Kapitel 3: Ljusintensitet, IF och WHILE — Övning 1: Mäta ljusintensitet

Syfte:
- Läsa av ljusintensitetsgivaren
- Introducera **While** loopen
- Använda **Output**(-satsen
- Introducera **toString**(och sammanlänkning

I denna första aktivitet för kapitel 3 ska vi undersöka den inbyggda ljusintensitetsgivaren och introducera **Output**(-satsen i TI-Basic som illustrerar en teknik att visa tal som har olika längd (olika antal siffror).

I de tidigare aktiviteterna har vi bara skickat instruktioner till hubben för att påverka dess inbyggda utrustning (LIGHT, COLOR och SOUND).

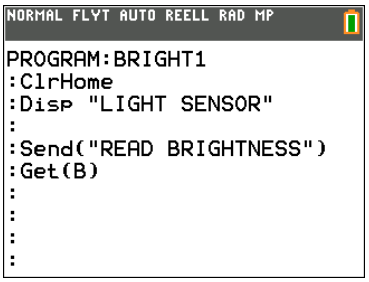
I denna enhet ska vi arbeta med den inbyggda ljusgivaren och använda ett värde därifrån i vårt program för att bygga en ”ljusmätare”. Ljussensorn producerar värden i intervallet 0 till 100 i decimalform.

För att erhålla värdet på ljusintensiteten från hubben behövs TVÅ satser:

- **Send(“READ BRIGHTNESS”)**
- **Get**(<var>**)**

Konfigurera programmet:

1. Starta ett nytt program och ge det namnet BRIGHT1.
2. Lägg till kommandona **ClrHome** och **Disp** för att visa programrubriken. Se skärmbilden till höger.
3. Tryck [prgm] och gå med piltangenten till ***HUB***-menyn.
4. Välj **2: Send(“READ…** och sedan **1: BRIGHTNESS**.
5. Tryck [prgm] och gå med piltangenten till ***HUB*** menyn.
6. Välj **5:Get(.** Mata in variabeln **B** och en högerparentes.



Hur det fungerar:

- **READ BRIGHTNESS** ger information till hubben att läsa nivån på ljus-intensiteten och lagra det värdet i ett inbyggt buffertminne.
- **Get(B)** är ett kommando för att erhålla ett värde från hubben. Denna sats överför värdet i buffertminnet hos hubben till variabeln B hos TI-84 Plus CE-T. B kan ersättas av någon av grafräknarens numeriska variab-
  ler, A...Z och Θ (theta).

Lärarkommentar: En "buffert" är en minnesplats hos hubben som tillfälligt lagrar ett värde. Den uppdateras närhelst ett annat *READ* kommando bearbetas, så vi rekommenderar starkt att *READ* och omedelbart *Få* (*Get*) värdet från hubben till en variabel på grafräknaren. Det är möjligt att samla in en uppsättning data från hubben och lagra dessa data i en lista för framtida dataanalys, men det ligger utanför ramen för denna introduktion.

While-loop

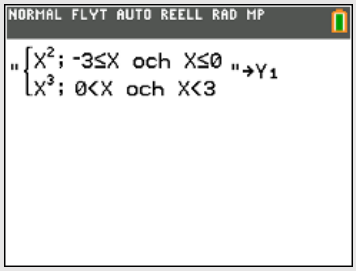
While…End-loopen (**CTL**-menyn i programeditorn) används för att processa ett block av kod (en följd av kodrader) medan (**while**) ett *villkor* är sant. Ett villkor är

ett logiskt påstående som kan bedömas som sant eller falskt. Relationsoperatorer och logiska operatorer finns i **test**-menyn hos räknaren (tryck [2nd] [math]).

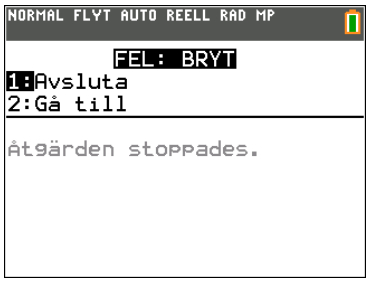
- Relationsoperatorer är **=**, **≠**, **<**, **>**, **≤**, och **≥**.
- Logiska operatorer är **and**, **or**, **not**, och **xor** (på svenska **och, eller, inte** och **xeller**

Dessa operatorer kan användas tillsammans för att bygga upp sammansatta villkor som **x>0 och y>0.**

Lärarkommentar: I testmenyn finns en undermeny VILLKOR som innehåller olika symboler som kan användas för att skapa styckvisa funktioner med den speciella mallen för styckvisa funktioner. Tryck [math], välj **MA** i menyn och sedan alternativet **styckvis**.



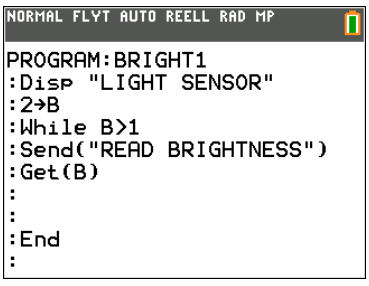
Vi ska nu använda en enkel **While**-loop som stoppar när ljusintensitetsvärdet är mindre än 1. För att avsluta programmet kan du helt enkelt täcka för givaren med din hand.



Ett annat sätt att avbryta eller avsluta en pågående programkörning är att trycka på tangenten [on]. Då får du ett felmeddelande: **FEL: BRYT** i skärmens överkant och du har sedan möjlighet att **Avsluta** programkörningen och återgå till startskärmen eller **Gå till** programeditorn och den plats i programmet där tryckningen på tangenten [on] stoppade programkörningen. Det är också ett bekvämt sätt att fortsätta med redigeringen av programmet.

Lägga till en While-loop

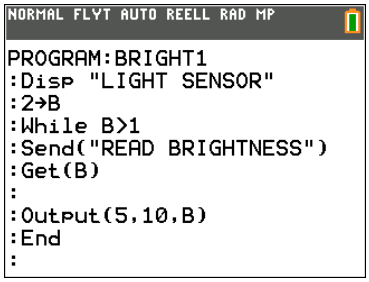
1. Innan **Send(**-satsen i ditt program ska du lägga till satserna:
   2→B
   While B>1 (använd [test]-menyn för tecknet **>**)
   - Dessa satser startar (initierar) loopen. Så länge som villkoret B>1 är *sant* kommer loopen att fortsätta att läsa av ljusgivaren. Så fort villkoret blir falskt, t.ex. om du täcker givaren med din hand så att inget ljus träffar den, kommer loopen och programmet att avslutas.
2. Slutet (**End**) av **While**-loopen måste också skrivas in. Lägg till en **End**-sats nedanför **Get(**-satsen. End finns i CTL-menyn.



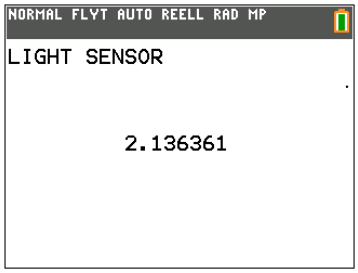
Lärarkommentar: Det är alltid en bra idé att ge dina program ett sätt att komma ur en loop. Kom ihåg att ***End*** inte är slutet på programmet utan snarare slutet på en kontrollstruktur. (***If…Then…End***, ***For…End***, ***While…End***, and ***Repeat…End***). I större program finns det vanligtvis många förekomster av End-satser. Programprocessorn i grafräknaren "vet" vilket **End** som tillhör en viss kontrollstruktur men det är upp till programmeraren att konstruera den korrekta koden.

3. Lägg till **Output(**-satsen *efter* **Get(**-satsen och *före* **End** i loopen. Tryck [prgm] och gå med piltangenten till **I/O** i menyn och väljs där **6: Output(**.
   - **Output(**-satsen ger dig utmärkt kontroll *var* på startskärmen något ska visas. Strukturen hos satsen är: **Output(**<rad#>, <kolumn#>, <sträng eller variabel>).

   Exempel:
   - **Output(3,7,"HELLO")** placerar bokstaven "H" på rad 3, kolumn 7 på startskärmen och resten av bokstäverna i ordet följer efter "H".
   - **Output(5,10,B)** placerar värdet på variabeln B med start på rad 5, kolumn 10 på startskärmen och resten av siffrorna följer därefter.



4. Avsluta programeditorn och kör programmet med hubben ansluten.
   - Du kommer att se programrubriken högst upp på skärmen och ett värde, som visar den avlästa ljusintensiteten, mitt på skärmen. Tyvärr är det så att de värden du ser inte är korrekta!

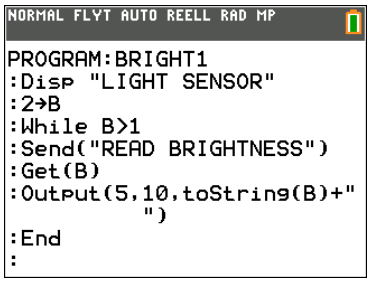


Output(-satsen tar inte bort avslutande siffror när ett kortare tal (färre siffror) visas efter ett längre tal (fler siffror). Om t.ex. ett värde 1.23456 visas och nästa värde är 55, så kommer du att se 5523456 på skärmen. Vår slutliga modifiering av programmet visar på ett listigt sätt att korrigera för detta problem.
För att korrigera för detta problem så konverterar man variabeln B till en sträng och lägger till lite extra mellanslag I slutet av strängen för att helt och hållet få bort det föregående visade värdet. Vi använder då i Output-satsen kommandot

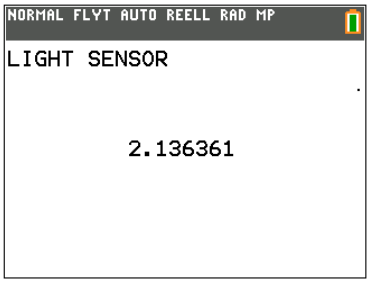
toString(som du hittar i programeditorns I/O-meny.

Den slutliga Output-satsen är då: **Output(5,10,toString(B)+" ")**

OBS! Det ska vara ca 10 mellanslag (tryck på [alpha][0]) mellan citattecknen. Se skärmbilden till höger.



När du nu kör programmet kommer du att se att vissa värden är kortare än andra beroende på att tomutrymmet som vi adderade till strängrepresentationen hos B tar bort de föregående siffrorna.



Sammanfogning:
"**+**"-tecknet i **Output(**-satsen används *inte* för addition utan det används för att sammanfoga (kombinera) två strängar (teckenuppsättningar). Tomutrymmet inom citattecken *läggs till* i slutet *av* strängrepresentationen av variabeln B.

Lärarkommentarer: Strängmanipulation krävs inte för dessa övningar, men det bör vara klart från detta exempel att om man vet vilka funktioner som finns i ett programmeringsspråk så förbättrar kodningsfärdigheterna.