TI-NSpire™-ohjelmaeditori Guidebook

Lue lisää TI teknologiaan käytönaikaisessa ohjeessa education.ti.com/eguide.

Tärkeää tietoa

Jollei toisin mainita Ohjelmiston mukana toimitetussa lisenssissä, Texas Instruments ei anna mitään nimenomaista tai epäsuoraa takuuta, mukaan lukien mutta ei rajoittuen mihinkään epäsuoraan kauppatavaraa ja sopivuutta tiettyyn tarkoitukseen liittyvistä takuista ohjelmista tai kirjamateriaaleista ja tekee tällaisista materiaalit, jotka ovat saatavilla vain "as-on" -periaatteella. Texas Instruments ei missään tapauksessa saa olla vastuussa mistään erityisistä, vakuudoista, satunnaisista tai välillisistä vahingoista näiden materiaalien ostamisen tai käytön yhteydessä tai siitä aiheutuvista vahingoista ja Texas Instrumentsin yksinomaisesta ja yksinomaisesta vastuusta riippumatta siitä, ei saa ylittää ohjelman lisenssissä vahvistettua määrää. Lisäksi Texas Instruments ei ole vastuussa mihinkään muuhun osapuoleen kohdistuvasta vaatimuksesta näiden materiaalien käyttämiseen.

TI-Nspire™ -ohjelmisto käyttää Luaa kirjoitusympäristönä. Tekijänoikeus- ja lisenssitiedot, katso http://www.lua.org/license.html.

TI-Nspire™ -ohjelmisto käyttää Chipmunk Physics -versiota 5.3.4 simulaatioympäristönä. Katso lisenssitiedot http://chipmunk-physics.net/release/Chipmunk-5.x/Chipmunk-5.3.4-Docs/.

Microsoft® ja Windows® ovat Microsoft Corporationin rekisteröityjä tavaramerkkejä Yhdysvalloissa ja / tai muissa maissa.

Mac OS®, iPad® ja OS X® ovat Apple Inc: n rekisteröityjä tavaramerkkejä.

Unicode® on Unicode, Inc: n rekisteröity tavaramerkki Yhdysvalloissa ja muissa maissa.

Todelliset tuotteet saattavat erota hieman mukana tulevista kuvista.

Contents

Ohjelmaeditorin käytön aloitus

Voit luoda käyttäjän määrittämiä toimintoja tai ohjelmia näppäilemällä määritelausekkeita laskimen syöteriville tai käyttämällä ohjelmamuokkainta. Ohjelmaeditorilla on eräitä etuja ja sen toiminnot käsitellään tässä osiossa. Lisätietoja löytyy osiosta *Laskin*.

- Editorissa on ohjelmointimallineita ja valintaikkunoita, joiden avulla voit määrittää funktioita ja ohjelmia käyttämällä oikeaa syntaksia
- Editorissa voit syöttää monirivisiä ohjelmalausekkeita tarvitsematta käyttää erityistä näppäinjärjestystä kunkin rivin syöttämisessä.
- Voit luoda omia ja julkisia kirjasto-objekteja (muuttujia, funktioita ja ohjelmia) helpolla tavalla. Lisätietoja löytyy osiosta *Kirjastot*.

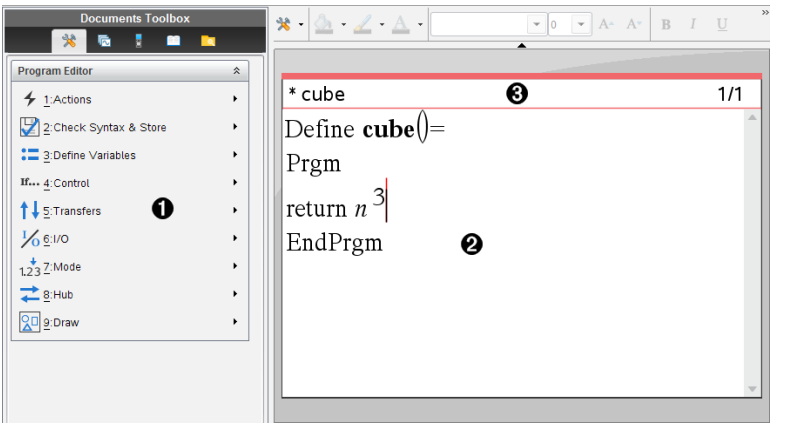
Ohjelmaeditorin käynnistys

▶ Ohjelmaeditorin-sivun lisääminen sen hetkiseen tehtävään:

Napsauta työkalupalkissa **Lisää > Ohjelmaeiditori > Uusi**.

Kämmenlaite: Paina doc ja valitse **Lisää > Ohjelmaeditori > Uusi**.

Huomaa: Editoriin pääsee myös Laskin-sivun **Toiminnot & Ohjelmat** -valikosta.



❶ Ohjelmaeditori-valikko – Tämä valikko on käytettävissä aina, kun olet ohjelmaeditorin työalueella ja näyttötila on Normaali.

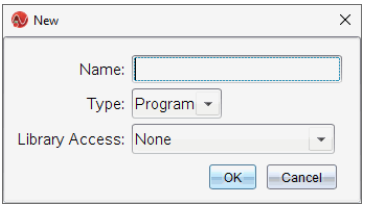
❷ Ohjelmaeditorin työalue

❸ Tilarivi näyttää rivin numerotiedot ja muokattavan toiminnon tai ohjelman nimen. Asteriski (*) osoittaa, että tämä funktio on "likainen", joka tarkoittaa, että funktio on muuttunut siitä, kun sen syntaksi viimeisen kerran tarkistettiin ja tallennettiin.

Ohjelman tai funktion määrittely

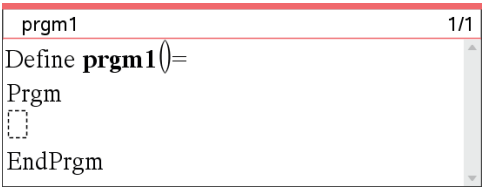
Uuden ohjelmaeditorin käynnistäminen

1. Varmista, että olet siinä asiakirjassa ja ongelmassa, johon haluat luoda ohjelman tai funktion.
2. Napsauta sovelluksen työkalupalkin **Lisää**-painiketta ja valitse **Ohjelmaeditori > Uusi**. (Paina kämmenlaitteessa doc ja valitse **Lisää > Ohjelmaeditori > Uusi**.)



3. Kirjoita funktion tai ohjelman nimi, jota olet määrittämässä.
4. Valitse **Tyyppi** (**Ohjelma** tai **Funktio**).
5. Aseta **Kirjastoon pääsy**:
   - Jos valitset **Ei mitään**, voit käyttää funktiota tai ohjelmaa ainoastaan nykyisessä asiakirjassa tai tehtävässä.
   - Jos valitset **LibPriv**, asetat funktion tai ohjelman minkä tahansa asiakirjan käytettäväksi, mutta ei näkyväksi Katalogissa.
   - Jos napautat **LibPub (Näytä katalogissa)**, asetat funktion tai ohjelman minkä tahansa asiakirjan käytettäväksi ja näkyväksi katalogissa. Lisätietoja löytyy osiosta *Kirjastot*.
6. Klikkaa **OK**.

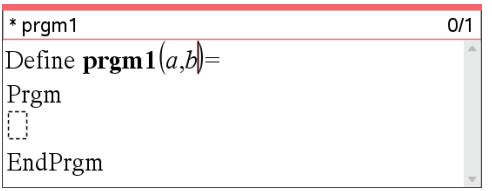
   Uusi Ohjelmaeditori-istunto ja malli, joka vastaa tekemiäsi valintoja, avautuu.



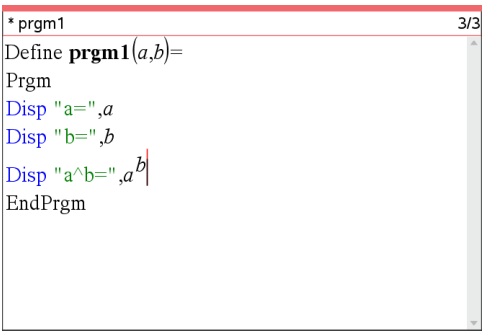
Rivien syöttäminen funktioon tai ohjelmaan

Ohjelmaeditori ei suorita komentoja tai laske lausekkeita, kun kirjoitat niitä. Ne suoritetaan vasta, kun lasket funktion tai suoritat ohjelman.

1. Jos funktiosi tai ohjelmasi vaatii käyttäjää toimittamaan argumentteja, kirjoita parametrin nimet nimeä seuraaviin sulkuihin. Erota parametrit pilkulla.



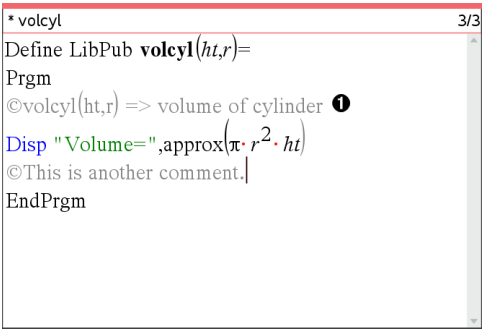
2. Kirjoita lauserivit, jotka muodostavat funktiosi tai ohjelmasi rivien Func ja EndFunc (tai Prgm ja EndPrgm) väliin.



- Voit joko kirjoittaa funktion ja komentojen nimet tai syöttää ne katalogista.
- Jos rivi on pidempi kuin näytön leveys, voit tarkastella koko lauseketta vierittämällä.
- Paina **Enter**-painiketta aina rivin lopussa. Enter-painike syöttää uuden tyhjän rivin, jolloin voit jatkaa uusien rivien syöttöä.
- Selaa funktiota tai ohjelmaa komentojen syöttämiseksi tai muokkaamiseksi käyttäen nuolinäppäimiä ◄, ►, ▲ ja ▼.

Kommenttien lisääminen

Kommentit voivat olla hyödyllisiä henkilöille, jotka tarkastelevat tai muokkaavat ohjelmaa. Niitä ei näytetä, kun ohjelma on käynnissä, eivätkä ne vaikuta ohjelman työnkulkuun. ©-symboli näkyy rivin alussa kommentin kera.



❶ Kommentti näyttää vaaditun syntaksin. Koska tämä kirjasto-objekti on julkinen ja tämä kommentti on ensimmäinen rivi Func- tai Prgm-lohkossa, kommentti näkyy katalogissa ohjeena. Lisätietoja löytyy osiosta *Kirjastot*.

Kommentin lisääminen:

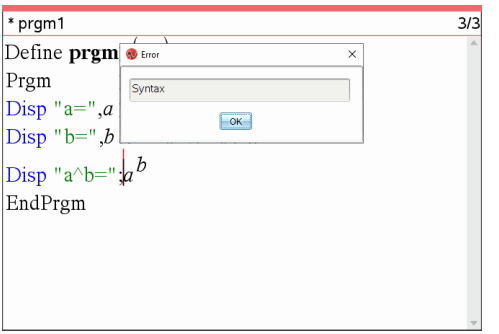
1. Sijoita kohdistin sen rivin loppuun, johon haluat lisätä kommentin.
2. Napsauta **Toiminnot**-valikossa **Lisää kommentti** tai paina **Ctrl+T**.
3. Kirjoita kommentti ©-symbolin jälkeen.

Syntaksin tarkastaminen

Ohjelmaeditorin avulla voit tarkistaa funktion tai ohjelman oikean syntaksin.

- Napsauta **Tarkista syntaksi ja tallenna** -valikossa **Tarkista syntaksi**.

  Jos syntaksin tarkastuksessa löytyy syntaksivirheitä, näkyviin tulee virheilmoitus, ja editori yrittää asettaa kohdistimen ensimmäisen virheen kohdalle, jotta voit korjata sen.



Funktion tai ohjelman tallentaminen

Sinun tulee tallentaa funktio tai ohjelma, jotta sitä voidaan käyttää. Ohjelmaeditori tarkastaa syntaksin automaattisesti ennen tallentamista.

Ohjelmaeditorin vasemmassa yläkulmassa näkyvä tähtimerkki (*) osoittaa, että funktiota tai ohjelmaa ei ole tallennettu.

- Napsauta **Tarkista syntaksi ja tallenna** -valikossa **Tarkista syntaksi ja tallenna**.

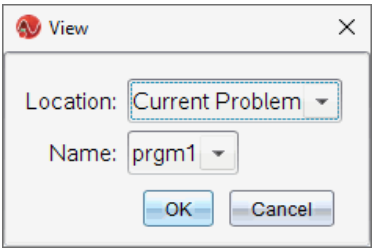
  Jos syntaksin tarkastuksessa löytyy syntaksivirheitä, näkyviin tulee virheilmoitus, ja editori yrittää asettaa kohdistimen ensimmäisen virheen kohdalle.

  Jos syntaksivirheitä ei löydy, tilariville ohjelmaeditorin yläosaan tulee näkyviin viesti ”Stored successfully” (tallennettu onnistuneesti)

Huomaa: Jos funktio tai ohjelma on määritetty kirjasto-objektiksi, sinun tulee myös tallentaa asiakirja määritettyyn kirjastokansioon ja päivittää kirjastot asettaaksesi objektin muiden asiakirjojen käytettäväksi. Lisätietoja löytyy osiosta *Kirjastot*.

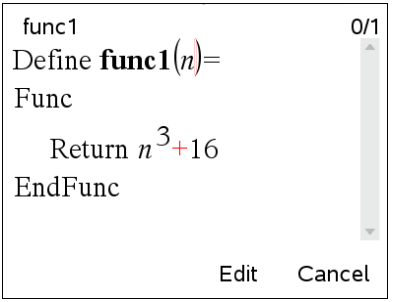
Ohjelman tai funktion näyttäminen

1. Valitse **Toiminnot**-valikon kohta **Näytä**.



2. Jos funktio tai ohjelma on kirjasto-objekti, valitse sen kirjasto **Location (Sijainti)** -luettelosta.
3. Valitse funktion tai ohjelman nimi **Name (Nimi)** -luettelosta.

   Funktio tai ohjelma avautuu katseluohjelmaan.



4. Voit tarkastella funktiota tai ohjelmaa nuolipainikkeiden avulla.
5. Jos haluat muokata ohjelmaa, paina **Muokkaa**-painiketta.

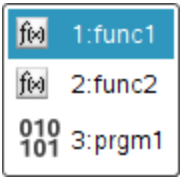
   Huomaa: Edit (Muokkaa) -valinta on käytettävissä vain nykyiseen ohjelmaan määritetyille funktioille ja ohjelmille. Jos haluat muokata kirjasto-objektia, sinun on ensin avattava sen kirjastoasiakirja.

Funktion tai ohjelman avaaminen muokkausta varten

Voit avata funktion tai ohjelman vain nykyisestä tehtävästä käsin.

Huomaa: Lukittua ohjelmaa tai funktiota ei voi muokata. Vapauta objekti siirtymällä Laskin-sovelluksen sivulle ja käyttämällä **unLock**-komentoa.

1. Hae näkyviin käytettävissä olevien funktioiden ja ohjelmien luettelo.
   - Valitse **Toiminnot**-valikon kohta **Avaa**.

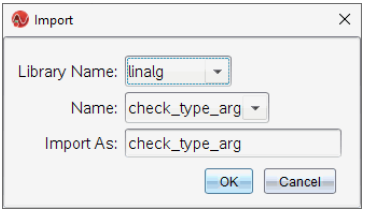


2. Valitse jokin avattavaksi.

Ohjelman tuominen kirjastosta

Voit tuoda kirjasto-objektiksi määritetyn funktion tai ohjelman Ohjelmaeditoriin nykyisen tehtävän sisällä. Tuotu kopio ei ole lukittu, vaikka alkuperäinen olisi lukittu.

1. Valitse **Toiminnot**-valikon kohta **Vie**.



2. Valitse **Library Name (Kirjaston nimi)**.
3. Valitse objektin **Name (Nimi)**.
4. Jos haluat muuttaa siirretyn objektin nimeä, kirjoita nimi kohtaan **Import As (Tuo nimellä)**.

Kopion luominen funktiosta tai ohjelmasta

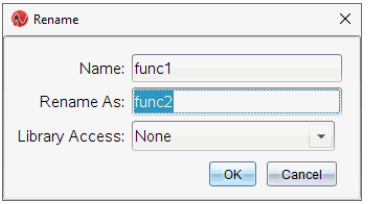
Uuden funktion tai ohjelman luominen on mahdollisesti helpompaa aloittaa kopioimalla nykyinen funktio tai ohjelma. Luomasi kopio ei ole lukittu, vaikka alkuperäinen olisi lukittu.

1. Valitse **Toiminnot**-valikon kohta **Luo kopio**.
2. Kirjoita uusi nimi tai hyväksy ehdotettu nimi napsauttamalla **OK**-painiketta.
3. Jos haluat muuttaa käyttötasoa, valitse **Library Access (Kirjaston käyttö)** ja määritä uusi taso.

Ohjelman tai funktion nimeäminen uudelleen

Voit nimetä nykyisen funktion tai ohjelman uudelleen ja (halutessasi) muuttaa sen käyttötasoa.

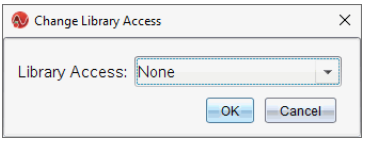
1. Valitse **Toiminnot**-valikon kohta **Nimeä uudelleen**.



2. Kirjoita uusi nimi tai hyväksy ehdotettu nimi napsauttamalla **OK**-painiketta.
3. Jos haluat muuttaa käyttötasoa, valitse **Library Access (Kirjaston käyttö)** ja määritä uusi taso.

Kirjaston käyttötason muuttaminen

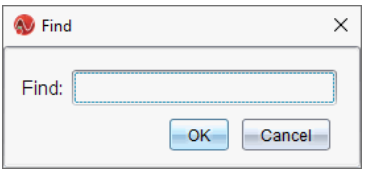
1. Valitse **Toiminnot**-valikon kohta **Muuta kirjaston käyttöä**.



2. Valitse **Library Access (Kirjaston käyttö)**:
   - Jos haluat käyttää funktiota tai ohjelmaa vain nykyisestä Laskin-sovelluksen tehtävästä, valitse **None (Ei mitään)**.
   - Jos haluat käyttää funktiota tai ohjelmaa mistä tahansa asiakirjasta mutta et halua sen näkyvän katalogissa, valitse **LibPriv**.
   - Jos haluat käyttää funktiota tai ohjelmaa mistä tahansa asiakirjasta ja haluat sen myös näkyvän katalogissa, valitse **LibPub**.

Tekstin etsiminen

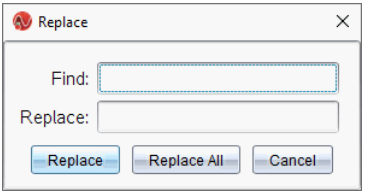
1. Valitse **Toiminnot**-valikon kohta **Etsi**.



2. Kirjoita etsittävä teksti ja napsauta **OK**.
   - Jos teksti löytyy, se näkyy korostettuna ohjelmassa.
   - Jos tekstiä ei löydy, näkyviin tulee ilmoitusviesti.

Tekstin etsiminen ja korvaaminen

1. Valitse **Toiminnot**-valikon kohta **Etsi ja korvaa**.



2. Kirjoita etsittävä teksti.
3. Kirjoita korvaava teksti.
4. Korvaa kohdistimen perässä oleva ensimmäinen esiintymä napsauttamalla **Korvaa**-painiketta tai korvaa kaikki esiintymät napsauttamalla **Korvaa kaikki** -painiketta.
5. **Huomaa:** Jos teksti löytyy matematiikkamallin sisältä, näkyviin tulee viesti, jossa varoitetaan, että korvausteksti korvaa koko mallin eikä ainoastaan löytynyttä tekstiä.

Nykyisen funktion tai ohjelman sulkeminen

▶ Valitse **Toiminnot**-valikon kohta **Sulje**.

Jos funktiossa tai ohjelmassa on tallentamattomia muutoksia, sovellus kehottaa tarkastamaan syntaksin ja tallentamaan ennen sulkemista.

Ohjelmien suorittaminen ja toimintojen arviointi

Ohjelman tai toiminnon määrittelyn ja tallennuksen jälkeen voit käyttää sitä sovelluksesta. Kaikilla sovelluksilla voidaan suorittaa toimintoja, mutta vain Laskin- (Calculator) ja Muistiinpanot (Notes) -sovelluksilla voidaan käyttää ohjelmia.

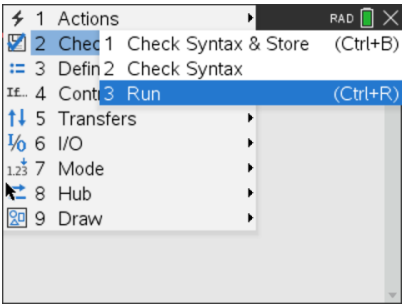
Ohjelman lausekkeet suoritetaan peräkkäisessä järjestyksessä (vaikka jotkut komennot muuttavatkin ohjelmavirtaa). Mahdollinen tulostus näkyy sovelluksen työalueella.

- Ohjelman suorittaminen jatkuu, kunnes se saavuttaa viimeisen lausekkeen tai **Lopeta**-komennon.
- Toimintojen suoritus jatkuu, kunnes se saavuttaa **Return**-komennon.

Ohjelman tai toiminnon suorittaminen Ohjelmaediittorista

1. Varmista, että olet määritellyt ohjelman tai toiminnon, ja Ohjelmaediittori on aktiivinen paneeli (tietokone) tai sivu (kämmenlaite).
2. Klikkaa työkalupalkissa **Asiakirjan työkalut** -painiketta  ja valitse **Tarkista syntaksi ja Tallenna > Suorita**.
   —tai—
   Paina **Ctrl+R**.

   Kämmenlaite: Paina menu 2 3 tai paina ctrl R.



Tästä seuraa automaattisesti:

- syntaksin tarkistus ja ohjelman tai toiminnon tallennus,
- ohjelman tai toiminnon nimen liittäminen Laskin-sovelluksen ensimmäiseen käytettävissä olevaan riviin välittömästi Ohjelmaediittorin jälkeen. Jos kyseisessä sijainnissa ei ole Laskinta, uusi laskin lisätään.

3. Jos ohjelma tai toiminto vaatii sinua toimittamaan yhden tai useamman argumentin, kirjoita arvot tai muuttujan nimet suluissa.
4. Paina enter.

Huomaa: Voit myös suorittaa ohjelman tai toiminnon Laskin- tai Muistiinpanot-sovelluksissa kirjoittamalla ohjelman nimen suluissa tarvittavine argumentteineen ja painamalla enter.

Lyhyiden ja pitkien nimien käyttö

Aina kun sinulla on sama ongelma, jossa kohde on määritelty, voit lähestyä ongelmaa kirjoittamalla lyhyen nimen (annettu nimi objektin **Määrittele**-komennossa). Tämä koskee kaikkia määriteltyjä kohteita, kuten yksityisiä, julkisia ja ei-kirjasto-objekteja.

Voit käyttää kirjasto-objektia mistä tahansa asiakirjasta kirjoittamalla objektin pitkän nimen. Pitkä nimi koostuu objektin kirjastoasiakirjan nimestä, jota seuraa kääntöviiva "\" ja sen jälkeen objektin nimi. Esimerkiksi kirjastoasiakirjassa **lib1** määritelty objektin pitkä nimi **func1** on **lib1\func1**. Jos haluat kirjoittaa kämmenlaitteen "\"-merkin, paina shift ÷.

Huomaa: Jos et muista täsmällistä nimeä tai yksityisen kirjasto-objektin vaatimien argumenttien järjestystä, voit avata kirjastoasiakirjan tai tarkastella objektia Ohjelmaediittorilla. Voit myös käyttää komentoa **getVarInfo** nähdäksesi luettelon kirjaston objekteista.

Julkisen kirjaston ohjelman tai toiminnon käyttäminen

1. Varmista, että olet määritellyt objektin asiakirjan ensimmäisessä ongelmassa, tallentanut objektin, tallentanut kirjastoasiakirjan MyLib-kansioon ja päivittänyt kirjastot uudelleen.
2. Avaa TI-Nspire™ -sovellus, jossa haluat käyttää ohjelmaa tai toimintoa.

   Huomaa: Kaikilla sovelluksilla voidaan arvioida toimintoja, mutta vain Laskin- ja Muistiinpanot-sovelluksilla voidaan suorittaa ohjelmia.
3. Avaa Katalogi ja etsi kirjasto-välilehti etsiäksesi ja lisätäksesi objektin.
   —tai—
   Kirjoita objektin nimi. Jos kyseessä on ohjelma tai toiminto, liitä aina perään nimi suluissa.

   ```
   libs2\func1()
   ```
4. Jos ohjelma tai toiminto vaatii sinua toimittamaan yhden tai useamman argumentin, kirjoita arvot tai muuttujan nimet suluissa.

   ```
   libs2\func1(34,power)
   ```
5. Paina enter.

Yksityisen kirjaston ohjelman tai toiminnon käyttäminen

Käyttääksesi yksityisen kirjaston objektia sinun on tunnettava sen pitkä nimi. Esimerkiksi kirjastoasiakirjassa **lib1** määritelty objektin pitkä nimi **func1** on **lib1\func1**.

Huomaa: Jos et muista täsmällistä nimeä tai yksityisen kirjasto-objektin vaatimien argumenttien järjestystä, voit avata kirjastoasiakirjan tai tarkastella objektia Ohjelmaediittorilla.

1. Varmista, että olet määritellyt objektin asiakirjan ensimmäisessä ongelmassa, tallentanut objektin, tallentanut kirjastoasiakirjan MyLib-kansioon ja päivittänyt kirjastot uudelleen.
2. Avaa TI-Nspire™ -sovellus, jossa haluat käyttää ohjelmaa tai toimintoa.

   Huomaa: Kaikilla sovelluksilla voidaan arvioida toimintoja, mutta vain Laskin- ja Muistiinpanot-sovelluksilla voidaan suorittaa ohjelmia.
3. Kirjoita objektin nimi. Jos kyseessä on ohjelma tai toiminto, liitä aina perään nimi suluissa.

   ```
   libs2\func1()
   ```
4. Jos kohde pyytää sinua syöttämään yhden tai useamman argumentin, kirjoita arvot tai muuttujan nimet suluissa.

```
libs2\func1(34,power)
```

5. Paina enter.

Käynnissä olevan ohjelman tai toiminnon keskeyttäminen

Kun ohjelma tai toiminto on käynnissä, näytetään varattu osoitin.

- Jos haluat pysäyttää ohjelman tai toiminnon,
  - Windows®: Pidä **F12**-näppäintä painettuna ja paina **Enter** toistuvasti.
  - Mac®: Pidä **F5**-näppäintä painettuna ja paina **Enter** toistuvasti.
  - Kämmenkäyttöinen: Pidä on-näppäintä painettuna ja paina enter toistuvasti.

  Näyttöön tulee viesti. Jos haluat muokata ohjelmaa tai toimintoa Ohjelmaediittorissa, valitse **Siirry**. Osoitin tulee näkyviin komennolla, jossa tauko tapahtui.

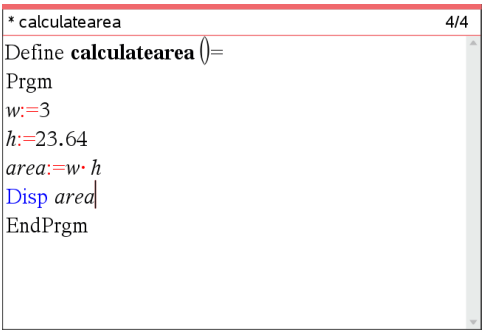
Arvojen hakeminen ohjelmaan

Arvot, joita funktio tai ohjelma käyttää laskutoimituksessa, voidaan hakea usealla eri tavalla.

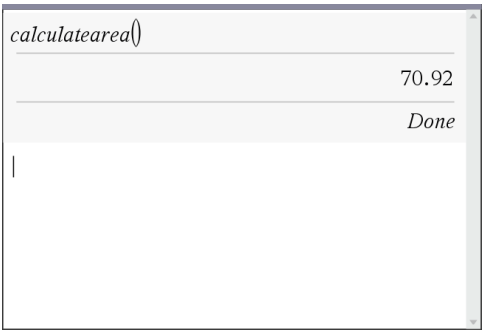
Arvojen upottaminen ohjelmaan tai funktioon

Tämä menetelmä on käyttökelpoinen ensisijaisesti sellaisten arvojen hakemisessa, joiden on oltava samoja aina kuin ohjelmaa tai funktiota käytetään.

1. Määritä ohjelma.



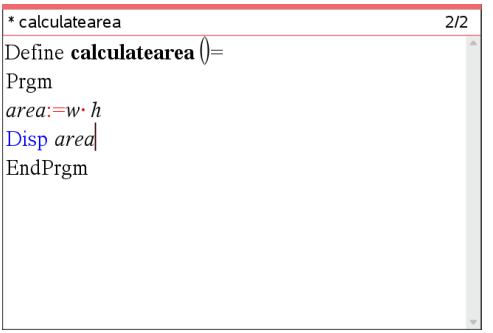
2. Suorita ohjelma.



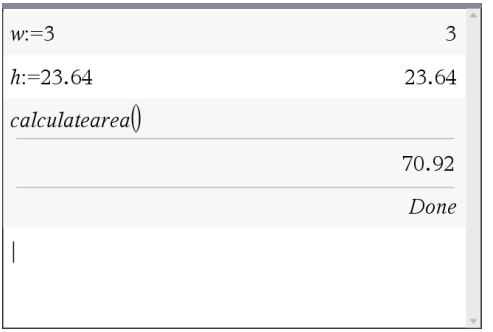
Käyttäjä määrittää arvot muuttujille

Ohjelma tai funktio voi viitata etukäteen määritettyihin muuttujiin. Tässä menetelmässä käyttäjien on muistettava muuttujanimet ja määritettävä niille arvot ennen objektin käyttöä.

1. Määritä ohjelma.



2. Anna muuttujat ja suorita ohjelma.

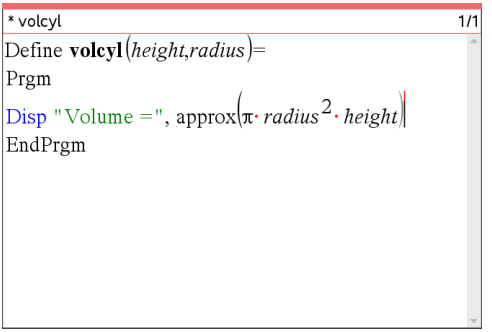


Käyttäjä antaa arvot argumentteina

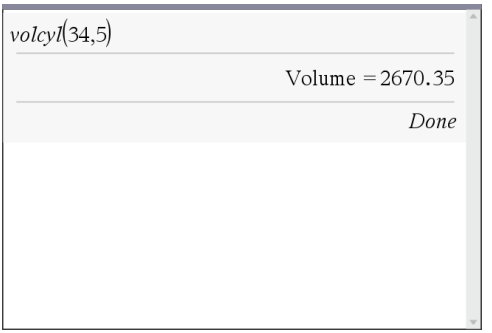
Tässä menetelmässä käyttäjät voivat antaa yhden tai useampia arvoja argumentteina ohjelmaa tai funktiota hakevan lausekkeen sisällä.

Seuraava ohjelma, **volcyl**, laskee lieriön tilavuuden. Käyttäjän on annettava kaksi arvoa: lieriön korkeus ja säde.

1. Määritä **volcyl**-ohjelma.



2. Kun suoritat ohjelman, näet lieriön tilavuuden, jonka korkeus on 34 mm ja säde 5 mm.



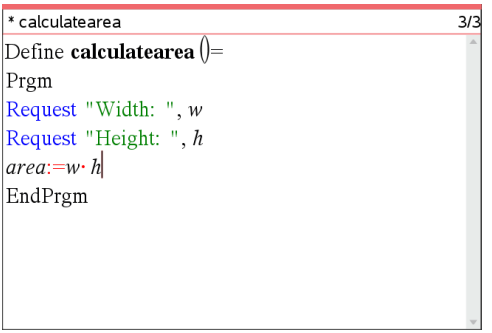
Huomaa: Sinun ei tarvitse käyttää parametrinimiä suorittaessasi **volcyl**-ohjelman, mutta sinun on annettava kaksi argumenttia (arvoina, muuttujina tai lausekkeina). Ensimmäisen arvon on oltava korkeus ja toisen säde.

Arvojen pyytäminen käyttäjältä (vain ohjelmissa)

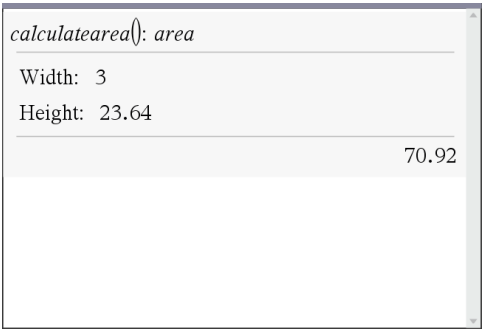
Voit käyttää ohjelmassa **Request**- ja **RequestStr**-komentoja sillä tavoin, että ohjelma keskeytyy ja näyttää valintaruudun, jossa käyttäjältä pyydetään tietoja. Tässä menetelmässä käyttäjien ei tarvitse muistaa muuttujanimiä eikä järjestystä, jossa ne tarvitaan.

Request- ja **RequestStr** -komentoja ei voi käyttää funktiossa.

1. Määritä ohjelma.



2. Suorita ohjelma ja anna tiedot pyydettäessä.

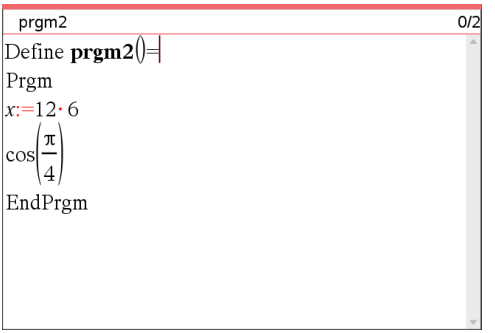


Käytä **RequestStr**-komentoa **Request**-komennon sijaan, kun haluat ohjelman tulkitsevan käyttäjän vastauksen merkkijonona eikä matemaattisena lausekkeena. Tällöin käyttäjän ei tarvitse merkitä vastausta lainausmerkkien (“”) sisään.

Tietojen näyttäminen

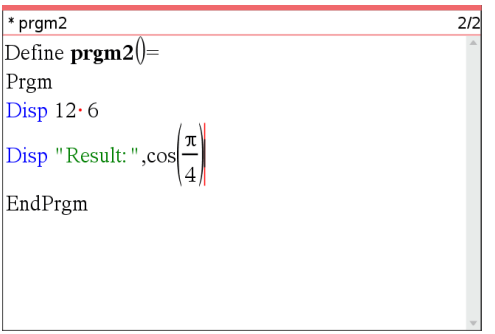
Funktio tai ohjelma ei näytä suorituksen aikana välilaskutuloksia, ellet sisällytä siihen komentoa, joka pyytää näyttämään nämä tulokset. Tässä kohtaa on merkittävä ero suoritettaessa laskutoimitus syöterivillä tai funktion tai ohjelman sisällä.

Esimerkiksi seuraavat laskutoimitukset eivät näy vastauksena funktiossa tai ohjelmassa (vaikka ne näkyvät syöterivillä laskettaessa).



Tietojen näyttäminen historiassa

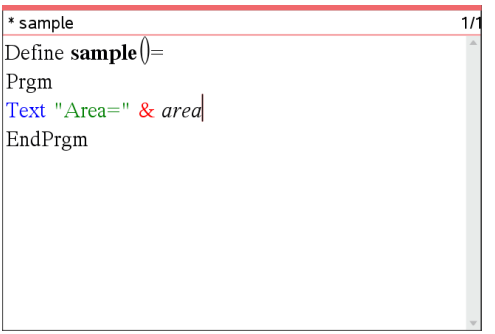
Voit käyttää **Disp**-komentoa ohjelmassa tai funktiossa, jotta tiedot, mukaan lukien välitulokset, näytetään historiassa.



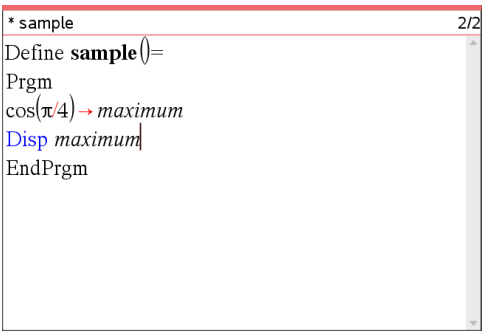
Tietojen näyttäminen valintaruudussa

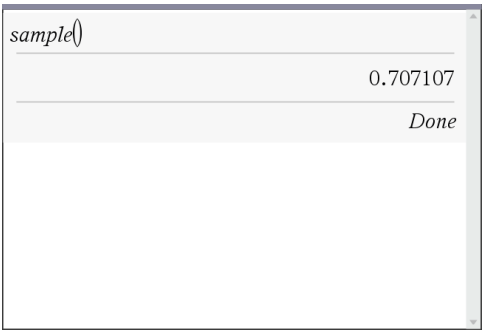
Voit käyttää **Text**-komentoa, jolloin käynnissä oleva ohjelma keskeytyy ja näyttää tiedot valintaruudussa. Käyttäjä voi jatkaa ohjelmaa painamalla **OK**-painiketta tai pysäyttää ohjelman painamalla **Peruuta**-painiketta.

Text-komentoa ei voi käyttää funktiossa.



Huomaa: Tuloksen näyttäminen **Disp**- tai **Text**-komennolla ei tallenna kyseistä tulosta. Jos aiot viitata myöhemmin tulokseen, tallenna se globaaliin muuttujaan.



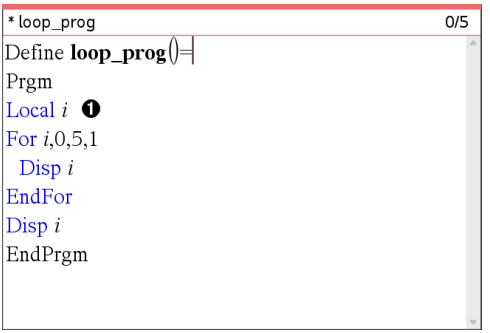


Paikallisten muuttujien käyttö

Paikallinen muuttuja on väliaikainen muuttuja, joka on olemassa vain käyttäjän määrittämän funktion ratkaisun aikana tai käyttäjän määrittämän ohjelman suorituksen aikana.

Esimerkki paikallisesta muuttujasta

Seuraava ohjelmasegmentti sisältää **For...EndFor**-silmukan (jota käsitellään myöhemmin tässä moduulissa). Muuttuja *i* on silmukan laskuri. Useimmissa tapauksissa muuttujaa *i* käytetään vain ohjelman suorittamisen aikana.



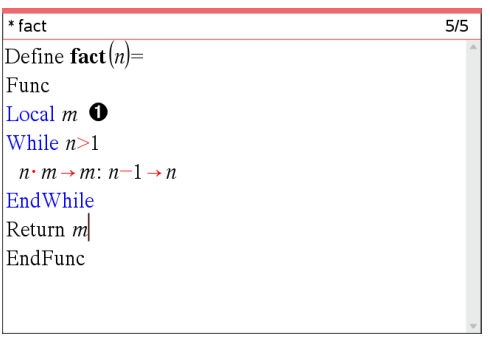
❶ Määrittää muuttujan *i* paikalliseksi.

Huomaa: Mikäli mahdollista, määritä paikallisiksi kaikki muuttujat, joita käytetään vain ohjelman sisällä ja joiden ei tarvitse olla käytettävissä ohjelman pysähtymisen jälkeen.

Mikä aiheuttaa määrittämättömän muuttujan virheilmoituksen?

Määrittämätön muuttuja -virheilmoitus tulee näkyviin, kun ratkaiset käyttäjän määrittämän funktion tai suoritat käyttäjän määrittämän ohjelman, jossa viitataan alustamattomaan paikalliseen muuttujaan (jolle ei ole määritetty arvoa).

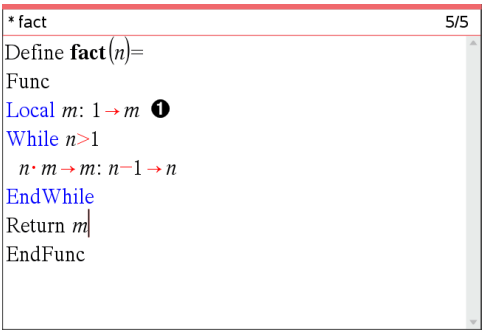
Esimerkki:



❶ Paikalliselle muuttujalle *m* ei ole määritetty alkuarvoa.

Paikallisten muuttujien initialisointi

Kaikille paikallisille muuttujille on määritettävä alkuarvo, ennen kuin niihin viitataan.



❶ 1 tallennetaan alkuarvoksi muuttujalle *m*.

Huomaa (CAS): Funktiot ja ohjelmat eivät pysty käyttämään paikallista muuttujaa symbolisten laskutoimitusten suorittamiseksi.

CAS: Symbolisten laskutoimitusten suorittaminen

Jos haluat funktion tai ohjelman suorittavan symbolisia laskutoimituksia, sinun on käytettävä globaalia muuttujaa paikallisen muuttujan sijaan. Sinun on kuitenkin varmistettava, että globaali muuttuja ei ole jo olemassa ohjelman ulkopuolella. Seuraavista menetelmistä voi olla apua.

- Viittaa globaalin muuttujan nimeen, tyypillisesti kaksi tai useampia merkkejä, jota ei todennäköisesti ole olemassa funktion tai ohjelman ulkopuolella.
- Sisällytä **DelVar**-komento ohjelmaan poistaaksesi globaalin muuttujan, mikäli se on olemassa, ennen kuin viittaat siihen. (**DelVar** ei poista lukittuja tai linkitettyjä muuttujia.)

Funktioiden ja ohjelmien väliset erot

Ohjelmaeditorissa määritetty funktio on samanlainen kuin TI-Nspire™ -ohjelmiston sisältämät funktiot.

- Funktioiden tulee antaa tulos, joka voidaan piirtää tai syöttää taulukkoon. Ohjelmat eivät pysty antamaan vastausta.
- Voit käyttää funktiota (mutta et ohjelmaa) lausekkeen sisällä. Esimerkki: **3 • func1 (3)** on kelvollinen lauseke, mutta **3 • prog1(3)** ei ole.
- Voit suorittaa ohjelmia vain Laskin- ja Muistiinpanot-sovelluksista. Voit kuitenkin arvioida funktioita Laskimessa, Muistiinpanoissa, Listat & taulukoissa, Kaaviot & geometriassa sekä Data & tilastoissa.
- Funktio voi viitata mihin tahansa muuttujaan, mutta se voi tallentaa arvon vain paikalliseen muuttujaan. Ohjelmiin voidaan tallentaa sekä paikallisia että globaaleja muuttujia.

Huomaa: Argumentteja, joita käytetään arvojen siirtämisessä funktioon, käsitellään automaattisesti paikallisina muuttujina. Jos haluat tallentaa arvoja muihin muuttujiin, sinun on annettava niille määrite **Local (Paikallinen)** funktion sisältä käsin.

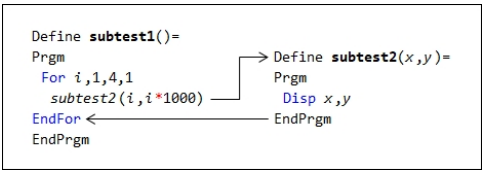
- Funktio ei voi hakea ohjelmaa aliohjelmana, mutta se voi hakea toisen käyttäjän määrittämän funktion.
- Ohjelmaa ei voi määrittää funktion sisään.
- Funktio ei voi määrittää globaalia funktiota, mutta se voi määrittää paikallisen funktion.

Ohjelman hakeminen toisen ohjelman sisältä

Ohjelma voi hakea toisen ohjelman aliohjelmana. Aliohjelma voi olla ulkoinen (erillinen ohjelma) tai sisäinen (sisältyy pääohjelmaan). Aliohjelmat ovat hyödyllisiä, kun ohjelman on suoritettava toistuvasti sama komentoryhmä useissa eri paikoissa.

Erillisen ohjelman hakeminen

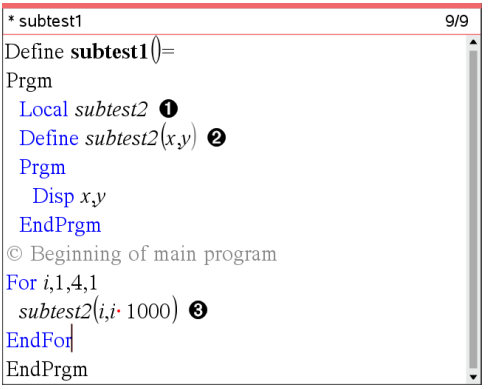
Kun haluat hakea erillisen ohjelman, käytä samaa syntaksia kuin suorittaessasi ohjelman syöteriviltä.



Sisäisen aliohjelman määrittäminen ja hakeminen

Kun haluat määrittää sisäisen aliohjelman, käytä **Define**-komentoa ja silmukkaa **Prgm...EndPrgm**. Koska aliohjelma on määritettävä ennen kuin sen voi hakea, hyvä käytäntö on määrittää aliohjelmat pääohjelman alussa.

Sisäinen aliohjelma haetaan ja suoritetaan samalla tavalla kuin erillinen ohjelma.



❶ Määrittää aliohjelman paikalliseksi muuttujaksi.

❷ Määrittää aliohjelman.

❸ Hakee aliohjelman.

Huomaa: Siirry ohjelmaeditorin **Var**-valikon kautta **Define**- ja **Prgm...EndPrgm**-komentoihin.

Aliohjelmien käyttöön liittyviä huomautuksia

Aliohjelman lopussa ohjelman suoritus palaa hakevaan ohjelmaan. Voit poistua aliohjelmasta muina ajankohtina käyttämällä **Return**-komentoa ilman argumenttia.

Aliohjelma ei voi käyttää hakevassa ohjelmassa määritettyjä paikallisia muuttujia. Samaten hakeva ohjelma ei voi käyttää aliohjelmassa määritettyjä paikallisia muuttujia.

Lbl-komennot ovat paikallisia ohjelmille, joissa ne sijaitsevat. Niinpä hakevassa ohjelmassa oleva **Goto**-komento ei voi haarautua aliohjelmassa olevaan tunnukseen tai päin vastoin.

Kehämääritysvirheiden välttäminen

Kun lasket käyttäjän määrittämän funktion tai suoritat ohjelman, voit määrittää argumentin, joka sisältää saman muuttujan, jota on käytetty funktion määrittämisessä tai ohjelman luomisessa. Välttääksesi kehämääritysvirheitä sinun on kuitenkin määritettävä arvo muuttujille, joita käytetään funktion ratkaisemisessa tai ohjelman suorittamisessa. Esimerkki:

$x+1 \rightarrow x$ ❶

– tai –

```
For i,i,10,1
  Disp i ❶
EndFor
```

❶ Aiheuttaa **kehämääritys**virheilmoituksen, jos x:llä tai i:llä ei ole arvoa. Virhettä ei esiinny, jos x:lle tai i:lle on jo määritetty arvo.

Funktion tai ohjelman suorituksen kontrollointi

Kun suoritat ohjelman tai ratkaiset funktion, ohjelmarivit suoritetaan järjestyksessä. Jotkin komennot kuitenkin muuttavat ohjelman suorituksen kulkua. Esimerkki:

- Kontrollirakenteet, kuten **If...EndIf**-komennot, käyttävät ehtotestiä määrittääkseen, mikä osa ohjelmasta tulee suorittaa.
- Silmukkakomennot, kuten **For...EndFor**, toistavat komentojen ryhmää.

Komentojen If, Lbl ja Goto käyttäminen ohjelman suorittamisen kontrolloinnissa

If-komennon ja useiden **If...EndIf**-rakenteiden avulla voit laskea lausekkeen tai lausekelohkon ehdollisesti eli testin tulokseen perustuen (kuten x>5). **Lbl** (tunnus)- ja **Goto**-komentojen avulla voit haarautua tai hypätä paikasta toiseen funktiossa tai ohjelmassa.

If-komento ja useat **If...EndIf**-rakenteet sijaitsevat ohjelmaeditorin **Control (Kontrolli)** -valikossa.

Kun lisäät rakenteen, kuten **If...Then...EndIf**, kohdistimen kohdalle lisätään malli. Kohdistin sijoittuu siten, että voit syöttää ehtotestin.

If-komento

Kun haluat suorittaa yhden komennon ehtotestin ollessa tosi, käytä yleistä muotoa:

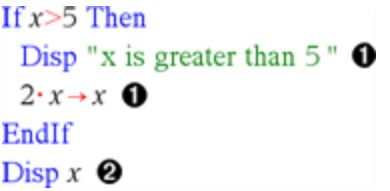
❶ Suoritetaan vain, jos x>5; muussa tapauksessa ohitetaan.

❷ Näyttää aina x:n arvon.

Tässä esimerkissä sinun on tallennettava arvo x:lle ennen **If** -komennon suorittamista.

If...Then...EndIf-rakenteet

Kun haluat suorittaa yhden komentojen ryhmän, jos ehtotesti on tosi, käytä rakennetta:



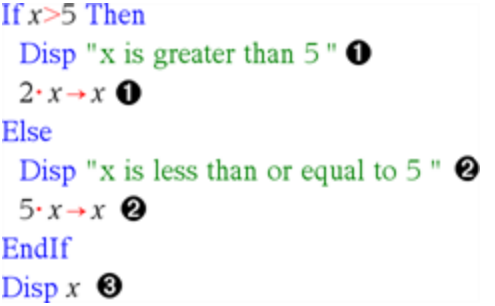
❶ Suoritetaan vain, jos x>5.

❷ Näyttää seuraavien arvon:
2x, jos x>5
x, jos x≤5

Huomaa: EndIf merkitsee loppukohdan **Then**-lohkolle, joka suoritetaan, jos ehto on tosi.

If...Then...Else... EndIf-rakenteet

Kun haluat suorittaa yhden komentojen ryhmän, jos ehtotesti on tosi, ja toisen ryhmän, jos ehto on epätosi, käytä seuraavaa rakennetta:



❶ Suoritetaan vain, jos x>5.

❷ Suoritetaan vain, jos x≤5.

❸ Näyttää seuraavien arvon:
2x, jos x>5
5x, jos x≤5

If...Then...ElseIf... EndIf-rakenteet

Monimutkaisemman If-komennon avulla voit testata useita ehtoja. Oletetaan, että haluat ohjelman testaavan käyttäjän syöttämää argumenttia, joka viittaa yhteen neljästä vaihtoehdosta.

Testataksesi jokaisen vaihtoehdon (If Vaihtoehto=1, If Vaihtoehto=2 ja niin edelleen) käytä **If...Then...ElseIf...EndIf**-rakennetta.

Lbl- ja Goto-komennot

Voit kontrolloida ohjelman suoritusta myös **Lbl** (tunnus)- ja **Goto**-komentojen avulla. Nämä komennot sijaitsevat ohjelmaeditorin **Transfers (Siirrot)** -valikossa.

Lbl-komennon avulla voit merkitä tietyn kohdan funktiossa tai ohjelmassa (määrittää nimen).

Lbl *labelName*	tälle paikalle määritettävä nimi (käytä samaa nimeämistapaa kuin muuttujan nimille)

Sen jälkeen voit käyttää **Goto**-komentoa missä tahansa funktion tai ohjelman kohdassa haaroittaaksesi funktion/ohjelman määritettyä tunnusta vastaavaan kohtaan.

Goto *labelName*	määrittää, mihin **Lbl**-komentoon siirrytään

Koska **Goto**-komento on ehdoton (se haarautuu aina määritettyyn tunnukseen), sitä käytetään usein yhdessä **If**-komennon kanssa, jotta voit määrittää ehtotestin. Esimerkki:

```
If x>5
  Goto GT5 ❶
Disp x

.............

Lbl GT5 ❷
Disp "The number was > 5"
```

❶ Jos x>5, haarautuu suoraan tunnukseen GT5.

❷ Tässä esimerkissä ohjelman tulee sisältää komentoja (kuten **Stop**), jotka estävät lausekkeen **Lbl** GT5 suorittamisen, jos x≤5.

Silmukoiden käyttäminen komentoryhmän toistamiseksi

Voit toistaa samaa komentoryhmää peräkkäin käyttämällä jotakin silmukkarakenteista. Käytettävissä on useita erilaisia silmukkatyyppejä. Jokainen niistä suorittaa silmukan eri tavalla ehtotestin mukaisesti.

Silmukkakomennot ja silmukoihin liittyvät komennot sijaitsevat ohjelmaeditorin **Control (Kontrolli)**- ja **Transfers (Siirrot)** -valikoissa.

Kun lisäät jonkin silmukkarakenteen, sen malli lisätään kohdistimen kohdalle. Sen jälkeen voit aloittaa silmukassa suoritettavien komentojen syöttämisen.

For...EndFor-silmukat

For...EndFor-silmukka käyttää laskuria kontrolloidakseen silmukan toistokertojen määrää. **For**-komennon syntaksi on:

Huomaa: Loppuarvon on oltava pienempi kuin alkuarvo, mikäli lisäys on negatiivinen.

For *muuttuja*, *alku*, *loppu* [, *lisäys*]
❶ ❷ ❸ ❹

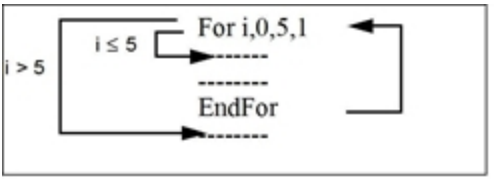
❶ *Muuttuja*, jota käytetään laskurina

❷ Laskurin arvo, jota käytetään, kun **For** suoritetaan ensimmäisen kerran

❸ Poistuu silmukasta, kun *muuttuja* ylittää tämän arvon

❹ Lisätään laskuriin aina, kun **For** suoritetaan seuraavan kerran (Jos tämä valinnainen arvo jätetään pois, *lisäys* on 1.)

Kun **For** suoritetaan, *muuttujan* arvoa verrataan *loppu*arvoon. Jos *muuttuja* ei ylitä *loppu*arvoa, silmukka suoritetaan; muussa tapauksessa kontrolli hyppää **EndFor**-komentoa seuraavaan komentoon.



Huomaa: For-komento lisää automaattisesti laskurin muuttujan arvoa, jotta funktio tai ohjelma voi poistua silmukasta tietyn toistokertojen määrän jälkeen.

Silmukan (**EndFor**) lopussa kontrolli hyppää takaisin **For**-komentoon, jossa muuttujan arvoa lisätään ja sitä verrataan *loppu*arvoon.

Esimerkki:

```
For i,0,5,1
  Disp i ❶
EndFor
Disp i ❷
```

❶ Näyttää vastauksena 0, 1, 2, 3, 4 ja 5.

❷ Näyttää vastauksen 6. Kun *muuttujan* arvo on lisääntynyt lukuun 6, silmukkaa ei suoriteta.

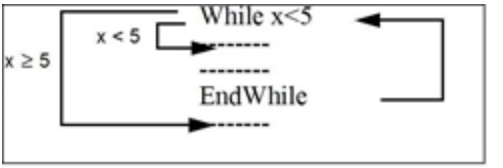
Huomaa: Voit määrittää laskurin muuttujan paikalliseksi, jos sitä ei tarvitse tallentaa funktion tai ohjelman päättymisen jälkeen.

While...EndWhile-silmukat

While...EndWhile-silmukka toistaa komentolohkoa niin kauan kuin määritetty ehto on tosi. **While**-komennon syntaksi on:

While -*ehto*

While-komentoa suoritettaessa määritetään *ehdon* totuusarvo. Jos *ehto* on tosi, silmukka suoritetaan; muussa tapauksessa kontrolli hyppää **EndWhile**-komentoa seuraavaan komentoon.



Huomaa: While-komento ei muuta ehtoa automaattisesti. Sinun on syötettävä komennot, jotka mahdollistavat funktion tai ohjelman poistumisen silmukasta.

Silmukan (**EndWhile**) lopussa kontrolli hyppää takaisin **While**-komentoon, jossa ehdon totuusarvo määritetään uudelleen.

Jotta silmukka voidaan suorittaa ensimmäisen kerran, ehdon on oltava aluksi tosi.

- Ehtoon viittaavat mahdolliset muuttujat on asetettava ennen **While**-komentoa. (Voit syöttää arvot valmiiksi funktioon tai ohjelmaan tai voit kehottaa käyttäjää syöttämään arvot.)
- Silmukan tulee sisältää komennot, jotka muuttavat ehdon arvoja siten, että se on lopuksi epätosi. Muussa tapauksessa ehto on aina tosi, ja funktio tai ohjelma ei voi poistua silmukasta (= ikuinen silmukka).

Esimerkki:

```
0→x ❶
While x<5
  Disp x ❷
  x+1→x ❸
EndWhile
Disp x ❹
```

❶ Asettaa aluksi x:n arvon.

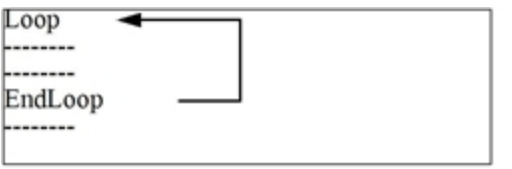
❷ Näyttää vastauksena 0, 1, 2, 3 ja 4.

❸ Lisää x:n arvoa.

❹ Näyttää vastauksen 5. Kun x:n arvo on lisääntynyt lukuun 5, silmukkaa ei suoriteta.

Loop...EndLoop-silmukat

Loop...EndLoop luo ikuisen silmukan, jota toistetaan loputtomasti. **Loop**-komennossa ei ole argumentteja.



Tyypillisesti silmukkaan lisätään komennot, jotka sallivat ohjelman poistua silmukasta. Yleisesti käytettyjä komentoja ovat: **If**, **Exit**, **Goto** ja **Lbl** (tunnus). Esimerkki:

❶ **If**-komento tarkistaa ehdon.

❷ Poistuu silmukasta ja hyppää tähän, kun x on lisääntynyt arvoon 6.

Huomaa: Exit-komento poistuu nykyisestä silmukasta.

Tässä esimerkissä **If**-komento voi olla missä tahansa kohtaa silmukassa.

Kun If-komento on:	Silmukka on:
Silmukan alussa	Suoritetaan vain, jos ehto on tosi.
Silmukan lopussa	Suoritetaan vähintään kerran ja toistetaan vain, jos ehto on tosi.

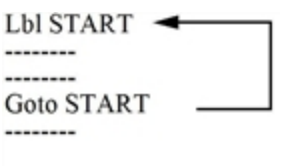
If-komento voi käyttää myös **Goto**-komentoa ohjelman kontrollin siirtämiseksi määritettyyn **Lbl** (tunnus) -komentoon.

Silmukan toistaminen välittömästi

Cycle-komento siirtää ohjelman kontrollin välittömästi seuraavaan silmukan iteraatioon (ennen kuin nykyinen iteraatio on suoritettu loppuun). Tämä komento toimii silmukoiden **For...EndFor**, **While...EndWhile** ja **Loop...EndLoop** kanssa.

Lbl- ja Goto-silmukat

Vaikka **Lbl** (tunnus)- ja **Goto**-komennot eivät ole tarkasti ottaen silmukkakomentoja, niiden avulla voidaan luoda ikuinen silmukka. Esimerkki:



Kuten **Loop...EndLoop**-silmukan, tämänkin silmukan tulee sisältää komennot, jotka sallivat funktion tai ohjelman poistua silmukasta.

Tila-asetusten muuttaminen

Funktioissa ja ohjelmissa tietyt laskenta- tai tulostilat voidaan asettaa väliaikaisesti **setMode()**-funktion avulla. Oikeaan syntaksiin voidaan siirtyä helposti ohjelmaeditorin **Mode (Tila)** -valikosta tarvitsematta muistaa numerokoodeja.

Huomaa: Funktion tai ohjelman määritelmän sisällä tehdyt tilamuutokset eivät ole voimassa funktion tai ohjelman ulkopuolella.

Tilan asettaminen

1. Sijoita kohdistin paikkaan, johon haluat lisätä **setMode**-funktion.
2. Valitse **Tila**-valikosta muutettava tila ja valitse uusi asetus.

   Oikea syntaksi lisätään kohdistimen kohdalle. Esimerkki:

```
setMode(1,3)
```

Ohjelmavirheiden etsiminen ja virheiden käsittely

Kirjoitettuasi funktion tai ohjelman voit etsiä ja korjata virheet usealla eri tavalla. Voit myös luoda virheenkäsittelykomennon itse funktioon tai ohjelmaan.

Jos funktio tai ohjelma sallii käyttäjän valita usean eri vaihtoehdon joukosta, suorita funktio/ohjelma ja testaa jokainen vaihtoehto.

Virheiden poistomenetelmät

Ajonaikaiset virheilmoitukset voivat löytää syntaksivirheitä, mutta eivät ohjelmalogiikan virheitä. Seuraavat menetelmät voivat olla hyödyllisiä.

- Lisää väliaikaisesti **Disp**-komentoja kriittisten muuttujien arvojen näyttämiseksi.
- Voit varmistaa, että silmukan suorituskertojen määrä on oikea **Disp**-komennon avulla, joka näyttää laskurin muuttujan tai ehtotestin arvot.
- Voit varmistaa aliohjelman suorittamisen **Disp**-komennon avulla, joka näyttää esimerkiksi viestit “Siirrytään aliohjelmaan” ja “Poistutaan aliohjelmasta” aliohjelman alussa ja lopussa.
- Ohjelman tai funktion pysäyttäminen manuaalisesti:
  - **Windows®:** Pidä **F12**-näppäintä painettuna ja paina toistuvasti **Enter**.
  - **Macintosh®:** Pidä **F12**-näppäintä painettuna ja paina toistuvasti **Enter**.
  - **Kämmenlaite:** Pidä [on]-näppäintä painettuna ja paina toistuvasti [enter].

Virheenkäsittelykomennot

Komento	Kuvaus
Try...EndTry	Määrittää lohkon, joka sallii funktion tai ohjelman suorittaa komennon ja tarvittaessa poistua tämän komennon luomasta virheestä.
ClrErr	Tyhjentää virhetilan ja asettaa järjestelmämuuttujan *errCode arvoksi*

Komento	Kuvaus
	nolla. Katso esimerkki errCode-muuttujan käytöstä hakuteoksen kohdasta ***Try-komento****.*
PassErr	Ohittaa virheen siirtyen **Try...EndTry**-lohkon seuraavalle tasolle.

Yleistä

Online-tuki

education.ti.com/eguide

Valitse maasi, niin näet lisää tuotetietoja.

Ota yhteyttä TI-tukeen

education.ti.com/ti-cares

Valitse maasi, niin näet teknisiä tietoja ja muita tukiresursseja.

Huolto- ja takuutiedot

education.ti.com/warranty

Valitse maasi, niin saat tietoa takuun kestosta ja ehdoista tai tuotepalvelusta.

Rajoitettu takuu. Tämä takuu ei vaikuta lainmukaisiin oikeuksiisi.

Texas Instruments Incorporated

12500 TI Blvd.

Dallas, TX 75243