Kapitel 4: Använda biblioteksmodulen ti-plotlib

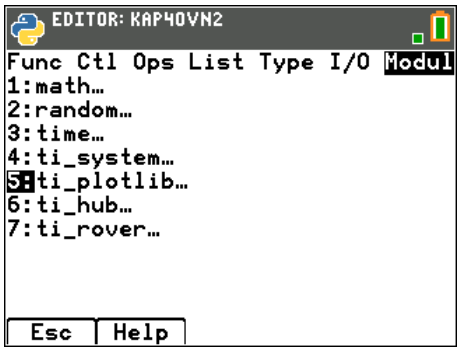
Övning 2: Grafiskt representera en funktion

I denna andra övning i kapitel 4 ska du lära dig att grafiskt representera en funktion med hjälp av biblioteksmodulen ti_plotlib.

Syfte:

- grafiskt representera en funktion
- repetera notationen för en avgränsad loop
- Göra inställningar för att visa grafik

Vi ska här skapa ett skript med hjälp av biblioteket **ti_plotlib,** som gör det möjligt att spåra i den grafiska representationen av en funktion *g* av *x* med värden i intervallet [a, b] och med N delsegment.

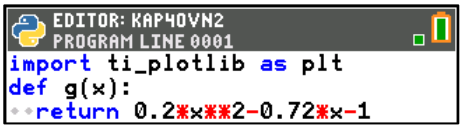


Skriptet du kommer att skapa är generellt så att du lätt kan skapa nya för andra exempel.

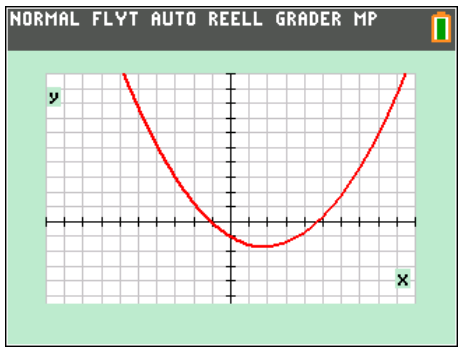
Lärarkommentar: Om du inte är bekant med loop-begreppet så föreslår vi att du går tillbaka till kapitel 1, 2 och 3 i TI-Code Python.

Starta ett nytt skript och döp det till KAP4OVN2

- Importera modulen ti_plotlib. Tryck på f1 (Fns...) och välj Modul och sedan 5: ti_plotlib.
- Definiera sedan funktionen $g(x) = 0{,}2x^2 - 0{,}72x - 1$.



Om du plottar denna funktion på räknaren utanför Python ser det ut som på skärmen här till höger. Vi ska se hur bilden blir efter Python-programmeringen. Vid denna plottning har vi valt att plotta kurvan lite tjockare. Man kan välja mellan tunna eller feta linjer och linjerna kan antingen vara hela eller prickade. Grafen här på bilden är plottad med tjocka heldragna linjer.

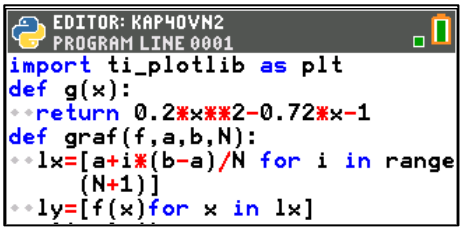


Man kan även ställa in färgerna på rutnätet, det finns fyra olika gråa nyanser.

Nu tillbaka till programmeringen!

Vi definierar först funktionen **graf** med argumenten f, a, b, och N.

x-listan **lx** konstrueras med hjälp av en avgränsad loop (slinga), vars instruktion du når genom att trycka på f1 (Fns…) och sedan välja **4: for i in range (size)** bland kontrollinstruktionerna. Listan **ly** beräknas från listan lx. Vi använder då instruktionen **7: for i in list**.



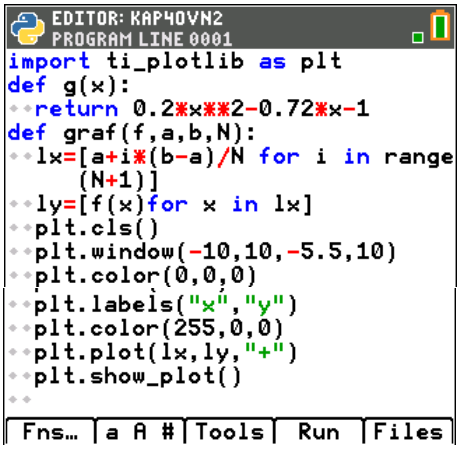
Lärarkommentar: Förklara instruktionen

lx=[a+i*(b-a)/N for i in range (N+1)]

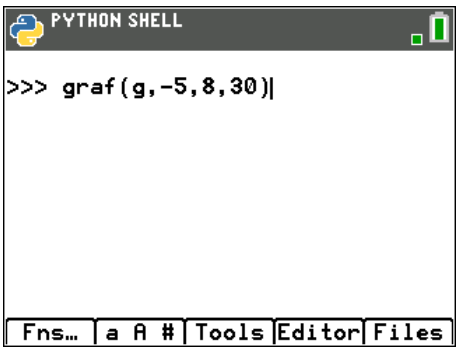
noggrant för eleverna. Visa och gör beräkningar med några värden på a, b och N.

Nu är du redo att utföra den grafiska representationen av funktionen. Du kan nu infoga instruktioner för att:

- Rensa skärmen: **plt.cls()**.
- Göra inställningar för grafikfönstret: **plt.window (xmin, xmax, ymin,ymax**)
- Visa axlar med axelmarkeringar: **plt.axes ("on").**.
- Namnge axlarna: **plt.labels ("x", "y").**
- Utföra beräkningar för den grafiska representationen: **plt.plot (lx,ly,"O").**
- Visa plottningen av funktionen: **plt.show_plot()**.

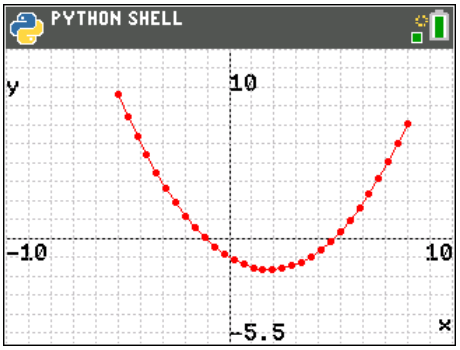


Alla satser ovan finns i **ti_plotlib**-modulen och i menyerna **Setup** och **Draw** (alternativ 5). Till höger har vi satt ihop två skärmar till en skärm.**Obs:** Anvisningarna för att välja färg i RGB (röd, grön, blå) sker genom kodning med varden 0 till 255 för varje färg. Man får tänka på att placera instruktionen för färgerna på ett klokt sätt så att man t.ex. undviker att ha koordinataxlarna i rött.

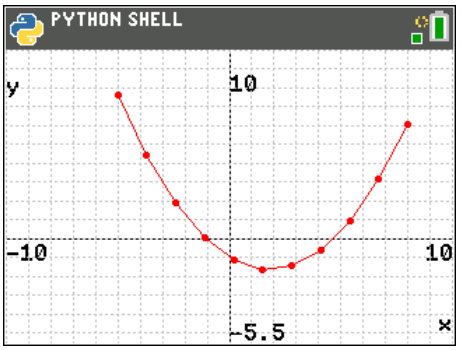


Gör nu en körning av skriptet genom att trycka på tangenten [vars] och sedan välja graf(). Sedan kan du t.ex. välja att visa grafen för funktionen g i intervallet [0; 3] med 30 delsegment.

Tryck nu på [enter]. Nu plottas en massa punkter med linjestycken emellan som bildar grafen för funktionen *g* mellan *x*-värdena -5 och 8.



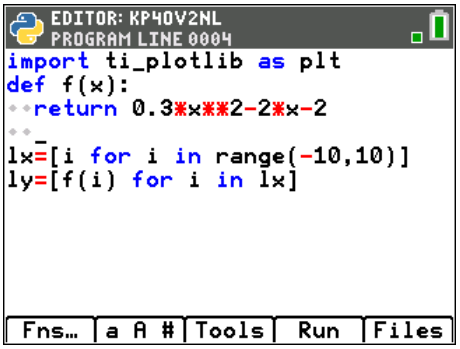
Pröva nu att ändra värdet på N till 10 t.ex. Du får då en mer kantig kurva men du ser linjestyckena mellan punkterna tydligt. Du ser nu att 11 punkter plottas.



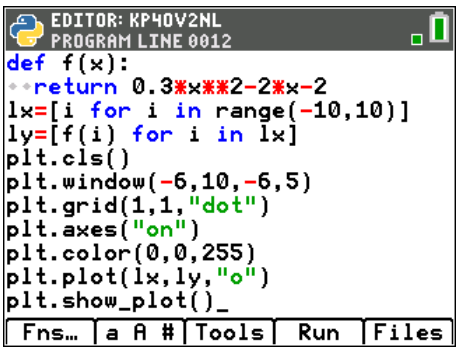
På nästa sida tar vi upp ett annat exempel som kanske är lite enklare.

Vi ska nu rita funktionen $0{,}3x^2 - 2x - 2$ i intervallet [-10; 10].

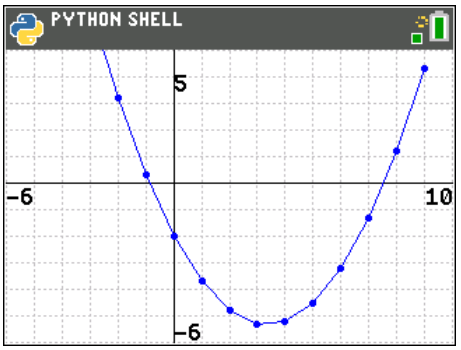
- Starta ett nytt Pythonprogram och döp det till KP4OV2NL och importera först biblioteksmodulen i ti_plotlib.
- Först definierar vi funktionen f.
- Sedan gör vi en lista över *x*-koordinater (lx).
- Och sedan en lista över motsvarande *y*-koordinater (ly).



Listorna för *x* och *y* görs med for-loopar från kontrollmenyn (Ctl), tryck på f1 (Fns…) och välj alternativen 4 och 7 för de två satserna. Se skärmbilden till höger:



Nu kan vi fortsätta med de olika instruktionerna för att plotta funktionen. När du exekverar skriptet plottas funktionen direkt. Du behöver alltså inte mata in några argument till funktionen.



För att göra plottningen ännu snyggare kan man skriva instruktionen för lx så här:

lx=[i/5 for i in range (-50,50)]

Då plottas även punkter som för *x* inte bara har heltalsvärden.

Pröva detta och ändra även punkterna vid plottningen till ”.” Då kan slutresultatet bli enligt den nedersta bilden i marginalen.