

Notat 1, del 6 (av totalt 9 deler)

Logiske funksjoner og betingelsestrukturer

Innhold

- Sammenlikningsoperatorer
- Logiske operatorer
- Forstå hvordan MATLAB tolker disse
- Bruk av **find**-funksjonen

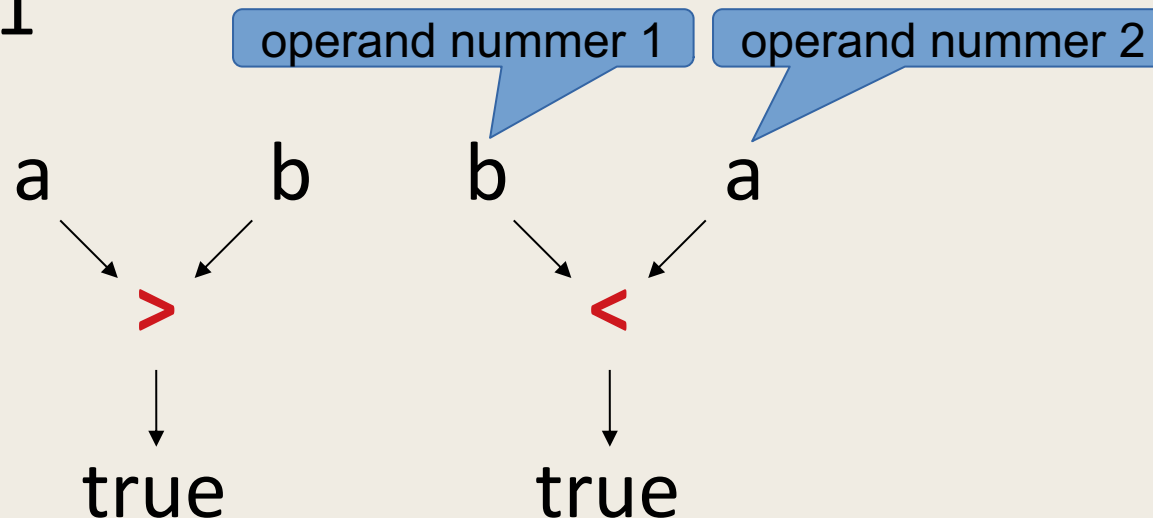
Sammenlikningsoperatoren

<	Mindre enn
<=	Mindre eller lik
>	Større enn
>=	Større eller lik
==	Er lik
!=	Er ikke lik

Binære operatører

- Binær operator betyr at det er **to operander** (argumenter, inngangsverdier)
- Rekkefølgen betyr noe. Vanligvis evaluert **fra venstre til høyre**

a=2 b=1



Sammenlikingen er enten true eller false

- MATLAB bruker 0 (også 0.000) for **usann** ("false") og alt annet (f.eks. 1) for **sann** ("true")
- MATLAB har egen datatype for logiske verdier

```
>> x=5; y=1;  
>> x<y  
  
ans =  
  
logical  
  
0
```

```
>> x>y  
  
ans =  
  
logical  
  
1
```

```
>> true  
  
ans =  
  
logical  
  
1  
  
>> false  
  
ans =  
  
logical  
  
0
```

```
>> 0 == false  
  
ans =  
  
logical  
  
1  
  
>> 1 == true  
  
ans =  
  
logical  
  
1
```

```
>> if 1 ← sann  
disp('MATLAB thinks this is true')  
end  
MATLAB thinks this is true  
>> if 321 ← sann  
disp('MATLAB thinks this is true')  
end  
MATLAB thinks this is true  
>> if -321 ← sann  
disp('MATLAB thinks this is true')  
end  
MATLAB thinks this is true  
>> if 0 ← usann  
disp('MATLAB thinks this is true')  
end
```

Resultatet av en sammenligning brukes i betingelse-strukturer og i repetisjonsstrukturer for å velge om kommandoer/setninger skal utføres eller ikke

Logiske operatorer

&	OG ("and")
 	ELLER ("or")
xor()	EKSKLUSIV ELLER ("exclusive or")
~	IKKE ("not") – kalles unær fordi kun 1 operand

x	y	x & y	x y	x xor y	~x
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

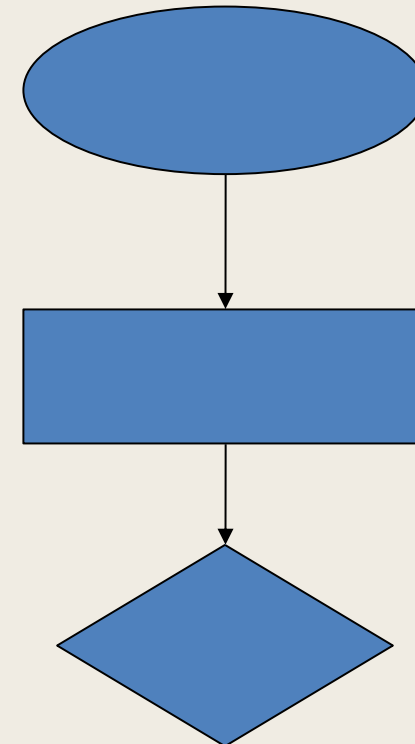
Flytskjema og pseudokode

Når programmene etterhvert blir mer kompliserte, er det viktig å PLANLEGGE hvordan koden skrives før du gjør det

Flytskjema — Grafisk planlegging

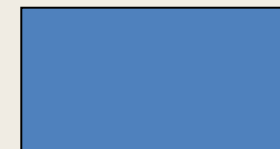
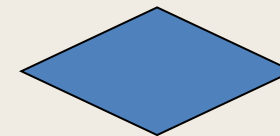
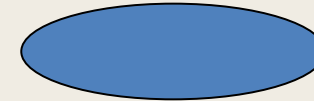
Pseudokode

Beskrivelse med en blanding av vanlige ord og nøkkelord. Denne beskrivelsen kan puttes inn i skriptet som kommentarer. Så skrives koden innimellom kommentarene!



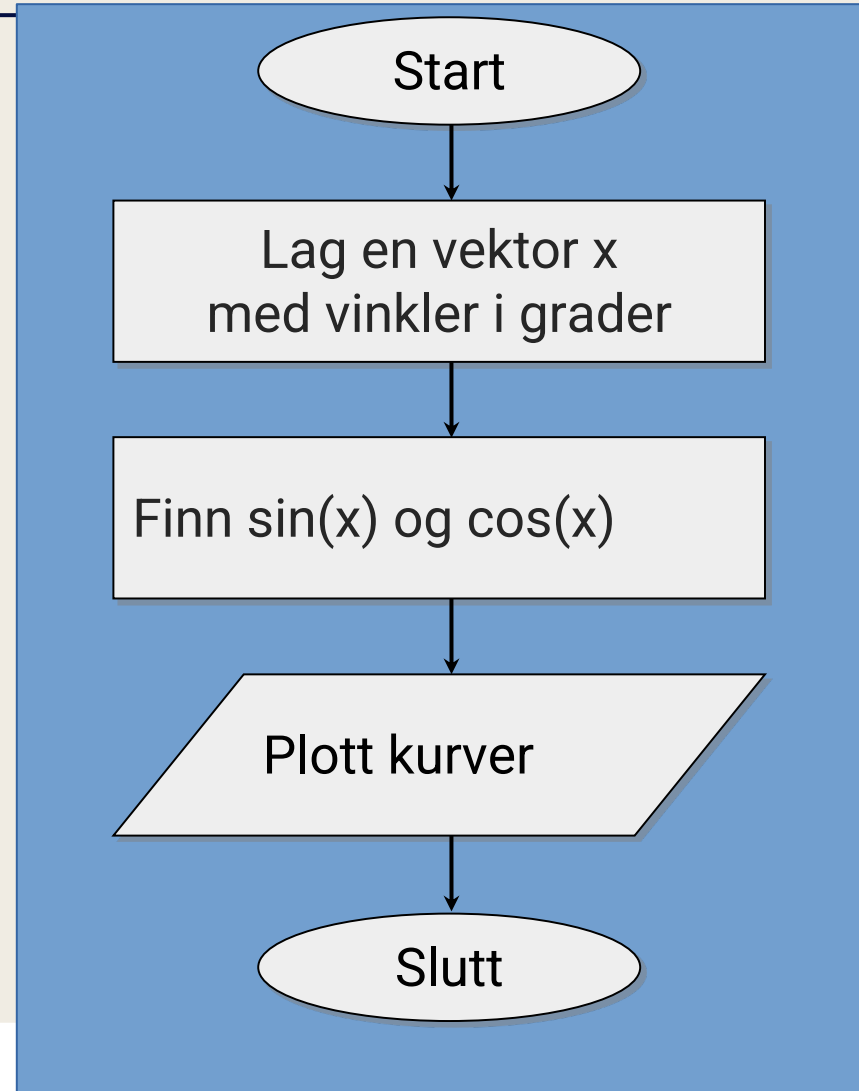
Symboler i flytskjema

- En oval indikerer start eller slutt
- Et parallelogram indikerer innmating eller utskrift
- En rombe (diamant) indikerer forgreining
- Utregninger og andre ting indikeres ved et rektangel



Eksempel på bruk av flytskjema

- Oppgave:
Lag et program som plotter
en sinus- og en cosinusurve
for -360 til 360 grader.



Logiske funksjoner

- MATLAB har tradisjonelle betingelsestrukturer som
 - if/end
 - if/elseif/else/end
 - switch/case
- ... men har også «logiske funksjoner» som ofte kan løse samme type oppgaver
- ... og de har som regel MYE bedre ytelse (raskere utføring)

Logiske funksjoner

find-funksjonen: Søker i en matrise/ vektor etter verdier som passer til et gitt søkekriterie

Hvis x er en vektor/matrise med tall:

- all-funksjonen: **all(x)** returnerer "true" (1) hvis alle verdier i x er ulik null. Ellers returneres "false" (0)
- any-funksjonen: **any(x)** returnerer "true" (1) hvis minst en av verdiene i x er ulik null. Ellers returneres "false" (0)

Eksempler på bruk av find

```
1 %Define a vector of heights
2 height = [63,67,65,72,69,78,75]
3 %Use the find command to determine which
4 %are greater than 66
5 accept = find(height>=66 )
6
7
```

Command Window

```
height =
    63    67    65    72    69    78
accept =
     2     4     5     6     7
>>
```

Merk at en slik bruk av find returnerer kun indeksverdier. For også å få ut de aktuelle høydene brukes 2 utargument:

```
>> [accept,heightvalue]=find(height>=66)
```

find i en matrise

find-funksjonen på en matrise kan returnere

- en enkelt indeksverdi
indeks = find(søkekriterium i matrise)
- både rad- og kolonnenummer **ved**
å definere en rad- og
kolonnevariabel:
[rad, kol] = find(søkekriterium i matrise)

For matriser vil 2 utargument gi rad og kolonne

```
>> g = [-1 0 2; 3 4 0];
```

```
>> [row,col]=find(g>=2)
```

Ved å bruke 3 utargument kan du få verdiene for de som er forskjellig fra 0 ved å bare skrive find(g) uten kriterium

```
>> g = [-1 0 2; 3 4 0];
```

```
>> [row,col,value]=find(g)
```

Ved å bruke 3 utargument sammen med kriterium får du ut logisk verdi i value-variabel for de som tilfredstiller kriteriet

```
>> g = [-1 0 2; 3 4 0];
```

```
>> [row,col,value]=find(g>2)
```

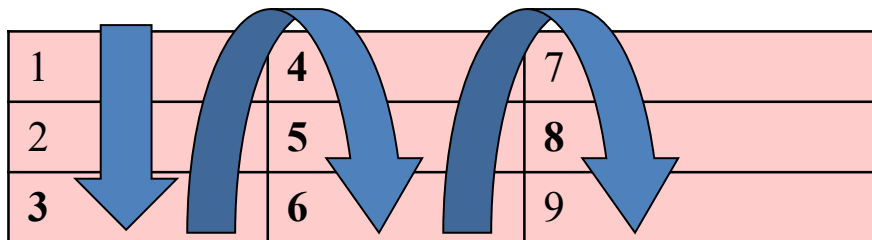
Eksempel: temperatur på pasienter (i Farenheit!)

Sengepost 1	Sengepost 2	Sengepost 3
95.3	100.2	98.6
97.4	99.2	98.9
100.1	99.3	97

Hvilke pasienter har for høy temperatur?

- Enkeltindeks

```
>> temp = [95.3, 100.2, 98.6; 97.4, 99.2, 98.9; 100.1, 99.3, 97]
temp =
    95.3000    100.2000    98.6000
    97.4000    99.2000    98.9000
    100.1000    99.3000    97.0000
>> element=find(temp>98.6)
element =
     3
     4
     5
     6
     8
>> |
```



Hvis vi vil ha rad og kolonne

```
temp =  
    95.3000    100.2000    98.6000  
    97.4000    99.2000    98.9000  
    100.1000    99.3000    97.0000
```

```
>> [row,col]=find(temp>98.6)
```

```
row =
```

```
    3
```

```
    1
```

```
    2
```

```
    3
```

```
    2
```

```
col =
```

```
    1
```

```
    2
```

```
    2
```

```
    2
```

```
    3
```

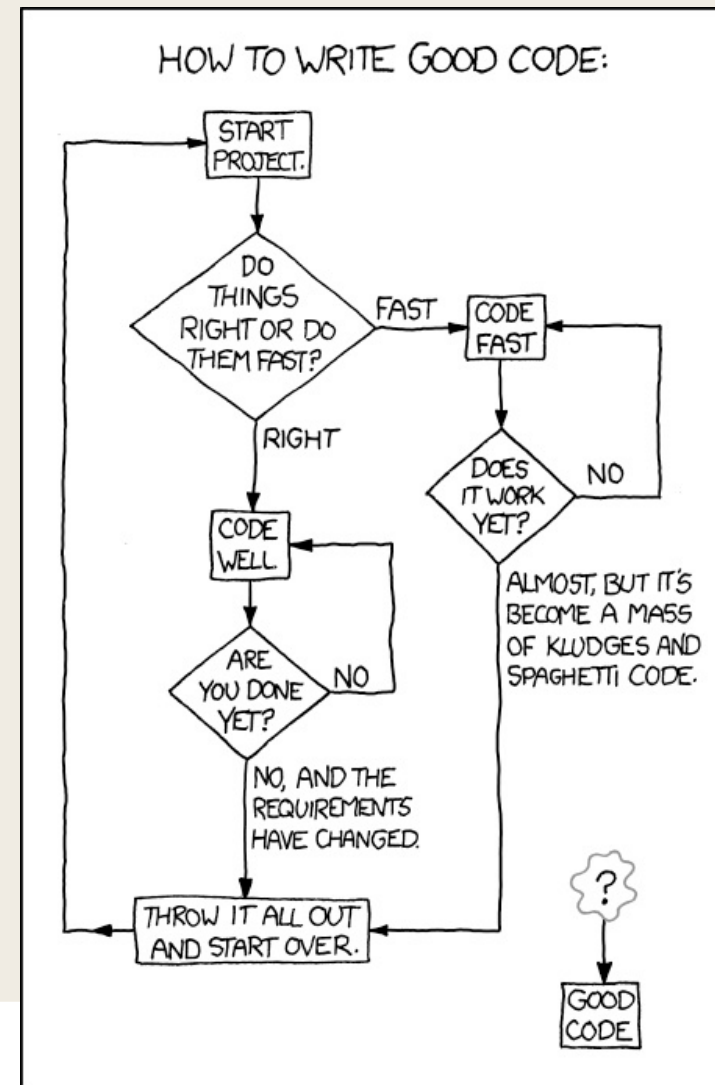
```
>> |
```

To utargumenter

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3

Betingelsesstruktur

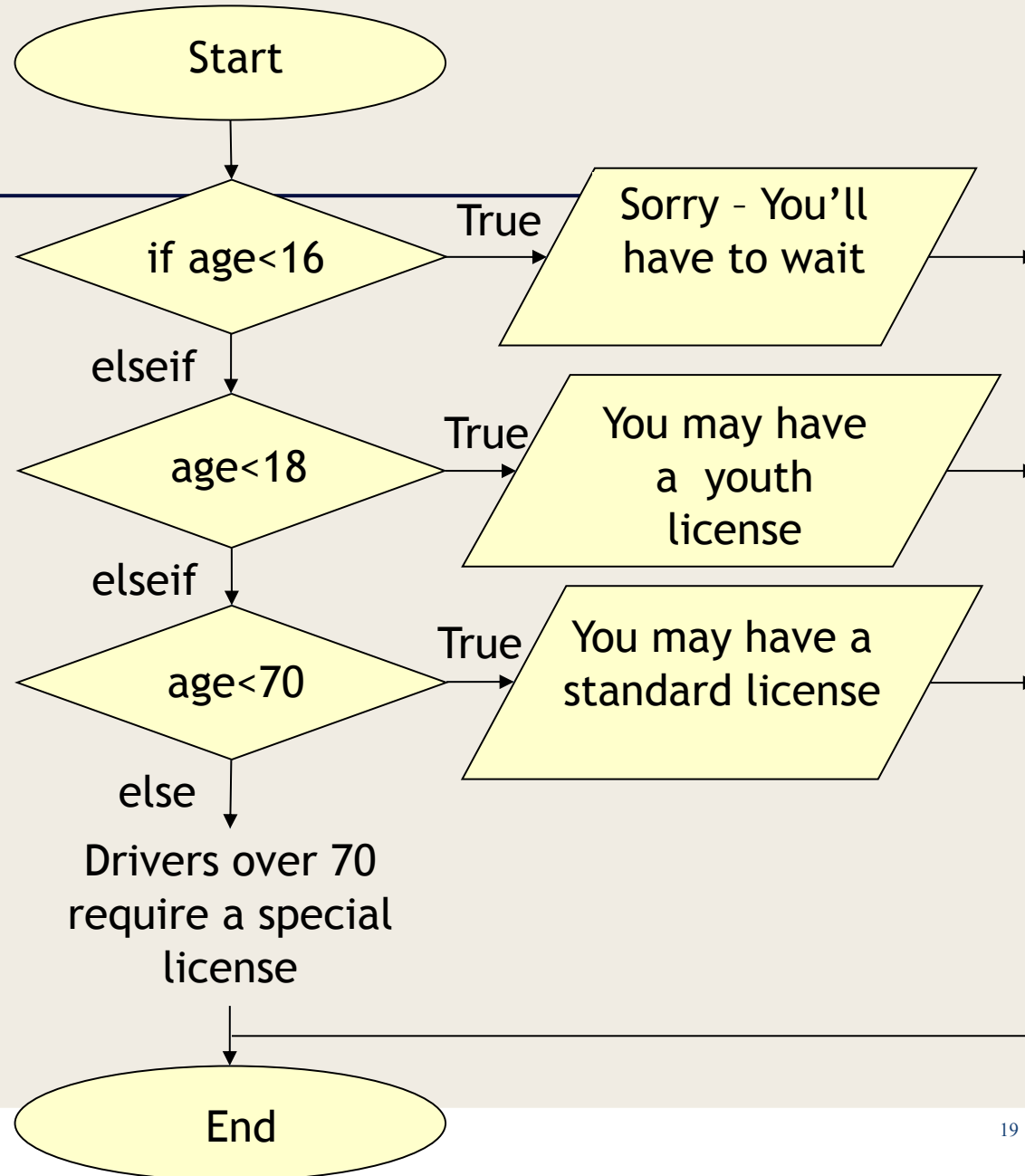
- if/end
- if/elseif/else/end
- switch and case
- **menu-funksjonen**



Eksempler på if – elseif – else – end

- Neste lysark viser flytskjema for et program som bestemmer om en person er kvalifisert til å kjøre bil eller ikke, ut fra alder.

Flytskjema



Program

```
1  disp('Are you eligible to drive?')
2  age=input('Enter your age:')
3  if age <16
4      disp('Sorry - You'll have to wait')
5  elseif age<18
6      disp('You may have a youth license')
7  elseif age<70
8      disp('You may have a standard license')
9  else
10     disp('Drivers over 70 require a special license')
11 end
12
```

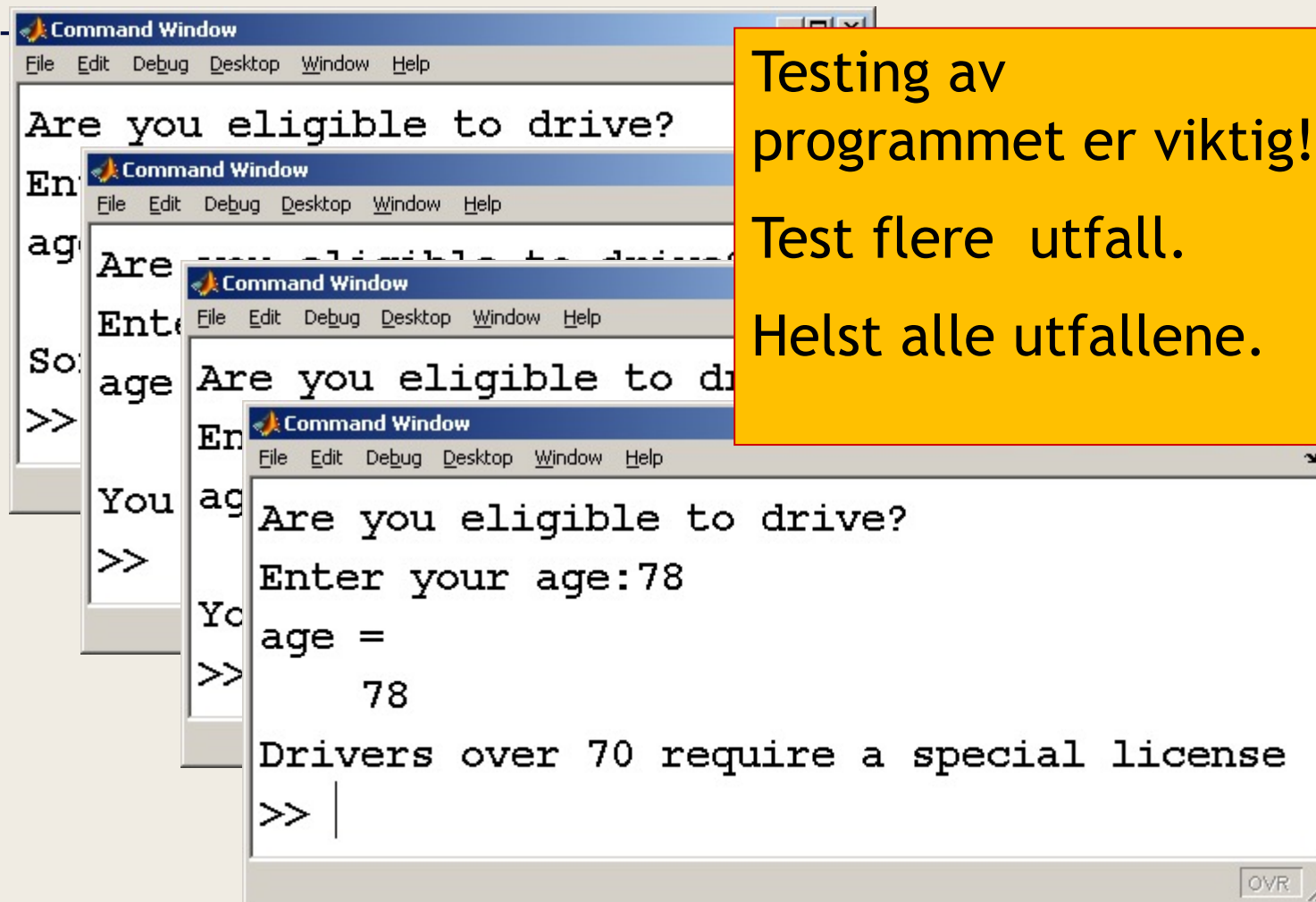
script

Ln 2

Col 27

OVR

Test av programvare



En annen flerbetingelsestruktur: switch-case som alternativ til if/elseif/else/end

```
switch variabel
case verdi1
    kode som utføres hvis variabel er lik verdi1
case verdi2
    kode som utføres hvis variabel er lik verdi2
...
case verdiN
    kode som utføres hvis variabel er lik verdiN
otherwise
    kode som utføres hvis variabel ikke er lik
    noen av de som ble nevnt
end
```

Eksempel: flypriser

```
1 city = input('Enter the name of a city in single quotes: ')
2 switch city
3     case 'Boston'
4         disp('$345')
5     case 'Denver'
6         disp('$150')
7     case 'Honolulu'
8         disp('Stay home and study')
9     otherwise
10         disp('Not on file')
11 end
12
```

script

Ln 5

Col 18

OVR

Command Window

File Edit Debug Desktop Window Help

Enter the name of a city in single quotes: 'Boston'

city =

Boston

\$345

Command Window

File Edit Debug Desktop Window Help

Enter the name of a city in single quotes: 'Denver'

city =

Denver

\$150

Command Window

File Edit Debug Desktop Window Help

Enter the name of a city in single quotes: 'Honolulu'

city =

Honolulu

Stay home and study

Command Window

File Edit Debug Desktop Window Help

Enter the name of a city in single quotes: 'New York'

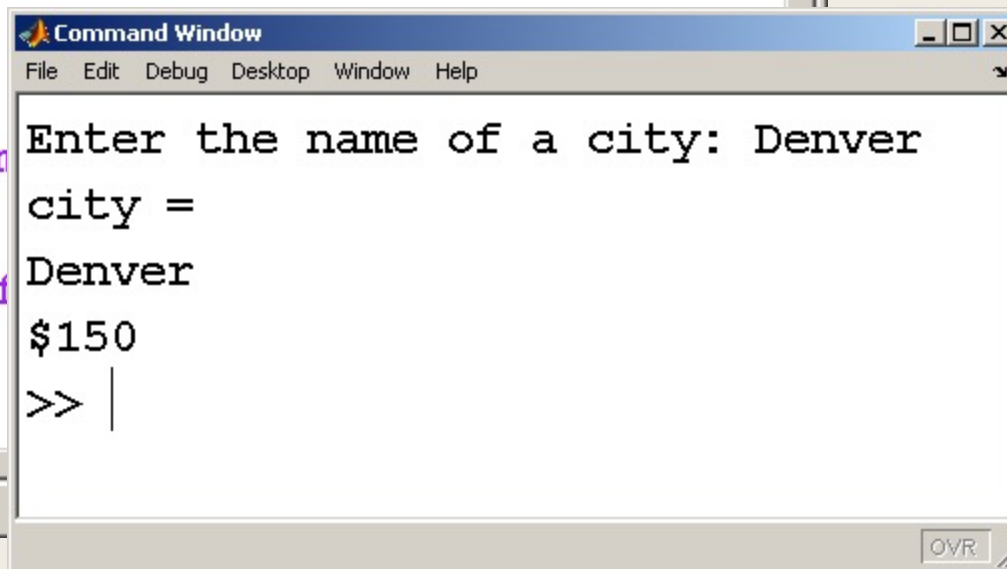
city =

New York

Not on file

En tekststreng som input ('s')

```
1 city = input('Enter the name of a city: ', 's')
2 switch city
3     case 'Boston'
4         disp('$345')
5     case 'Denver'
6         disp('$150')
7     case 'Honolulu'
8         disp('Stay home')
9     otherwise
10        disp('Not on the list')
11 end
12
```



Command Window

File Edit Debug Desktop Window Help

Enter the name of a city: Denver

city =

Denver

\$150

>> |

OVR

Meny: menu-funksjonen

```
1  city = menu('Select a city from the menu: ', 'Boston', 'Denver', 'Honolulu')
2  switch city
3      case 1
4          disp('$345')
5      case 2
6          disp('$150')
7      case 3
8          disp('Stay home and study')
9  end
10 |
11
12
13
```

script Ln 10 Col 1 OVR

Når koden kjøres



I stedet for å velge i kommandovindu, kommer valgene fram som knapper i et eget menyvindu.

Hvis jeg velger Honolulu

