Kapitel 2: Använda variabler och uttryck — Övning 3: Mata in ett värde för en variabel

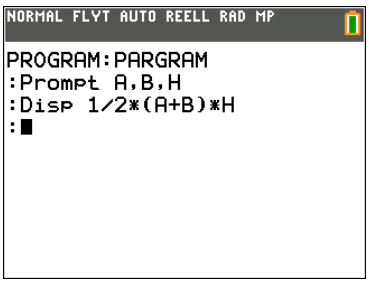
I denna tredje aktivitet för kapitel 2 kommer du att lära dig använda uttryck och lagra värden i variabler inom program.

Syfte:

- Lära dig hur man programmerar matematiska uttryck.
- Förstå i vilken ordning matematiska operationer utförs.
- Förstå skillnaden mellan matematiska variabler och datavariabler.
- Evaluera **uttryck.**
- **Lagra resultat** av uttryck i variabler.

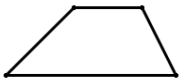
Uttryck

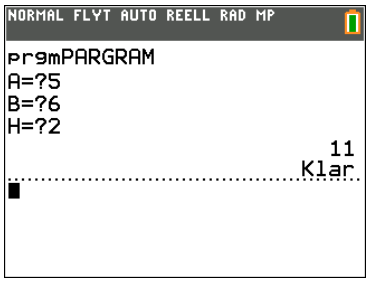
A² och **A+B** t.ex. brukar vi kalla uttryck. Uttryck förekommer som **matematiska** formler. Till exempel är **formeln** för arean av en triangel $A = 1/2 \times B \times H$. **Uttrycket** är $1/2 \times B \times H$.



Ett program evaluerar ett uttryck genom att använda alla nuvarande värden på alla variabler och ger resultatet som ett numeriskt värde. Uttryck evalueras med regler som avgör i vilken ordning olika delar av ett matematiskt uttryck ska beräknas.

Försök nu med programmet till höger som beräknar arean av en parallelltrapets med baserna **A** och **B** och höjden **H**.





Du kan inte använda variabler som B1 och B2. Räknaren beräknar dessa uttryck som B × 1 och B × 2 och det kan orsaka fel. Du kan heller inte använda variabler med mer än en bokstav som t.ex. AB. AB betyder ju faktiskt A × B. (Detta kallas underförstådd multiplikation.)

Matematiska uttryck och uttryck inom programmering

Det finns många likheter mellan uttryck i matematik och i dataprogram men det finns också viktiga skillnader. Den mest framträdande skillnaden är att i ett matematiskt uttryck så står variablerna för okända tal och kan ersättas med tal när det behövs. I ett utryck inom programmering är variablerna namn för tal.

I matematik använder vi formler för att uttrycka ett samband mellan saker som area och längd. I program använder vi uttrycken för att göra beräkningarna och variabelvärdena används för att beräkna ett resultat. När vi *skriver in* programmet så skriver vi in uttrycket men när vi *kör* programmet beräknas uttrycket och skapar ett resultat som ska användas senare.

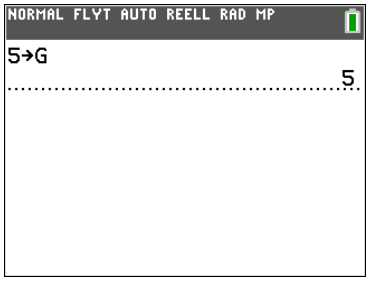
Lärarkommentar: En av de mer förbryllande satserna i nybörjarprogram är x+1→x. I syntaxen för TI-Basic är det alldeles tydligt att 1 adderas till variabeln x och sedan lagras i variabeln x. x på den vänstra sidan och x på den högra sidan representerar olika värden. Denna sats brukar kallas ett ”räkneverk” därför att varje gång den processas (i en loop) ökar den x med 1. I andra programspråk som B.A.S.I.C och Lua (och många fler) så skrivs satsen som x=x+1 vilket är alldeles klart ett falskt påstående i matematisk mening men det fungerar bra i ett program!

Lagra värden i variabler: Tilldelningssats

Operatorn [sto→] används för att lagra (tilldela) resultatet av ett uttryck till en variabel.

Om du trycker på tangenten [sto→] visas alltid symbolen →.

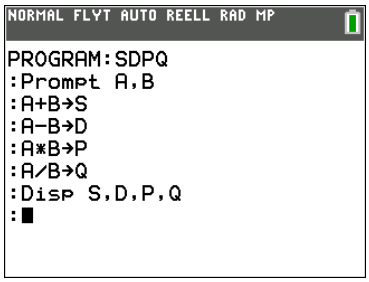
När du har tryckt på [enter] visar startskärmen resultatet av uttrycket och variabeln **G** innehåller nu värdet 5. Framför pilen måste det vara ett **värde** eller ett **uttryck** som ger ett **värde**. Detta kallas tilldelningssatsen eftersom den tilldelar ett värde till en variabel. Symbolen efter pilen måste vara en **variabel.**



Programmering med tilldelningssatser

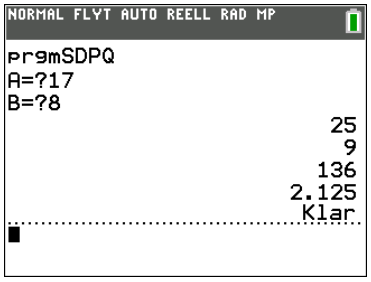
Vi ska nu skriva ett program som ber dig mata in två tal, lagra deras summa, differens, produkt och kvot i fyra variabler och sedan visa resultatet.

1. Starta ett nytt program. Vårt programnamn är SDPQ.
2. ”Prompta” för två variabler A och B.
3. Lagra de fyra uttrycken i fyra andra variabler S, D, P, and Q.
4. Visa S, D, P och Q.
5. Kör programmet.



Mata in ett värde för A och ett annat värde för B.

Använd nu tal där du lätt kan kontrollera att beräkningarna utförs korrekt. Det blir ett slags test att allt fungerar.



***Obs**: TI Basics tilldelningssats är unik i den meningen att uttrycket kommer först, sedan lagras-operatorn (→), sedan variabeln. Detta gör det lätt att läsa från vänster till höger. I de flesta andra programspråk är ordningen den motsatta, t.ex. som S=A+B. Det är lite baklänges eftersom datorn evaluerar uttrycket till höger först och sedan lagrar resultatet i variabeln till vänster.*

Lärarkommentar: Det finns många fördelar med att lagra värden i variabler. Det gör det enklare att visa resultatet. Det tillåter också att variabeln kan användas i efterföljande beräkningar. Om du upptäcker att du använder samma tal på många olika ställen i ditt program så kan du lagra talet i en variabel och använda den överallt där den förekommer i ditt program.

Göra förbättringar i programmet

Du kan förbättra programmet genom att använda **Output(** snarare än **Disp** för att visa de ursprungliga inmatade värdena och de fyra resultaten tydligt ”märkta”. Försök nu själv!

Till höger finns på en skärm en del av koden för ett sådant program och på en annan skärm en programkörning som visar summan av A och B. Försök nu själv att skriva färdigt koden så att programkörningen tydligt visar summa, differens, produkt och kvot av två tal A och B.

Kom ihåg att inkludera en **Pause**-sats i slutet av **Output**-satserna för att förhindra att meddelandet ”Klar” förstör visningen på skärmen.

Kom ihåg att **TI-84 Plus** och **TI-84 Plus C/CE**-T har olika storlekar på START-skärmen (liksom även på GRAF-skärmen).

TI-84 Plus startskärm har 16 tecken per rad och 8 rader. TI-84 Plus C/CE-T har 26 tecken per rad och 10 rader.

Lärarkommentar: Uppmuntra eleverna att experimentera med andra matematiska formler och använda tilldelningssatser för att lagra resultat. Detta för att göra det enklare att visa (**Display**) eller Mata ut (**Output**) resultaten. I *tillämpningen* för detta kapitel får eleverna tre matematiska formler att arbeta med.