Matlab-föreläsning 3 (4), 17 september, 2015

Innehåll

- Sekvenser (från förra föreläsningen)
- Upprepning med `for`-slingor och `while`-slingor
- Villkorssatser med `if - then -else - end`
- Logik

Sekvenser - repetion från förra gången

Man kan även se en vektor som en sekvens av skalärer. En sekvens kan exempelvis användas för att göra funktionstabeller. För att skapa en variabel `a` som innehåller alla heltal från 0-10 kan vi skriva `a=[0:1:10]`. Ett annat användbart kommando är `linspace`. Gör `help linspace` för att ta reda på hur det fungerar.

Komponentvisa operationer, .-notation

Multiplikation (`*`), division (`/`) och upphöj till (`^`) av vektorer följer vanliga vektoralgebraiska regler i Matlab.

Ibland kan det hända att man vill utföra ovanstående operationer på varje element för sig i en vektor. Antag att vi för värdena $x = 1, 2, 3, 4$ och 5 vill beräkna x^2 dvs $1^2, 2^2, 3^2, 4^2$ och 5^2. Om vi i Matlab skapar en vektor `x=[1, 2, 3, 4, 5]` och skriver `x^2` så kommer vi få ett felmeddelande som säger att operationen inte är definierad. Varför?

För att kunna hantera komponentvisa operationer för multiplikation, divison och upphöjt till har man infört tre nya operationer i Matlab. `(.*)` för multiplikation, `(./)` för division, och `(.^)` för upphöjt till. Matlabs inbyggda funktioner som `sin` och `exp` osv opererar elementvis på vektorer (och matriser).

Uppgift 1 - provuppgift

Låt `x1=[1, 1, 3, 4]` , `x2=[3, -1, 2, 1]` och `x3=[-2, -1, 0, 1, 2 ]`. Vad blir resultatet av följande uttryck

A. `x1.*x1`

B. `x1./x2`

C. `x1.^2`

D. `x1.*x2`

E. `x1.^x2`

F. `x2.*x3`

Uppgift 2

Utöka programmet i **Uppgift 1 från Fö2** så att det beräknar luftmotståndet hos objektet för hastigheterna $v = 10, 20, 30, 40, 50, 100$ m/s. Programmet ska även plotta luftmotståndet som funktion av hastigheten.

UPPREPNING MED FOR-SLINGA

Ibland vill man utföra samma beräkning eller ett antal satser (kommandoblock) flera gånger. Detta kan göras på olika sätt beroende på vad som styr hur många gånger satserna ska upprepas.

Ett sätt är att använda en så kallad `for`-slinga. Denna använder man om man **på förhand vet hur många gången ett visst kommandoblock ska upprepas**. Generellt så skriver man en `for`-slinga på följande sätt

```
for ii=1:slut
  kommandoblock
end
```

I definitionen av `for`-slingan specificeras att variabeln `ii` ska börja på värdet 1 och anta alla heltal från 1 till `slut`. För varje värde på `ii` utförs kommandona som är specificerade i blocket. En `for`-slinga avslutas alltid med `end`.

Som exempel skriver följande rader ut värdet på en variabel `tal` på skärmen 5 gånger. Värdet på `tal` kommer att ges av variabeln `ii`. Vilka tal kommer vi att se på skärmen?

```
for ii=1:5
  tal=ii
end
```

UPPREPNING MED WHILE-SLINGA

Om man inte på förhand vet hur många gånger ett kommandoblock måste upprepas utan satserna **upprepas så länge ett villkor är uppfyllt** använder man en `while`-slinga. Generellt skriver man på följande sätt:

```
while avbrottskriterium
  kommandoblock
end
```

I exemplet nedan kommer variabeln `ii` plussas på med ett så länge `ii` är mindre än 3. Vilka tal kommer vi att se på skärmen?

```
ii=0;
while ii<3
  ii=ii+1
end
```

Nästlade slingor

Ibland kan man ha användning för en slinga inuti en annan. Det kallas för en **nästlad slinga**.

Uppgift 3

Skriv ett Matlab-program som beräknar positionen för två partiklar som startar i origo, $(x, y) = (0, 0)$, och som rör sig enligt följande mönster

Partikel 1

1. Går ett steg i x-led (negativ riktning) och ett steg i y-led (positiv riktning) 50 gånger.
2. Går ett steg i x-led (positiv riktning) och ett steg i y-led (positiv riktning) 25 gånger.
3. Går ett steg i x-led (positiv riktning) och ett steg i y-led (negativ riktning) 25 gånger.

Partikel 2

1. Går ett steg i x-led (positiv riktning) och ett steg i y-led (positiv riktning) 50 gånger.
2. Går ett steg i x-led (negativ riktning) och ett steg i y-led (positiv riktning) 25 gånger.
3. Går ett steg i x-led (negativ riktning) och ett steg i y-led (negativ riktning) 25 gånger.

Rita ut positionen för partiklarna i varje steg.

Gör nu om programmet så att det under de 50 första stegen stannar om partikel 2s x-position blir större än 30.

Uppgift 4 - provuppgift

Följande rader finns i ett program `Uppg4.m`. Vad blir utskriften på skärmen när vi kör programmet?

```
clear all
A=[1 2 3;4 5 6; 7 8 9];
for ii=1:3
    for jj=1:3
      disp(['Matrisens element = ' num2str(A(ii,jj))])
    end
end
```

Uppgift 5 - provuppgift

Följande rader finns i ett program `Uppg5.m`. Vilket värde har `x` när det skrivs ut på skärmen?

```
clear all
x=0;
ii=-5;
while ii<0
 x=x+ii;
 ii=ii+1;
end
disp(['x = ' num2str(x)])
```

Villkorssatser

Om man vill utföra olika beräkningar eller operationer beroende på olika variablers tillstånd kan man använda en `if`-sats. En `if`-sats **undersöker om ett villkor är uppfyllt**. Generellt, skriver man en `if`-sats på följande sätt

```
if (villkor = sant)
  kommandoblock
end
```

Ibland behöver man undersöka flera villkor och utföra kommandon beroende på vad som är uppfyllt och inte. Då använder man en antingen en `if/else` eller `if/elseif/else`-sats enligt nedan

```
if (villkor 1=sant)
  kommandoblock 1
else
  kommandoblock 2
end
```

```
if (villkor 1=sant)
  kommandoblock 1
elseif (villkor 2=sant)
  kommandoblock 2
else
  kommandoblock 3
end
```

Logik

För att kunna kontrollera om ett villkor är uppfyllt eller inte måste vi testa variabler mot varandra. Detta görs med tre logiska operationer, `NOT (~)`, `AND (&&)` och `OR (||)`. Ett villkor som är uppfyllt (=sant) kommer att evalueras till 1 och ett villkor som inte är uppfyllt (=falskt) evalueras till 0.

Uppgift 6 - provuppgift

Låt `a=5` och `b=-2`. Vad blir resultatet av följande kommandorader

A.
```
if a>7
   disp(['a = ' num2str(a)])
elseif a<7 && b<0
   disp(['b = ' num2str(b)])
end
```

B.
```
if a==5 && b>0
   disp(['a = ' num2str(a)])
elseif a==5 || b>0
   disp(['b = ' num2str(b)])
end
```

C.
```
if a~=5 && b<0
   disp(['a = ' num2str(a)])
elseif a==5 && b~=-2
   disp(['b = ' num2str(b)])
end
```

D.
```
if a~=5 && b<0
   disp(['a = ' num2str(a)])
elseif a==5 && b~=-2
   disp(['b = ' num2str(b)])
else
   disp(['Inget ar ju uppfyllt!!'])
end
```

PROGRAMMERINGSSTRATEGI

- Specificera problemet och vad programmet ska göra.
- Vad är in- och utdata?
- Hitta algoritmen (de steg som behöver utföras).
- Omvandla algoritmen till MATLAB-kod.
- Testa programmet.

Uppgift 7

Skriv ett MATLAB-program som omvandlar poängen på en algebratentamen till ett bokstavsbetyg. Betyget ges enligt följande poängfördelning

Betyg	A	B	C	D	E	Fx
Totalt antal poäng	27	24	21	18	16	15
Varav poäng från C	6	3				

Programmet ska ta totala poäng totalt samt poäng på C-delen som indata och skriva ut betyget på skärmen.