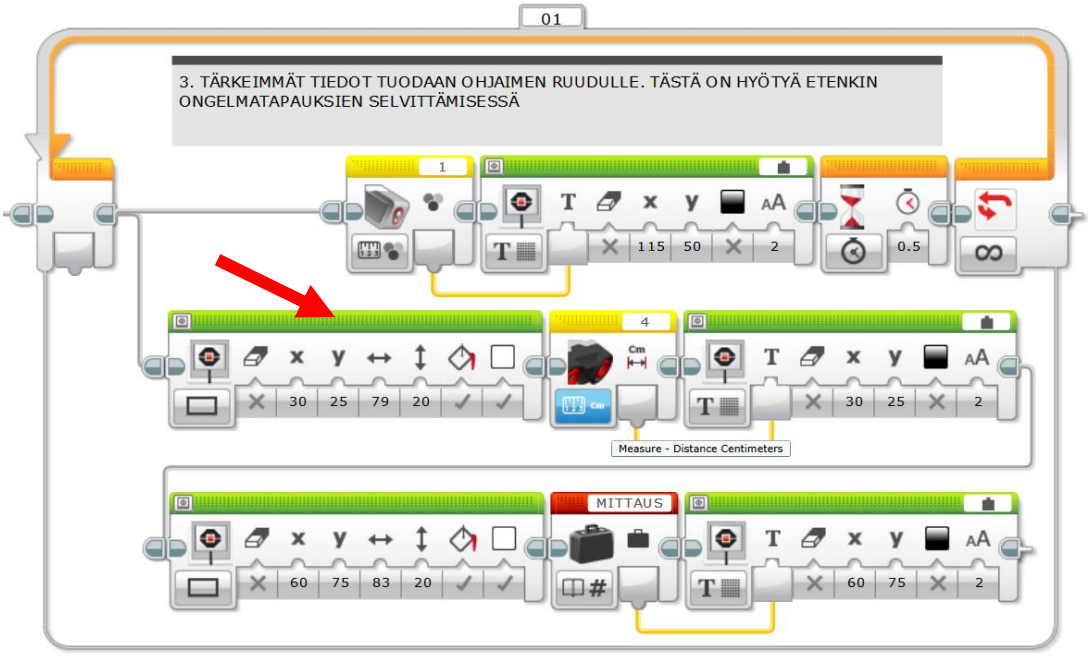


(kuva 12/5)

3. Täydennetään näyttöruudun ohjausta muuttuvilla tiedoilla.
   - Ylimmällä rivillä on valokennon mittaaman tiedon esityksen määrittelyt
   - Keskimmäisellä rivillä sama ultraäänianturin osalta.
   - ja alimmalla ajettavissa olevan matkan pituus.

Joukossa on myös näytönohjaustoimilohkoja, joissa on maaliämpärin näköinen kuva. Mitä luulisit niden tekevän?

Kun olet kirjoittanut kuvassa näkyvän osan ohjelmaa, voit ladata sen robottiin ja katsoa että mittaukset tulevat oikein ruudulle. Ota nyt kokeeksi nuolella merkitty näytönohjaustoimilohko pois ohjelmasta ja kokeile sitä sitten uudestaan. Joko keksit tämän toimilohkon merkityksen?



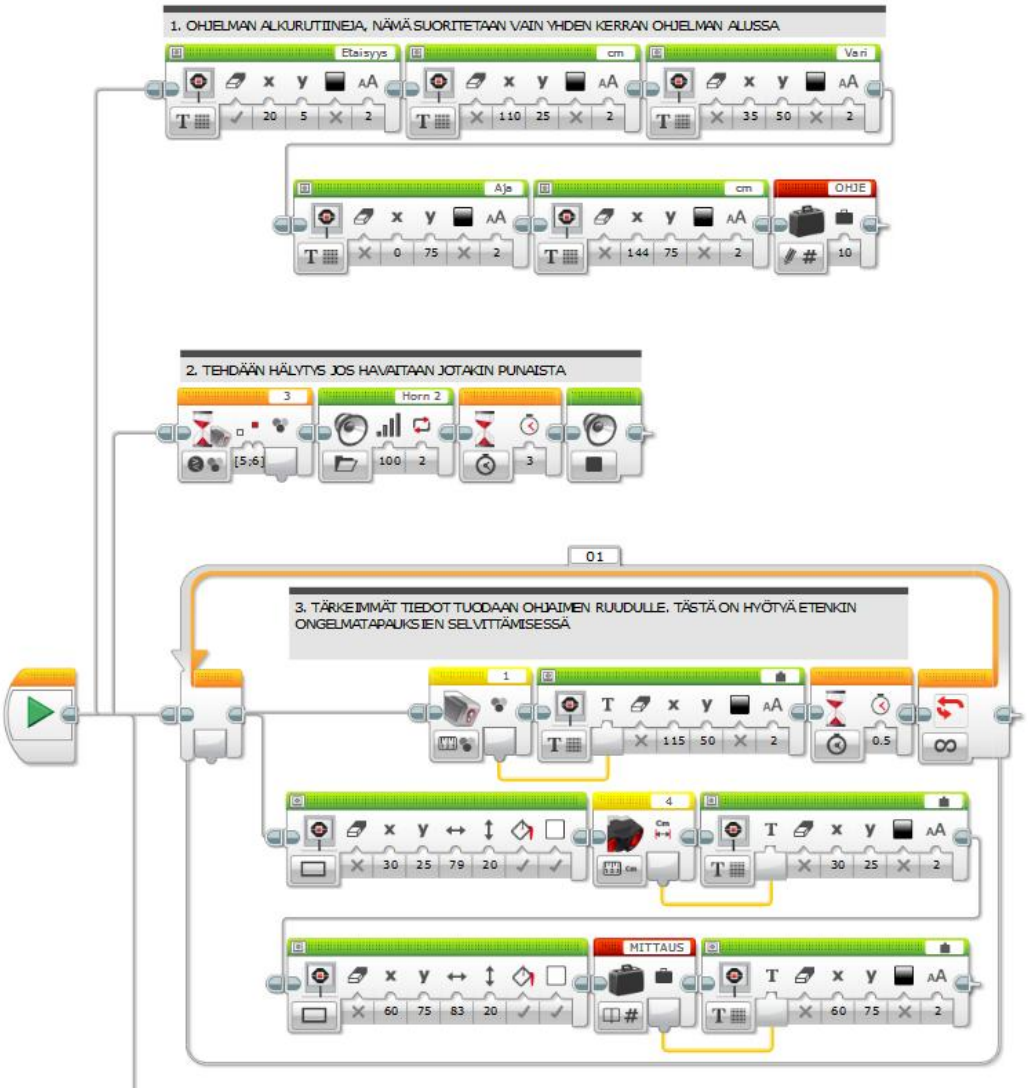
(kuva 13/5)

- Mitä valokenno oikeasti näyttää, kun kohde löytyy? Se kannattaa tarkastaa viemällä robotti kohteen luo.
- Kun robotti mittasi matkan, jota sen pitäisi seuraavaksi lähteä ajamaan, mittasiko se sen oikein? Tätä voi arvioida helposti, kun tuo matka kerrotaan näyttöruudulla. Ohjelma kirjoittaa näytölle ”mittauksen” sijaan lyhyemmän sanan ”AJA”, jotta koko asia saadaan mahdutettua näytöllä yhdelle riville.

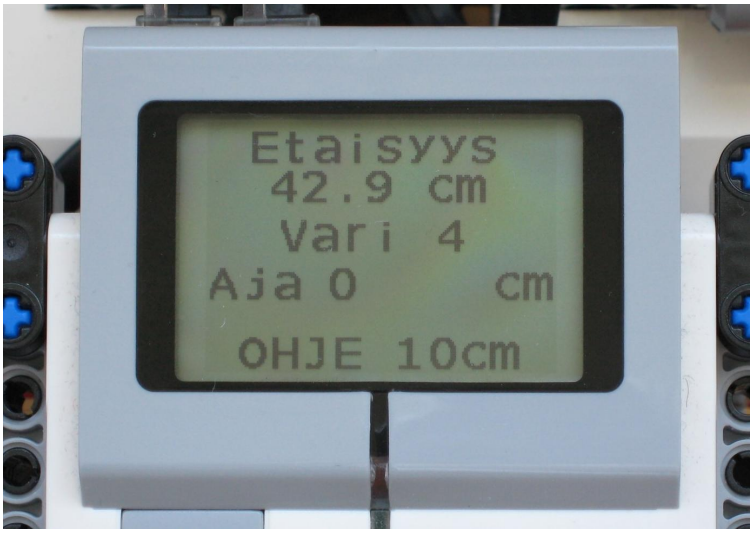
Vastaus edellisen sivun kysymykseen

Näytölle pitäisi tulla nyt näkyviin neljä ensimmäistä riviä. Koska näytönohjaustoimilohko kirjoittaa ruudulle aina vain vanhan päälle ja etäisyyden mittaustiedon pituus voi vaihdella parista numerosta neljään, on meidän joka silmukan kierrolla ensin tehtävä sille tyhjä suorakaiteen muotoinen tila. Muuten ruudulle voi jäädä jo aikaisemmin kirjoitettuja numeroita sotkemaan tuota lukemaa.

Ohjelma näyttää nyt sunnilleen tältä.



(kuva 14/5)



(kuva 15/5)

Seuraavaksi opetellaan, miten koneelle, kuten robotille voidaan kertoa jokin sen tarkoitetun toiminnan kannalta tärkeä tieto.

Kuvassa näytön alimmalla rivillä näkyy ohjearvo robotille, miten kaukana sen odotetaan kulkevan labyrintin seinästä (etäisys ultraäänianturista).

Ohjearvoa halutaan muuttaa ruudun alla olevista YLÖS ja ALAS -painikkeista, yksi sentti kerrallaan. Suurin etäisyys on 12cm ja pienin 5cm.

Harjoitellaan samalla tietotyypin muunnosta

Tämän kirjan ensimmäisen ohjelmointiharjoituksen yhteydessä esiteltiin erilaiset tietotyypit, tietojohtimet, sekä miten näitä voi käyttää. Tässä harjoituksessa voidaan hyödyntää yhtä tietotyyppien yhteensopivuutta.

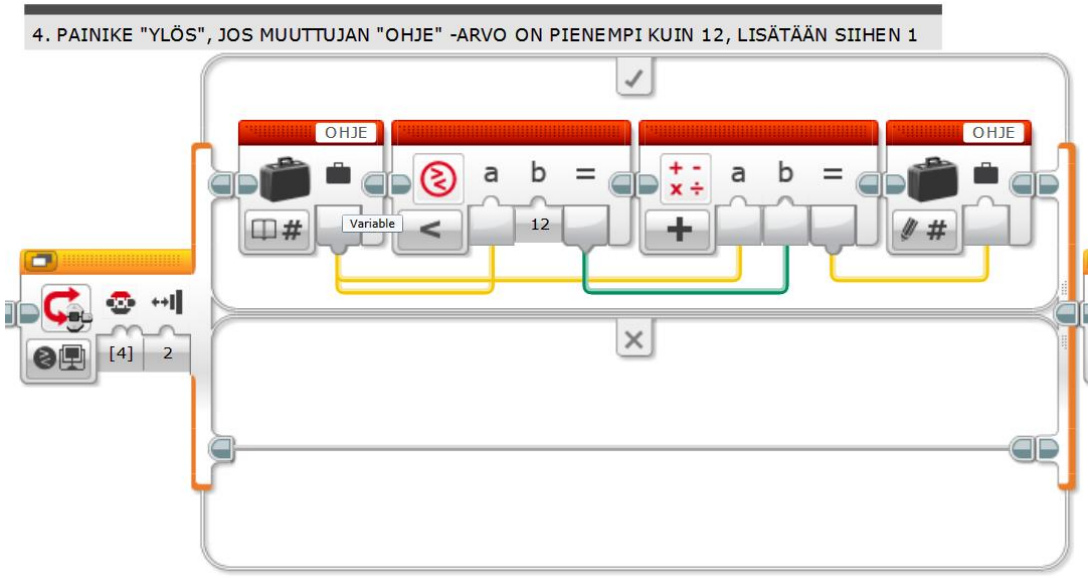
Lähtö	**Tulo**	**Mitä tuloon kirjoittuu**
Tosi/epätosi (Logic)	Numeerinen	Tosi=1, Epätosi=0

(taulukko 1/5) Tietotyyppiä Logic voi käyttää ohjelmassa myös kuin numeroa

Alla olevassa kuvassa IF-THEN –toimilohko on parametroitu seuraamaan YLÖS –painikkeen mahdollisia painalluksia.

4. Painalluksesta (YLÖS)

- Luetaan numero muistipaikasta OHJE.
- Verrataan onko tämä numero pienempi kuin 12, eli maksimi.
- Jos ohje oli alle 12, vertailun tulos on TOSI, eli 1
- Mutta jos OHJE on jo 12, vertailun tulos on EPÄTOSI, eli nolla
- Viedään OHJE, sekä vertailun tulos laskentatoimilohkoon.
- Suoritetaan yhteenlasku, OHJE +1, tai OHJE +0
- Talletetaan laskun tulos muistipaikkaan OHJE

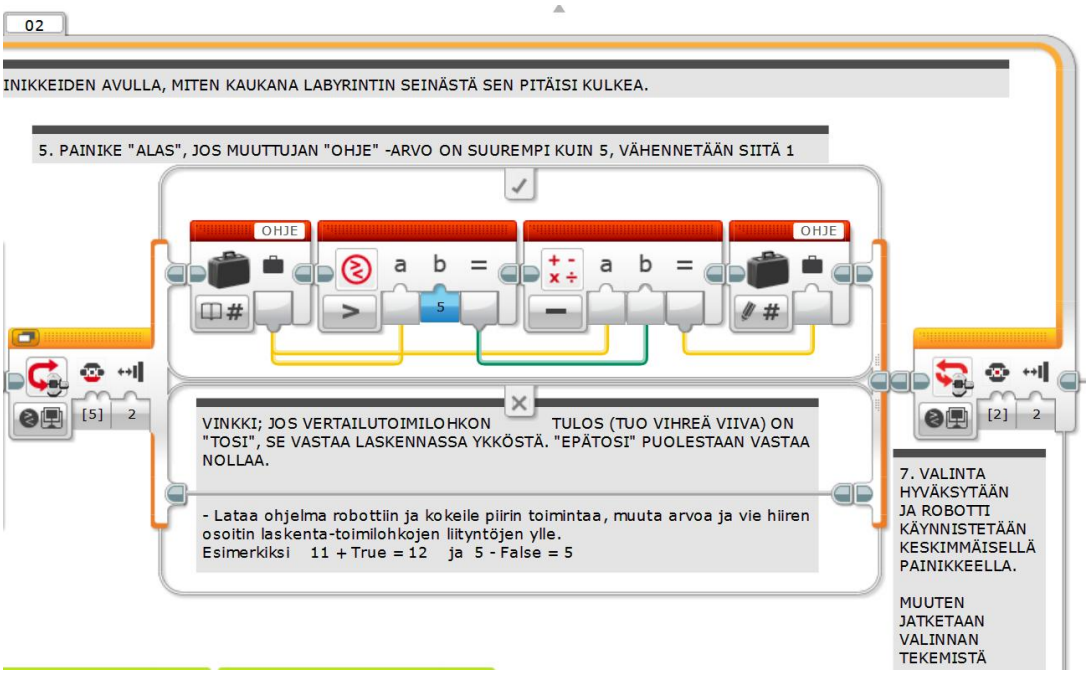


(kuva 16/5) Näin kirjoitettuna tämän ohjelman osan pituus jää alle puoleen tavallisesta

- Ja silloin kun YLÖS –painiketta ei paineta, ohjelma kulkee alempaa reittiä

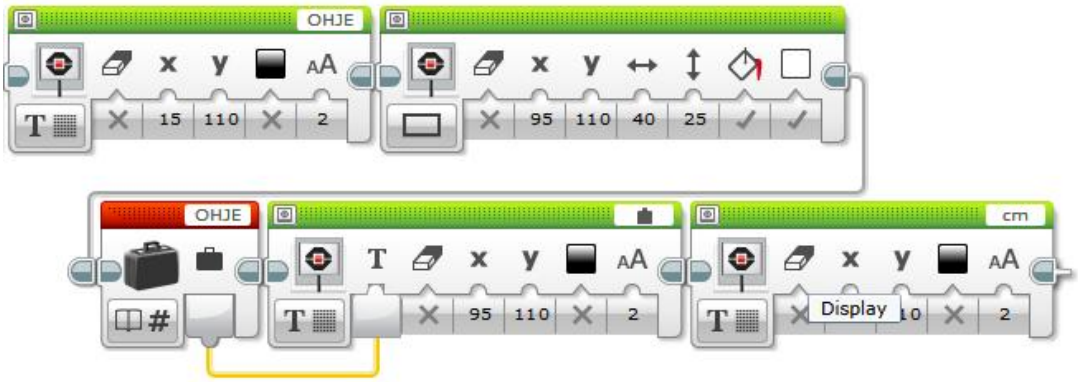
5. Seuraavaksi tehdään melkein samalainen IF-THEN –toimilohko.
   - Löydätkö kaikki neljä eroavaisuutta?

Samassa kuvassa on näytetty myös Silmukka-toimilohkon parametrointi.



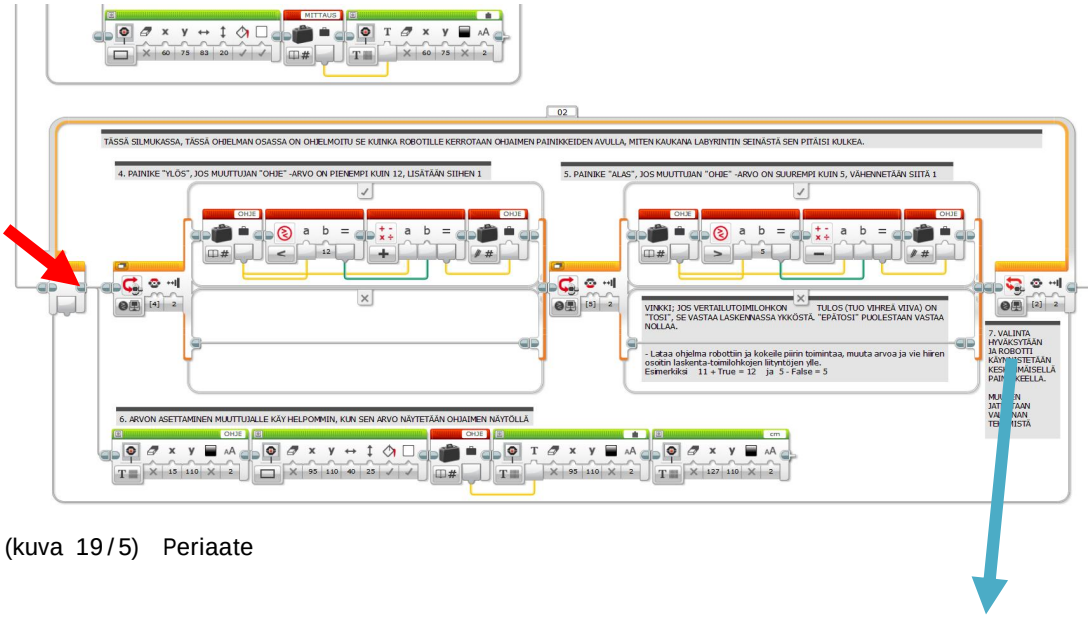
(kuva 17/5) Huom. Silmukan nimeksi on vaihdettu 02

6. Tehdään muuttujan arvon näytöllä esittävä ohjelman pätkä, ennen kuin viedään nämä aikaansaannokset silmukka-toimilohkoon. Seuraavalla sivulla on kuva valmiista silmukasta ohjelmaan liitettynä.



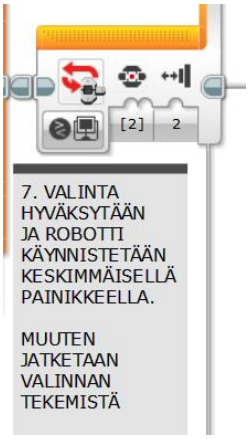
(kuva 18/5)

Vie ensin IF-THEN –paketit silmukkaan. Venytä sitten silmukan alalaitaa alemmaksi ja vie myös näytönohjauksen toimilohkot sen sisälle. Polun haaroittamisen takia ei tarvitse taistella. Kun lähdet vetämään hiirellä polkua silmukan alusta (Punainen nuoli) näytönohjauksen toimilohkoille, ohjelma tekee itse tilaa tuohon alkuun.



(kuva 19/5) Periaate

7. Tästä määritellään silmukasta ulostulo ja samalla robotin varsinaisen labyrintti-ajelun aloittaminen tehtäväksi ohjaimen keskimmäisen painikkeen painalluksesta.

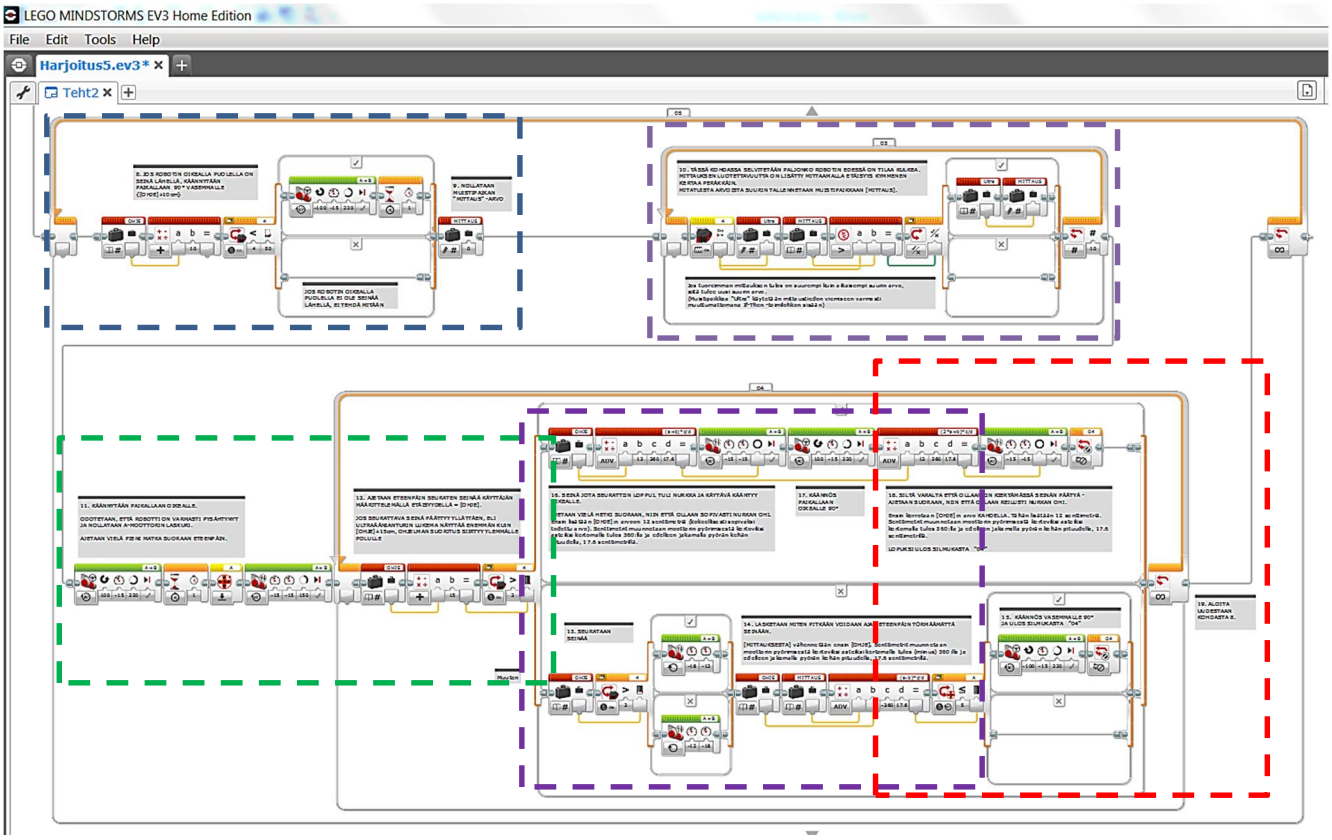


(kuva 20/5)

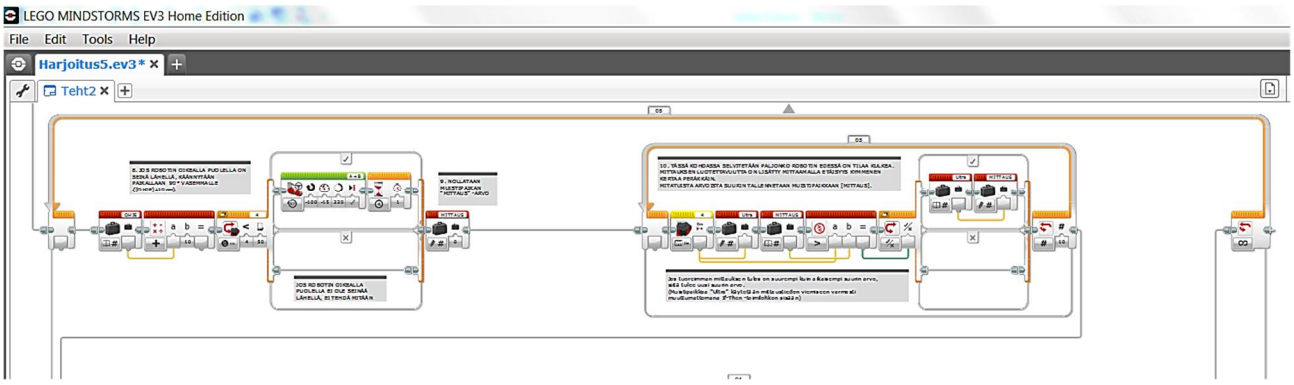
Nyt ohjelmaa voidaan taas kokeilla. Koska robotti ei ole vielä karkamassa mihinkään, voidaan ohjelmointikaapeli pitää kiinni robotissa. Käynnistä ohjelma tietokoneen näytöltä, joko ohjelman aloitus-toimilohkon PLAY-symbolia klikaten, tai näytön oikeasta alanurkasta laitemonitorin Download & PLAY-symbolia klikaten (lataa & käynnistä ohjelma).

Kokeile OHJE-muuttujan arvon muuttamista. Toimiiko se kuten piti? Miten tämä näkyy tietokoneen ruudulla? Viemällä hiiren osoittimen ohjelmaan piirrettyjen tietojohtimien päälle voit nähdä niissä kulkevat tiedot tietokoneen ruudulla.

Ohjelman viimeinen osuus, suuri silmukka numero 05 johon on ohjelmoitu robotin liikkuminen, on kätevämpää käsitellä useammassa osassa. Tässä yleiskuva jo ennakkoon siitä mitä nyt ryhdytään käymään läpi.

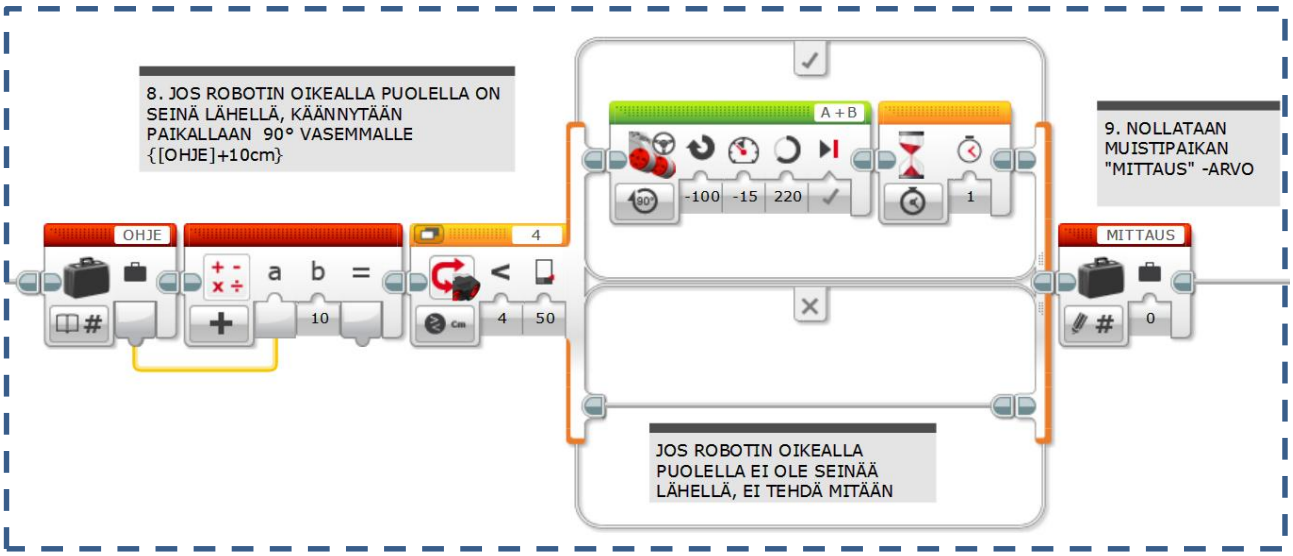


(kuva 21/5) Silmukan 05 sisältö on jaettu viiteen kuvaan, jotka käydään seuraavaksi läpi



(kuva 24/5) Toimintakuvauksen vaiheet 8,9, sekä 10 ruudulla, Suurennokset seuraavalla sivulla

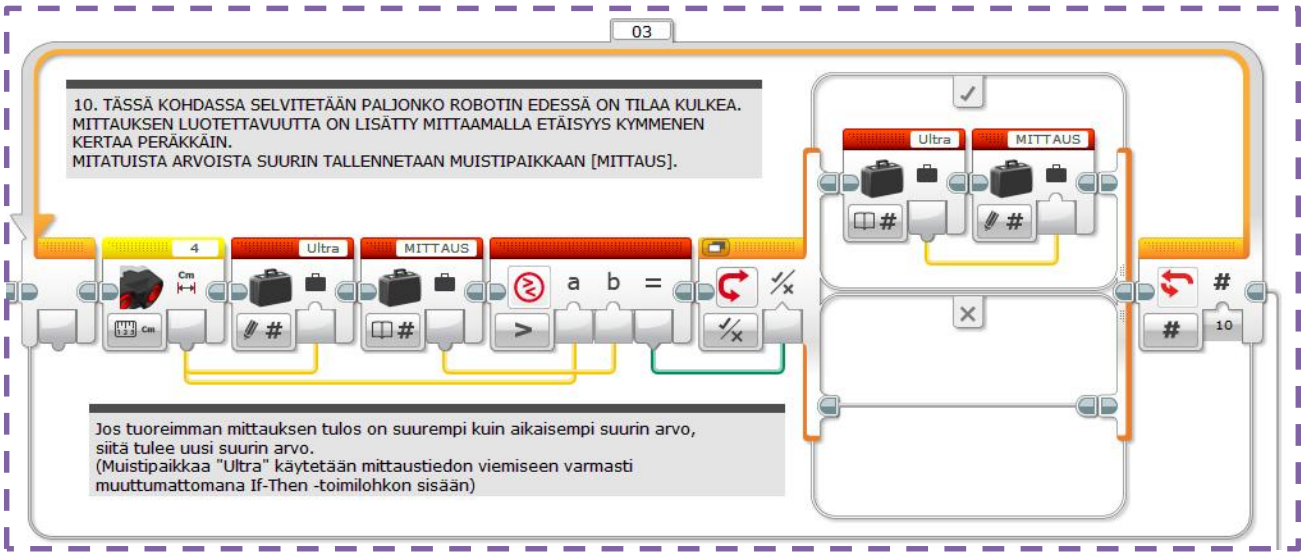
8. ja 9. Edessä olevan vapaan, eli kuljettavissa olevan matkan mittaamista valmistelevat toimenpiteet: Tässä robotti tarkistaa onko sen vieressä seinä. Jos on, suunnataan ultraäänianturi eteenpäin robottia kääntämällä.



(kuva 22/5)

10. Mittaus, tämän voisi ohjelmoida kahdella toimilohkolla, ultraäänianturi + muistipaikka MITTAUS.

- Mutta koska robotillamme ei ole kiire mihinkään ja labyrintissä piilee pieni satunnaisten äänen heijastumisen riski, harjoitellaan tässä samalla yksi tapa parantaa mittauksen luotettavuutta. Mitataan etäisyys kymmenen kertaa perätysten ja käytetään robotin ohjaamiseen suurinta saatua tulosta.



(kuva 23/5) Silmukka 03

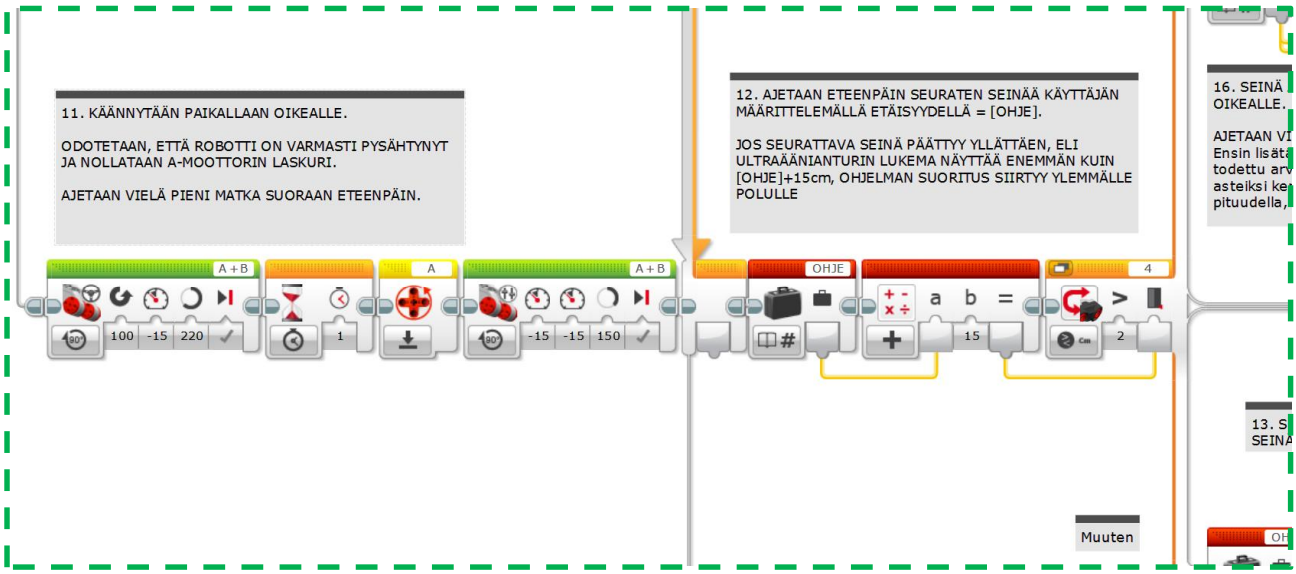
11. Käännytään (aina) paikallaan oikealle, joko takaisin alkuperäiseen kulkusuuntaan, tai jos ennen mittausta ei käännytty vasemmalla, niin sitten robotin kulkusuunta kääntyy tällä käännöksellä 90° oikealle.

Aivan oikein! Oikealle kääntyminen tapahtuu oikeastaan niin, että ei käännytty vasemmalle ennen mittaamista, kun robotin oikealla puolella ei ollut seinää (ja mittaamisen jälkeen käännytään aina oikealle). Ohjelmointi voi olla joskus aika nurinkurista.

Nollataan vielä matkamittari, eli tässä tapauksessa moottorin A pulssilaskuri, sekä ajetaan pieni matka suoraan eteenpäin ennen jatkamista. Tämä viimeksi mainittu liittyy mahdolliseen seinän päädyn sujuvaan kiertämiseen.

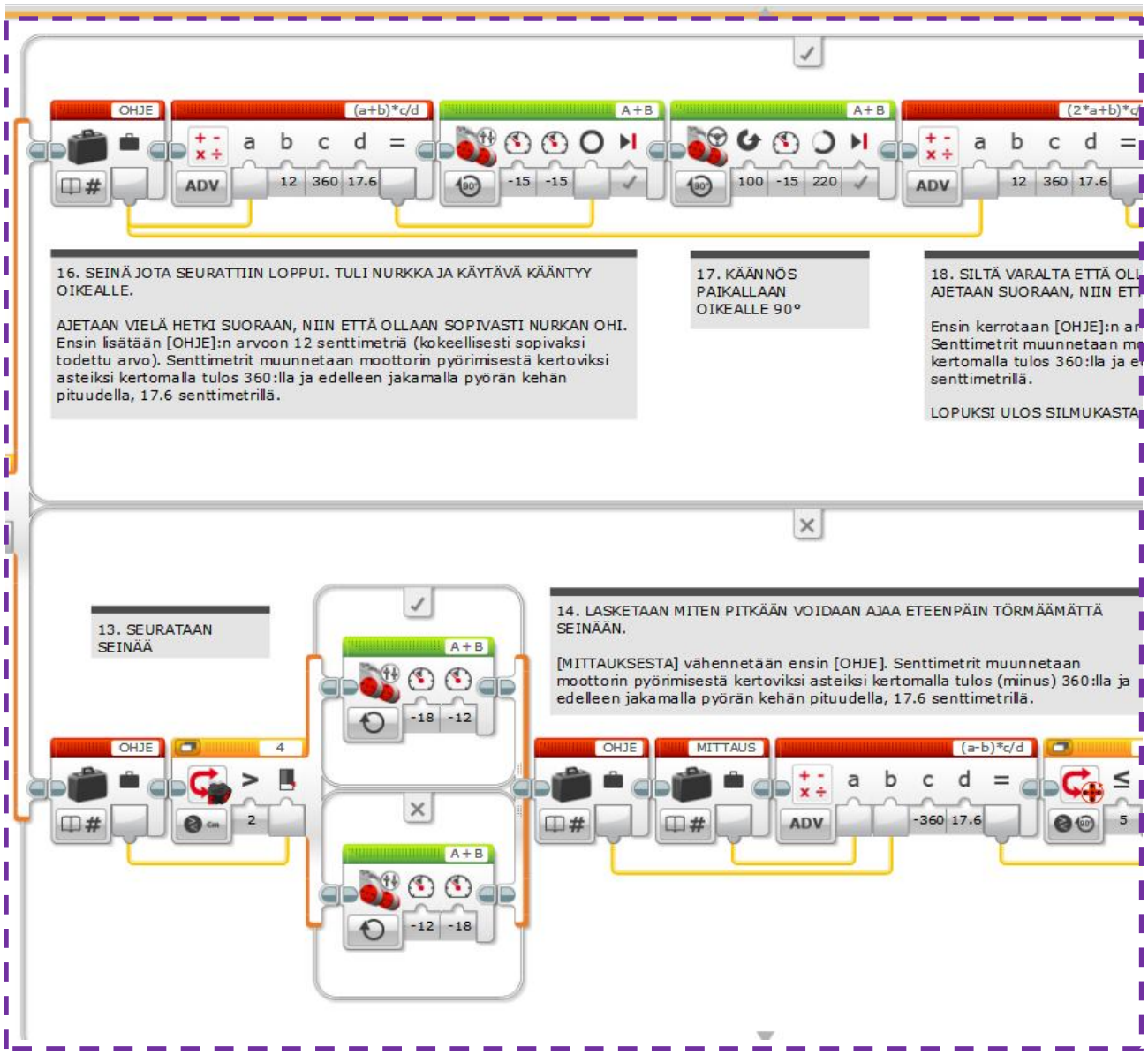
12. Silmukan 04 aloitus. Tässä silmukassa seurataan, ei viivaa, vaan seinää. Ohjelman pitää kuitenkin osata keskeyttää seinän seuraaminen ja tulla ulos tästä silmukasta (04) kahdessa hyvin erilaisessa tilanteessa.

- Ensimmäinen näistä tilanteista on tilanne jossa seinän seuraaminen pitää keskeyttää on tietysti kuljettavissa olleen matkan täyttyminen, edessä olevaan seinään törmäämisen välttäminen.



(kuva 25/5) Silmukka 04 ja seinän seurannan iso IF-THEN –toimilohko alkavat

- Toinen näistä tilanteista on käytävän kääntyminen oikealle, jonka robotti näkee ultraäänianturillaan mittaustuloksen äkillisenä kasvamisena, ikäänkuin seurattavan seinän loppumisena ennen ajettavan matkan täyttymistä. Ohjelmaan on laitettu tämän mittaustuloksen äkillisen kasvamisen raja-arvoksi 15cm (katso kuva).

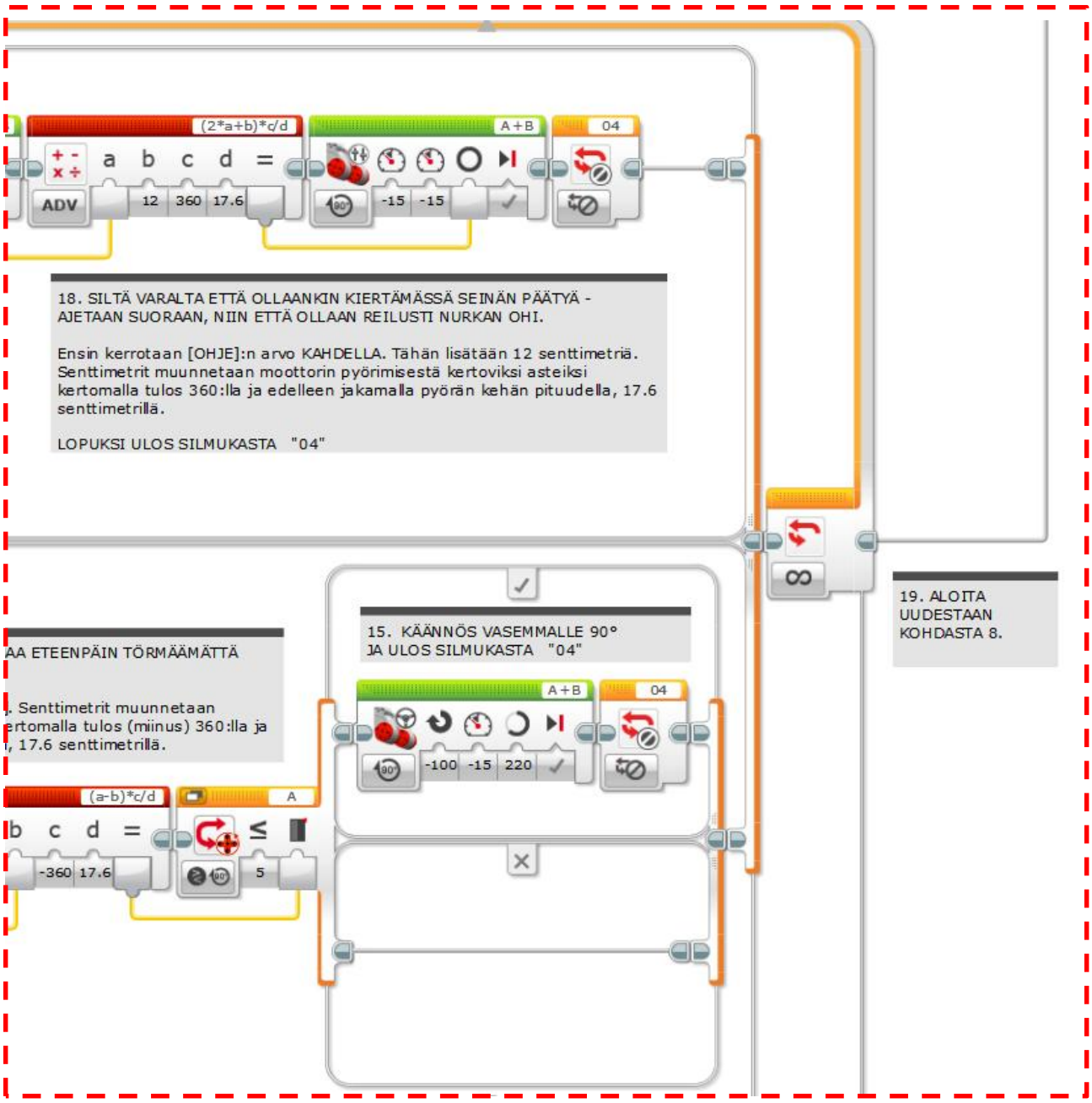


(kuva 26/5) Ison IF-THEN toimilohkon alkuosa. Muistathan olla aina tarkkana laskujärjestyksen kanssa, kun teet pitkiä laskukaavoja.

13. Yksinkertainen seinänseuranta on periaatteeltaan aivan samanlainen kuin viivanseuranta, jota harjoiteltiin aiemmin.

14. Ajettavissa olevan matkan laskeminen. Robottimme mittasi paljonko sillä on edessään vapaata tilaa, sanokaamme esimerkiksi 80cm. Tästä vähennetään seuraavaksi se matka, millä etäisyydellä seinästä sen pitäisi pysytellä esim. 10cm. Robotti voi siis ajaa huoletta eteenpäin 70cm.
Vaan eihän robotin moottorien ohjaus ymmärrä mitään senttimetreistä. Meidän pitää vielä muuttaa nuo senttimetrit sellaiseen muotoon, jota moottorinohjaus-toimilohko ymmärtää, eli joko moottorin kierroksiksi, tai kierroksen osiksi eli asteiksi (katso kuva).

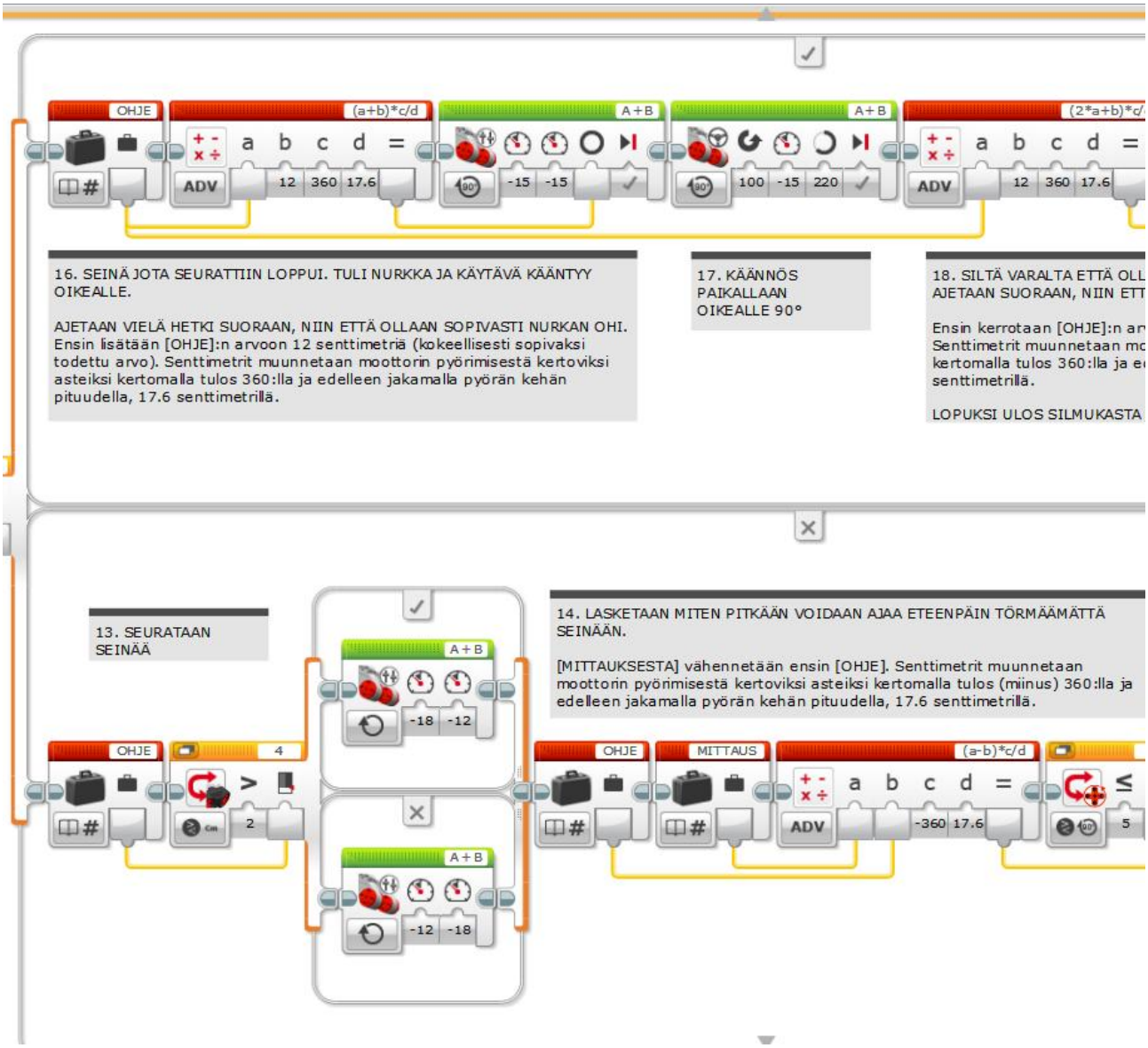
15. Kun matka tulee täyteen, käännytään 90° vasemmalle ja keskeytetään silmukan 04 toistaminen.



(kuva 27/5) Ison IF-THEN toimilohkon loppu-osa ja silmukan 04 loppu

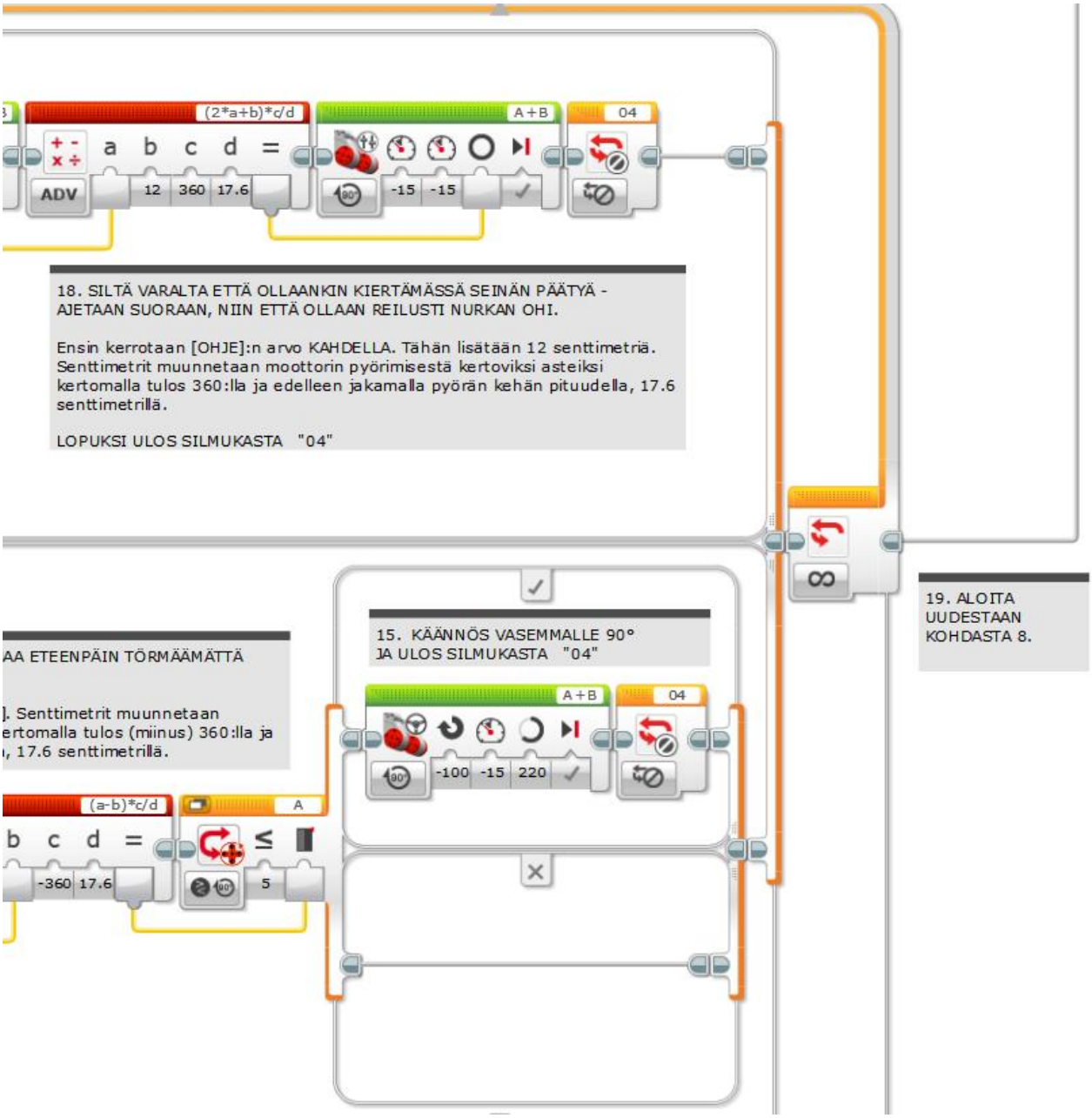
16. Mikäli nyt kuitenkin käy niin, että seinä jota robotti seurasi, kääntyykin oikealle ennen ajettavan matkan täyttymistä (katso kohta 12), ajetaan vielä suoraan sen verran nurkasta ohi, että kun robotti kääntyy uuteen suuntaan, se olisi heti suunnilleen halutulla etäisyydellä seinästä. Tarvittava matka saadaan laskemalla (katso kuva).

17. Käänytään 90° oikealle.



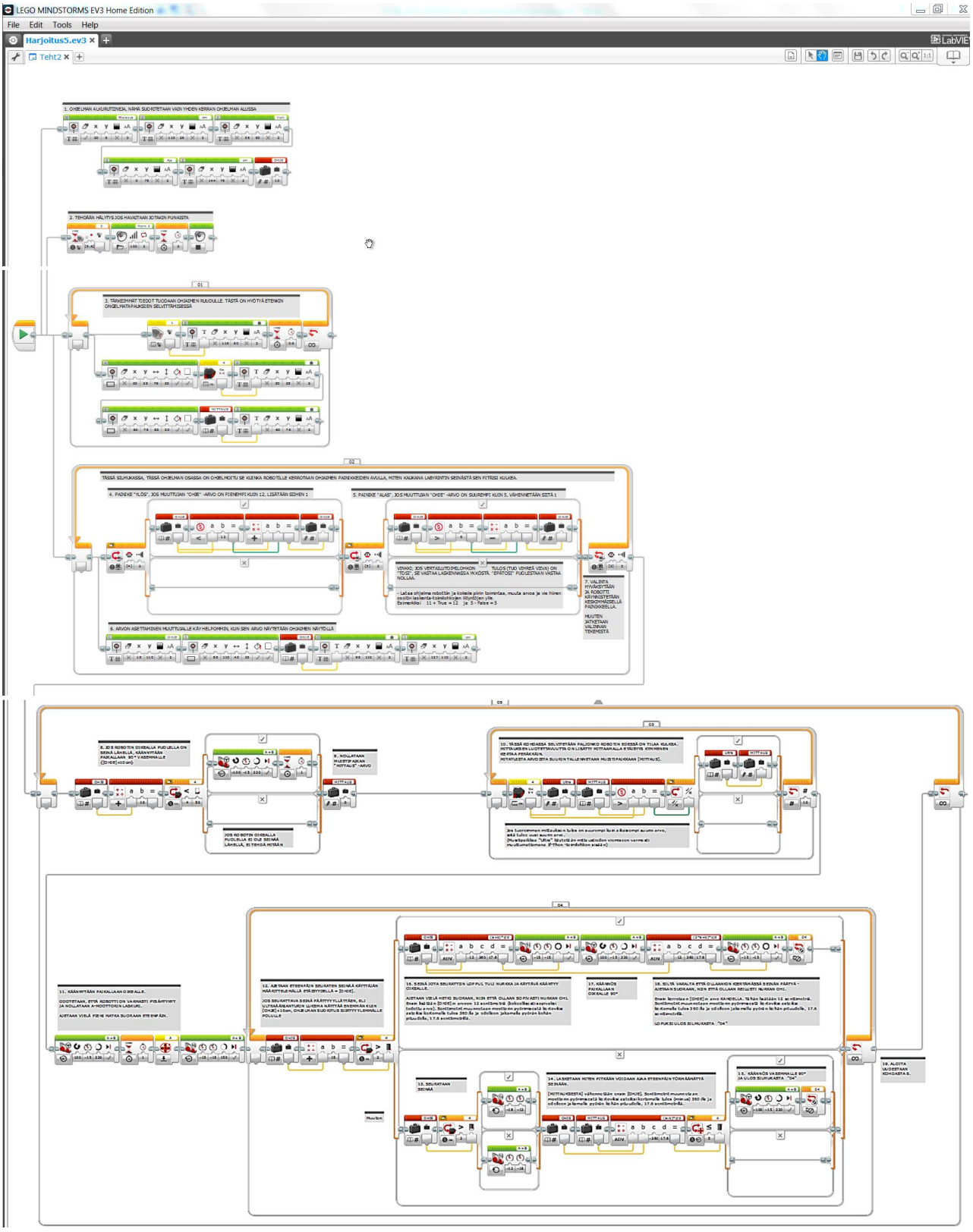
(kuva 28/5) Ison IF-THEN toimilohkon alkuosa. Muistathan olla aina tarkkana laskujärjestyksen kanssa, kun teet pitkiä laskukaavoja.

18. Käännöksen jälkeen, ennen kuin ryhdytään taas mittaamaan miten paljon edessä on tilaa edetä, ajetaan vielä pieni matka suoraan (katso kuva) ja keskeytetään silmukan 04 toistaminen.



(kuva 29/5) Ison IF-THEN toimilohkon loppu-osa ja silmukan 04 loppu

19. Palataan silmukan 05 alkuun, ajettavissa olevan matkan mittaamiseen.



(kuva 30/5) Lopputulos, ohjelma kolmen kuvan kuvakollaasina. Ohjelma on reilusti pidempi, kuin yksikään aikaisemmin. Onhan siinä kerrattu samalla miltei kaikki, mitä on tähän mennessä opittu.

Tässä harjoituksessa on kerrattu

- Ohjelman suorittamiseen vaikuttavien haarautumisien, silmukoiden ja ehdollisien ohjelman osien ohjelmoiminen.
- Muuttujien ja muistipaikkojen käyttöä ohjelmissa, sekä erilaisien tietotyyppien ja niiden muunnoksien käyttöä
- Moottori, anturi, ääni, laskenta ja vertailu –toimilohkojen käyttöä.
- Käyttöliittymän, eli näyttöruudun ja ohjaimen painikkeiden ohjelmointia.
- Matemaattisten kaavojen käyttöä ohjelmoinnissa ja ohjelmassa.
- Kirjoitettavan ohjelman toimintakuvauksen laatimista.

(kuva 31/5) Koeajolla.

Tässä labyrintissa on huomioitu sen mahdollinen esitteleminen pöydällä. Tekemällä labyrintin huoneen nurkkaan säästyy 2.5m puutavaraa.

Kokeilkaa lopuksi kiertää labyrinttia eri etäisyksillä seinästä

Tehtävä 24. Käyttäjän informointi ohjaimen merkkivaloilla

Olet varmasti jo huomannut EV3-ojaimen painikkeita ympäröivän valon, joka palaa tai vilkkuu milloin punaisena, milloin vihreänä. Ohjain kertoo sillä käyttäjälle monenlaisista asioista. Näistä kerrotaan tarkemmin EV3:n käyttöohjeessa sivulla 6.

Tätä merkkivaloa voidaan myös ohjelmoida. Sen avulla voi välittää käyttäjälle tietoa esimerkiksi ohjelman kulun etenemisestä, tai vaikka valmiudesta ryhtyä suorittamaan jotakin tehtävää.

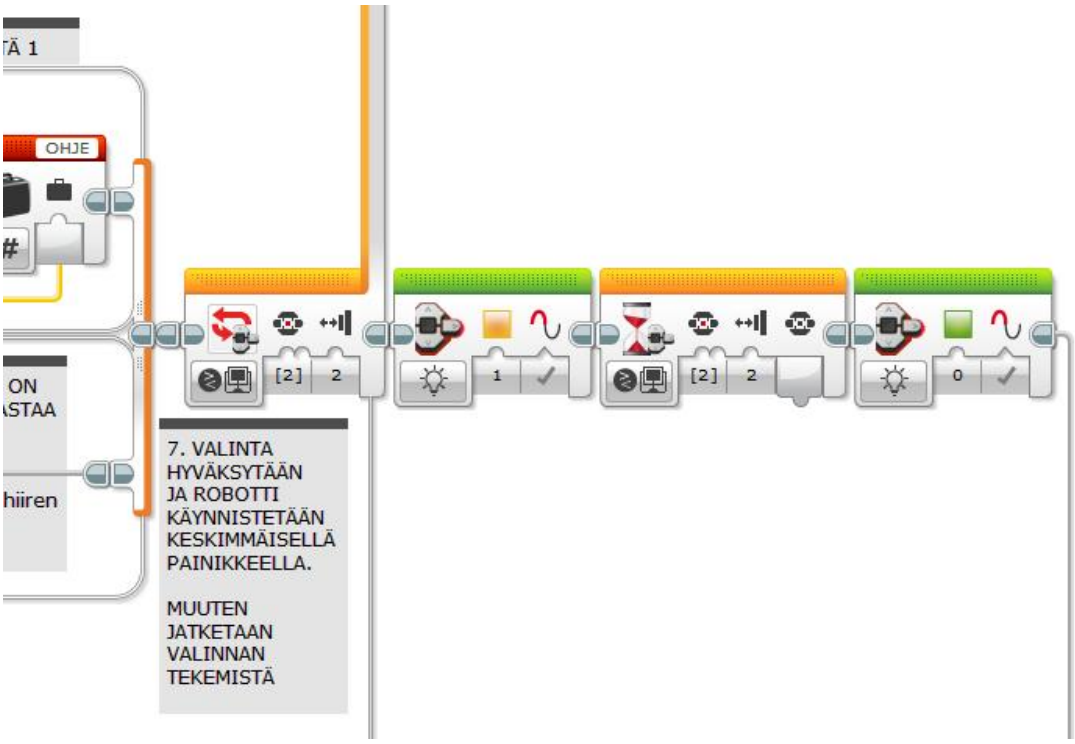
Täydenntään kokeeksi labyrintti-ohjelmaa muutamalla robotin käyttöä helpottavalla merkkivalolla. Eli lisätään ohjelmaan merkkivalo-toimilohkoja kertomaan käyttäjälle milloin se

1. odottaa käyttäjän asettavan ja hyväksyvän OHJE-arvon (punainen vilkkuva valo)
2. odottaa lähtökäskyä (keltainen vilkkuva valo)
3. suorittaa tehtävää (vihreä vilkkuva valo)

Kohdan 1 mukainen lisäys voidaan tehdä vaikka kuvan esittämään paikkaan ohjelmassa. Punainen valo alkaa nyt vilkkua ohjelman käynnistyessä.

(kuva 32/5)

Kohdan 2 ja 3 lisäykset on helppo laittaa ohjelmaan tuonne ohjearvon valinnan jatkoksi (katso kuva) Keltaisen ja vihreän vilkun väliin on lisätty viive-toimilohko, joka odottaa painikkeen painallusta liikkeelle lähdön merkiksi.



(kuva 33/5)

Kokeile jälleen robottia labyrintissa. Arvioi oliko sen käyttäminen nyt helpompaa / selkeämpää?