

Introduktion till MATLAB

1 Inledning

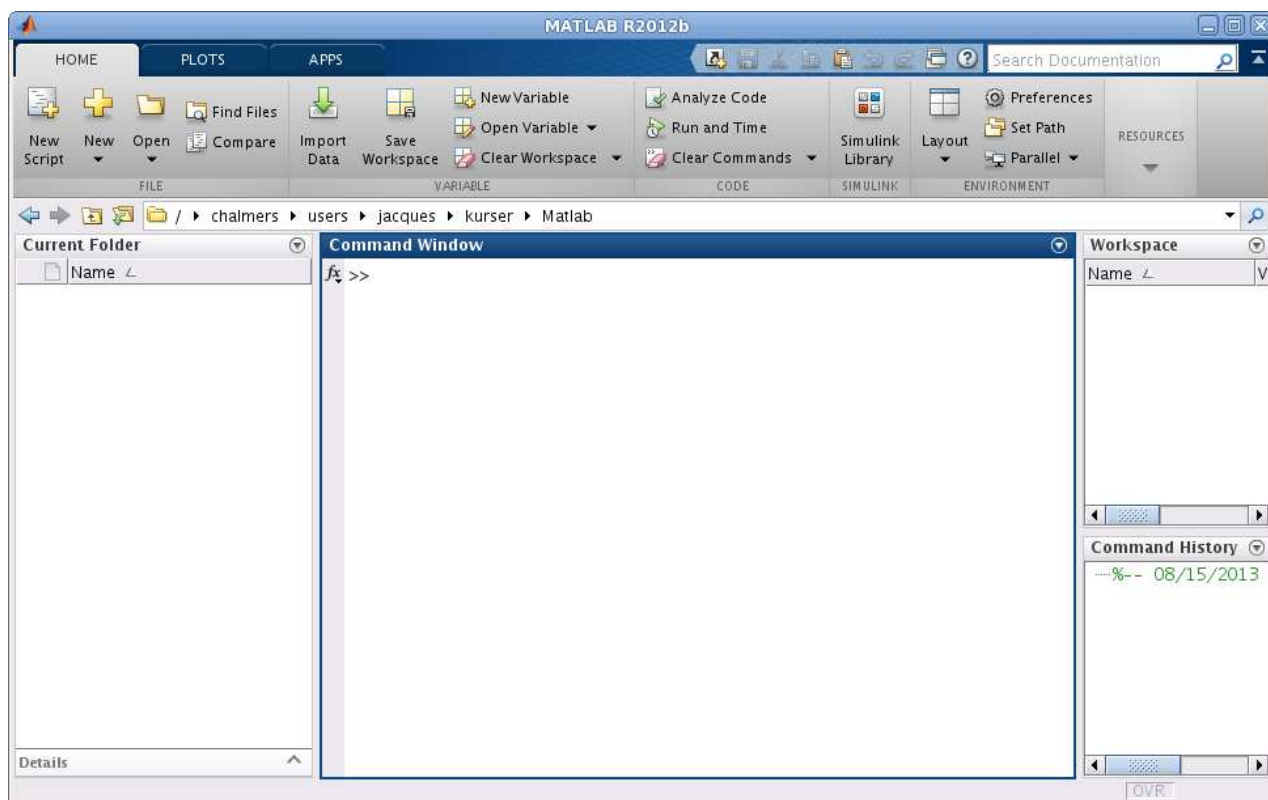
MATLAB är både en interaktiv matematikmiljö och ett programspråk, som används på många tekniska högskolor runt om i världen. Med tiden har MATLAB blivit ett viktigt ingenjörswerktyg och har stor användning även inom industrin.

En av styrkorna med MATLAB är att systemet är utbyggbart med bibliotek eller verktygslådor, toolboxes, för olika tillämpningsområden.

Studenterna på nästan alla ingenjörsprogram på Chalmers lär sig MATLAB och ni kommer använda MATLAB i många kurser i utbildningen. För det mesta introduceras MATLAB i matematikkurserna, men det är i teknikkurserna den mesta användningen sker. Det är viktigt att komma igång tidigt så att man hinner bli en tillräckligt erfaren användare.

2 Starta MATLAB

Vid en WINDOWS-dator startar man MATLAB genom att under WINDOWS-ikonen scrolla ned och välja MATLAB i listan av program. Sedan kommer MATLABs Desktop upp på skärmen.



Högst upp ser vi flikarna HOME, PLOTS och APPS. Det kommer bli fler när vi börjar arbeta, men mer om det senare.

De olika fönstren på Desktop har namn (namnet står överst i fönstret). Det stora fönstret i mitten kallas Command Window och där kommer vi ge kommandon, till höger ser vi Workspace och Command History där vi ser vilka variabler vi har respektive vilka kommandon vi givit tidigare, slutligen till vänster ser vi Current Folder som visar innehållet i aktuell mapp eller katalog.

3 En enkel beräkning och några grafer

Här följer några exempel så att vi snabbt kommer igång och ser lite resultat. Följ gärna med vid datorn och knappa in efter hand i Command Window och se vad som händer.

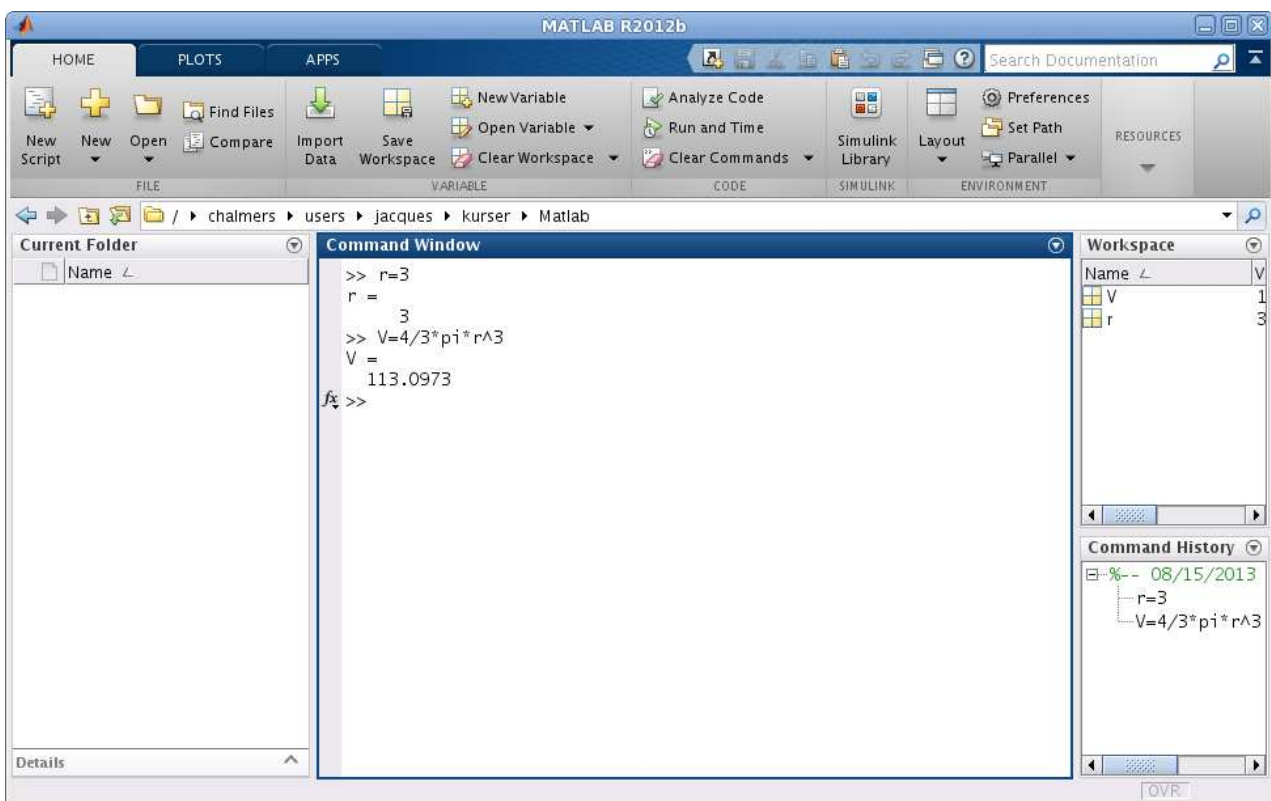
Exempel 1. Beräkna volymen av ett klot med radien $r = 3$ cm. Volymen ges av $V = \frac{4}{3}\pi r^3$.

Först inför vi en variabel `r`, för radien, som vi ger värdet 3.

```
>> r=3
```

Därefter beräknar vi volymen enligt formeln (`pi` ger en approximation av konstanten π) och låter variabeln `V` få detta värde.

```
>> V=4/3*pi*r^3
```



Ett variabelnamn skall börja med en bokstav (a-z, A-Z), därefter får vi ha bokstäver (a-z, A-Z), siffror (0-9) och understrykningstecken (`_`). MATLAB skiljer på stora och små bokstäver.

Den s.k. promptern (`>>`) skriver vi inte. Tecknet finns i Command Window på raden där vi skall skriva vårt kommando och visar att MATLAB är redo. Vi ser våra variabler och deras värden i Workspace och i Command History ser vi kommandona vi givit så långt.

Uppgift 1. Beräkna arean av en cirkelskiva med radien $r = 4$ cm. Arean ges av $A = \pi r^2$.

Exempel 2. Rita grafen av $f(x) = \sin(x) + 0.3 \sin(4x)$ för $0 \leq x \leq 4\pi$.

Först gör vi en lista eller radvektor $\mathbf{x}$ av x -värden mellan 0 och 4π , med kommandot

```
>> x=0:0.1:4*pi;
```

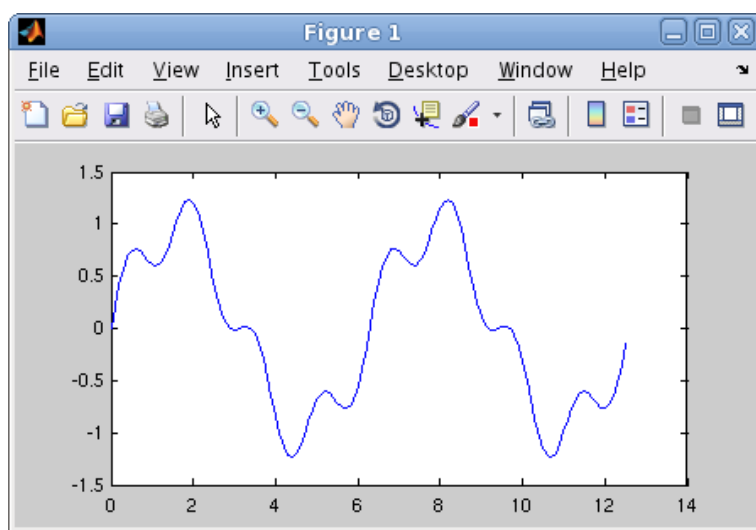
som vi skriver i Command Window.

Närmare bestämt får vi värdena 0, 0.1, 0.2, 0.3, $\dots$, 12.5, dvs. värden med start i 0, steget 0.1 och slut så nära upp mot 4π som möjligt. Om vi hade inte skrivit ett semikolon (;) sist i uttrycket för $\mathbf{x}$, hade alla x -värden skrivits ut i Command Window.

Därefter gör vi en lista eller radvektor $\mathbf{f}$ med $f(x)$ -värden för varje x -värde i $\mathbf{x}$ och ritar upp grafen med `plot`.

```
>> f=sin(x)+0.3*sin(4*x);  
>> plot(x,f)
```

Dessa kommandon skriver vi i Command Window och ett grafikfönster Figure kommer upp



Vi kan använda uppåtpil ($\uparrow$) för att komma till ett kommando vi givit tidigare, eller dra kommandot från Command History. Om vi vill kan vi gå längs raden med vänster- och högerpilarna ($\leftarrow$), ($\rightarrow$) och redigera kommandot. När kommandot ser ut som vi vill trycker vi på enter ($\leftrightarrow$).

Vi kan rensa Command Window med kommandot `clc` och med `clf` rensar vi Figure 1.

Uppgift 2. Rita grafen till $f(x) = \sin(x) + 0.3 \sin(5x)$ över intervallet $0 \leq x \leq 4\pi$.

Exempel 3. Rita graferna av $f(x) = \sin(x)$ och $g(x) = \sin(4x)$ för $0 \leq x \leq 2\pi$. Sätt rubrik och text på axlarna.

Vi använder funktionen `linspace` för att få 100 punkter jämnt fördelade mellan 0 och 2π , då blir graferna jämna och snygga.

```
>> x=linspace(0,2*pi);  
>> f=sin(x);  
>> g=sin(4*x);
```

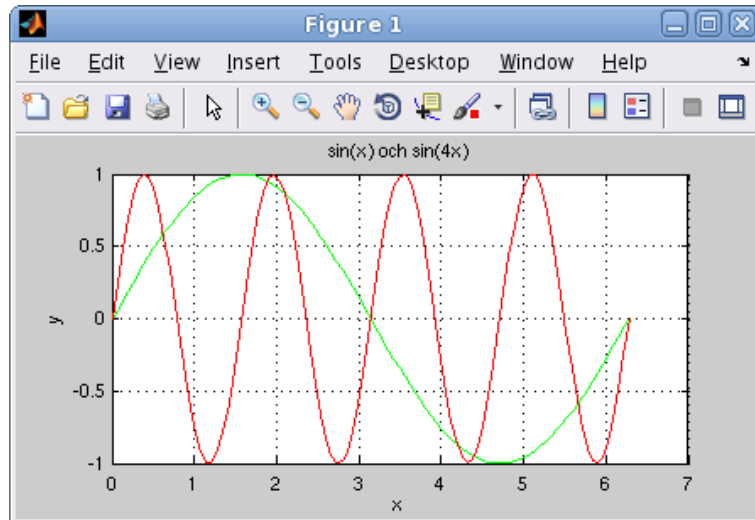
Vi ritar båda graferna samtidigt med `plot`, både paret x, f och paret x, g .

```
>> plot(x,f,'green',x,g,'red')
```

För att skilja graferna åt gjorde vi $\sin(x)$ -grafens gröna `'green'` och $\sin(4x)$ -grafens röda `'red'`.

Vi sätter text på axlarna och rubrik samt lägger på ett rutnät med

```
>> xlabel('x')
>> ylabel('y')
>> title('sin(x) och sin(4x)')
>> grid on
```

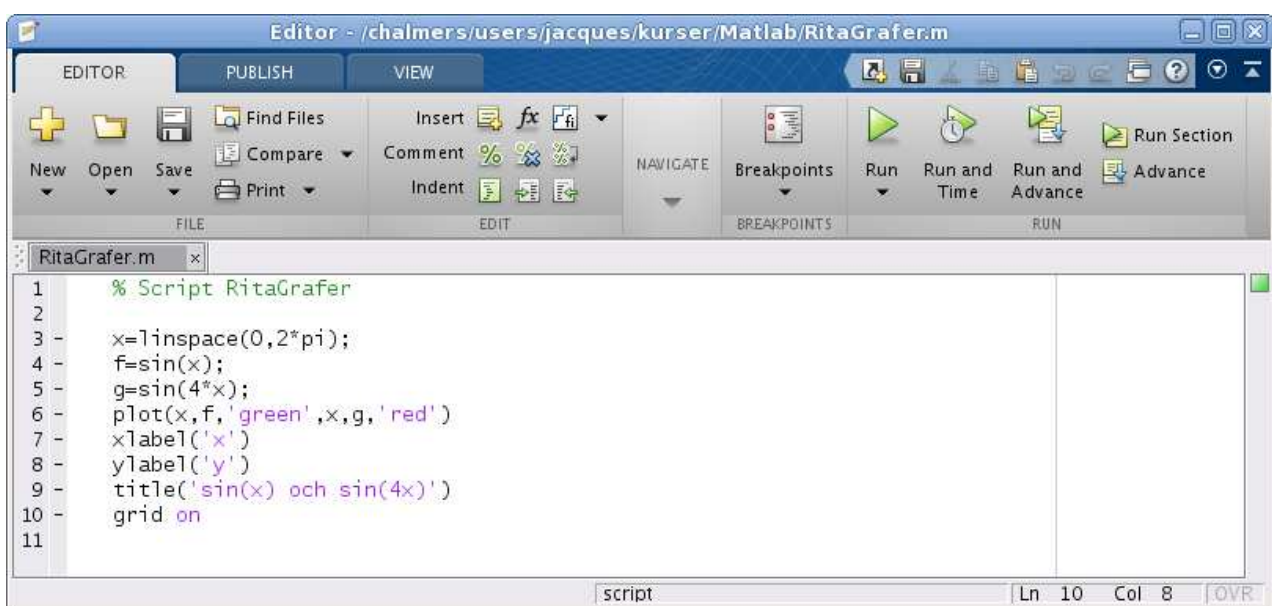


Texterna inom apostrofer (' '), t.ex. 'green' och 'x', är s.k. textsträngar.

4 Script

För att slippa skriva om sina kommandon, eller bläddra med uppåt- och nedåtpilar (↑), (↓) i kommandofönstrets historik eller dra från Command History, så brukar man skriva ett script. Ett script är en textfil som innehåller det man skulle kunna skriva direkt vid promptern (>>) i Command Window, och som utförs i MATLAB då man ger textfilens namn som kommando.

Som exempel ser vi på ett script för exempel 3 gjort med den i MATLAB inbyggda editorn.

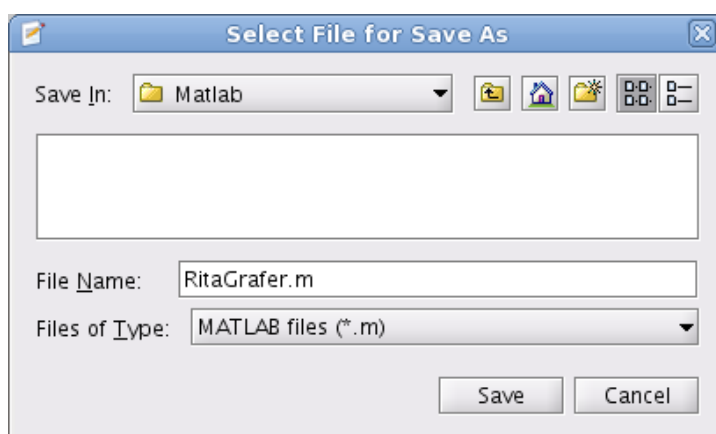


Editorn i MATLAB startas genom att man trycker på **New Script** eller det stor plustecknet på **HOME**-fliken (se bild av **Desktop** i avsnitt 2 eller 3).

Editorn markerar koden med olika färger för att visa vad som är kommentarer, nyckelord, textsträngar, etc. (Kommentarer inleds med procenttecken.)

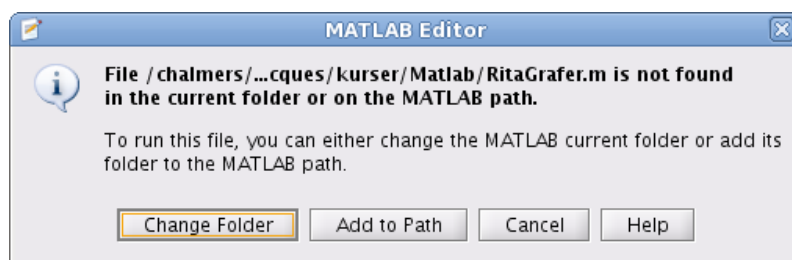
Spara kan vi göra under **Save** på **EDITOR**-fliken eller med diskett-symbolen i verktygsfältet och köra kan vi göra genom att trycka på  som finns på **EDITOR**-fliken. Då sparas vårt **script**, under ett namn vi väljer, och utförs som om vi gav namnet som ett kommando. När **scriptet** körs kommer MATLAB att utföra rad för rad (med start från första raden), och vi kommer få samma grafer som i exempel 3.

Så här ser dialogrutan ut som kommer upp då vi skall namnge vårt **script**.



Utanför MATLAB får namnet på ett **script** tillägget **.m** för att skilja denna typ av fil från andra filer.

För att MATLAB skall hitta filen, krävs det att katalogen där filen ligger är aktuell katalog. Om man försöker köra ett **script** som ligger i en annan katalog än den aktuella, så får man upp en fråga om att byta till den katalogen:



Välj **Change Folder** så byter MATLAB katalog.

Man kan byta katalog genom att antingen klicka sig fram i **Current Folder** eller använda navigeringsfältet precis under flikarna.

5 Lite programmering

I MATLAB finns repetitions- och villkorssatser som påminner om motsvarande i programspråk som C och Java. Vi kan göra egna funktioner i MATLAB. Om funktionen innehåller flera uttryck eller

satser gör vi en **function**, dvs. skapar en textfil med funktionsbeskrivningen. Består funktionen av ett enda uttryck så kan vi göra en s.k. anonym funktion (**anonymous function**).

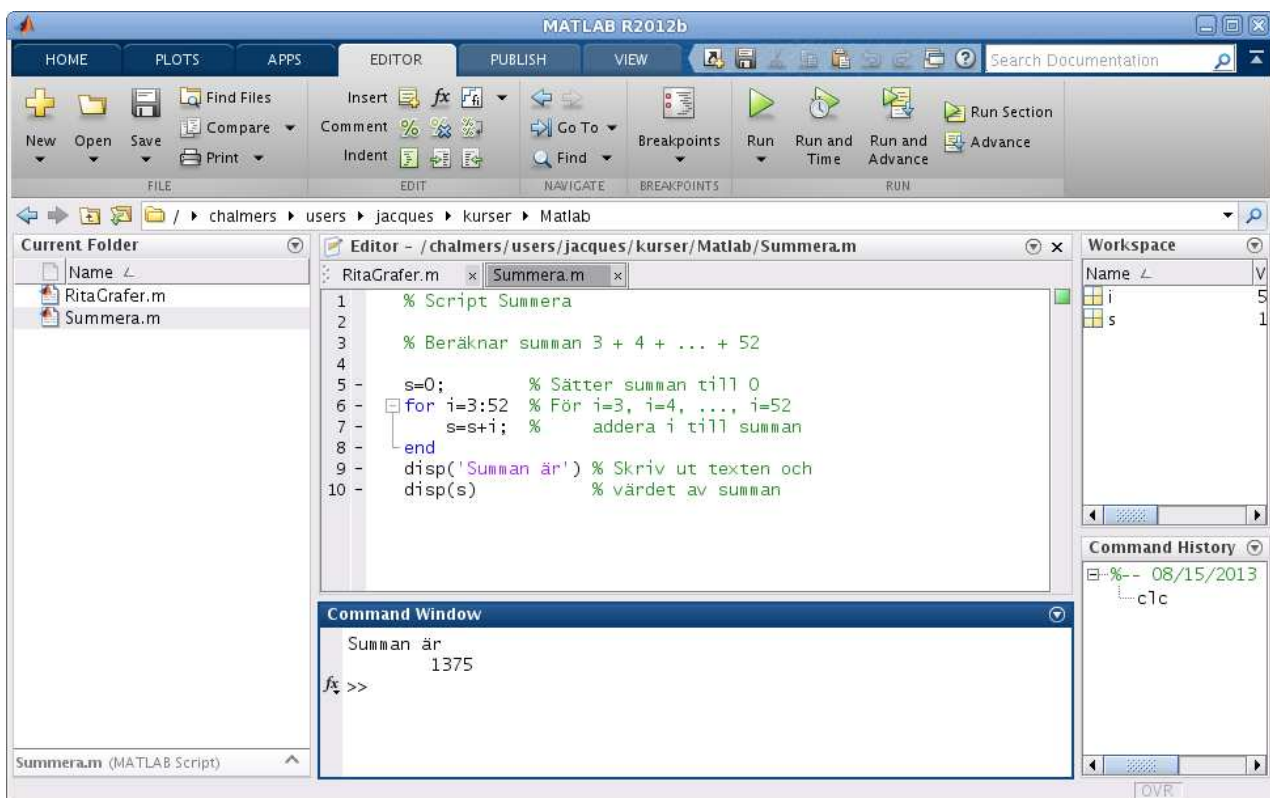
Vi nöjer oss för tillfället med att se på en repetitionssats, en **for**-sats, som vi använder för att beräkna en summa i följande exempel.

Exempel 4. Beräkna summan $s = 3 + 4 + 5 + \dots + 52$

Vi gör ett **script** med programkoden

```
s=0;
for i=3:52
    s=s+i;
end
```

Första satsen **s=0** är en tilldelningssats, variabeln **s** ges värdet 0. Andra satsen **for i=3:52** är en repetitionssats, den utför alla satser som följer ända ned till **end**, först för **i=3** sedan för **i=4** osv., ända tills satserna slutligen utförs för **i=52**. Första gången, dvs. då **i=3**, ges **s** ett nytt värde som är summan av gamla värdet på **s**, dvs. 0, och värdet på **i**, dvs. 3. Alltså kommer **s** ges värdet $0+3=3$. Andra gången, dvs. då **i=4**, ges **s** återigen ett nytt värde som är summan av gamla värdet på **s**, dvs. 3, och värdet på **i**, dvs. 4. Alltså kommer **s** ges värdet $3+4=7$. Så här fortsätter det ända tills **i=52** och **s** får sitt slutgiltiga värde.



Vi skriver lämpliga kommentarer (grön text) i programkoden och gör lämplig utskrift, först textsträngen **Summan är** och sedan summans värde.

Uppgift 3. Skriv ett **script** som beräknar summan $s = 1^2 + 2^2 + 3^2 + 4^2 + 5^2$. Använd en **for**-sats.