NTNU

Fakultet for informasjonsteknologi,
matematikk og elektroteknikk

Institutt for datateknikk
og informasjonsvitenskap

**BOKMÅL**

# LØSNINGSFORSLAG

# AVSLUTTENDE EKSAMEN TDT 4105

## Informasjonsteknologi, grunnkurs

## Onsdag 16. desember 2009, 9.00–13.00

**Faglig kontakt under eksamen:**
Jørn Amundsen (918 97 897), Jo Skjermo (922 36 618) og Alf Inge Wang (922 89 577).

**Hjelpemidler (C):**
Tilleggshefte I, "Introduksjon til HTML, CSS, JSP og MYSQL" (alle utgaver)
Tilleggshefte II, "introduksjon til: MATLAB" (alle utgaver)
Typegodkjent elementær kalkulator, enten HP 30S eller Citizen SR-270X.
*Det er tillatt å stryke over med markeringspenn, men ikke å skrive i tilleggsheftene.*

**Sensurdato:**
18. januar 2010. Resultater gjøres kjent på http://studweb.ntnu.no.

**Kvalitetssikrer:**
Lars Ivar Igesund

Oppgavesettet inneholder 4 oppgaver. Det er angitt i prosent hvor mye hver oppgave og hver deloppgave teller ved sensur. Les igjennom hele oppgavesettet før du begynner å lage løsning. Disponer tiden godt! Gjør rimelige antagelser der du mener oppgaveteksten er ufullstendig, skriv kort hva du antar.

Svar kort og klart, og skriv tydelig. Er svaret uklart eller lenger enn nødvendig trekker dette ned.

## Oppgave 1 – Flervalgsoppgaver (20 %)

Bruk vedlagt svarskjema for å svare på denne oppgaven. Du kan få nytt ark av eksamensvaktene dersom du trenger dette. Kun ett svar er helt riktig. For hvert spørsmål gir korrekt avkryssing 1 poeng. Feil avkryssing eller mer enn ett kryss gir -1/2 poeng. Blankt svar gir 0 poeng. Du får ikke mindre enn 0 poeng totalt på denne oppgaven. Der det er spesielle uttrykk står den engelske oversettelsen i parentes.

1. Hva brukes en prosessor hovedsakelig til?
   a) Direkte kjøre kode skrevet i JSP, Matlab eller andre programmeringsspråk
   b) Utføre regneoperasjoner på og sammenlikninger av registre
   c) Utføre regneoperasjoner på og sammenlikninger direkte på data lagret i minnet (RAM)
2. Hva er hovedoppgaven til kontrollenheten i en prosessor?
   a) Holde orden på hvor langt prosessoren har kommet i JSP- eller Matlab-programmet
   b) Kontrollere at regneoperasjonene i ALU er korrekte
   c) Ha kontroll på hvor neste instruksjon ligger i minnet (RAM)
3. Hva er riktig rekkefølge av minnehierarki sortert på hastighet i en datamaskin ?
   a) registre, cache, primærminne, sekundærminne
   b) cache, registre, primærminne, sekundærminne
   c) primærminne, cache, registre, sekundærminne
4. Hva er tallet $1365_{10}$ representert som et binært tall?
   a) 11100101011
   b) 10110101010
   c) 10101010101
5. Hva er en database?
   a) En samling strukturerte data
   b) Informasjon + metainformasjon
   c) Et program for å håndtere store datamengder
6. Hva menes med fysisk datauavhengighet i databaser?
   a) Databasens konseptuelle skjema kan endres uten å endre på applikasjonen
   b) Måten data er fysisk representert i databasen kan endres uten å endre på applikasjonen
   c) Fysiske data lagret er uavhengig av dataene representert i databasen
7. Hvilke tre hovedoppgaver brukes et databaseverktøy (DBMS) til?
   a) Designe en database, konstruere en database, endre en database
   b) Konstruere en database, manipulere en database, søke i en database
   c) Definere en database, konstruere en database, manipulere en database
8. Hvordan finner man databasekrav i henhold til designprosessen for databaser beskrevet i læreboka?
   a) Bestemmer hvilken informasjon fra den virkelige verden som trengs i applikasjonen
   b) Bestemmer hvilket type databaseverktøy som trengs for å støtte en applikasjon
   c) Bestemmer hvilke entiteter som skal representeres i databasen

9. I hvilket tilfelle kan man **ikke** benytte seg av binærsøkalgoritmen?

    a) Når man har for stor datamengde (tar for lang tid)
    b) Når dataene ikke er sortert
    c) Når man ikke vet om det man søker etter finnes i dataene

10. Hva er et nøkkelattributt i databasesammenheng?

    a) Et attributt som styrer hvem som skal ha tilgang til bestemt informasjon i en database
    b) Et attributt som gjør det mulig å logge seg inn på en database med brukernavn og passord
    c) Et attributt som gjør det mulig å identifisere en instans av data unikt

11. TCP/IP er en samling av protokoller som bla. inneholder protokollene:

    a) HTTP, HTML, PHP, TCP, SMTP
    b) HTTP, FTP, TCP, IP, SMTP
    c) Kun TCP og IP

12. Hvilke lag inngår i TCP/IP protokollen?

    a) Applikasjon, transport, nettverk, link
    b) Applikasjon, transponder, nettverk, lenke
    c) Applikasjon, trådløst, nettverk, link

13. Hva gjør nettverkslaget i TCP/IP?

    a) Gir en ende-til-ende forbindelse
    b) Ruter pakker gjennom nettet fra maskin til maskin
    c) Tar seg av fysiske overføringer

14. Hvor mange bits benyttes i nåværende IPv4-adresser ?

    a) 32 bits
    b) 16 bits
    c) 48 bits

15. Hva er den viktigste oppgaven til en nettverksrouter?

    a) Koble sammen flere nettverkssegmenter av samme type
    b) Forsterke og rense signaler på nettverket
    c) Koble sammen ulike nettverk og nettverkssegmenter

16. Hvilken av følgende alternativer blir i følge loven **ikke** ansett som sensitive personopplysninger ved opprettelse av personregistre?

    a) Etnisk bakgrunn
    b) Politisk oppfatning
    c) Økonomiske opplysninger

17. Hva er “Service Level Agreement” (SLA) i forbindelse med tjenestekvalitet?

    a) En kontrakt som omfatter priser på ulike hastigheter på nettverk
    b) En gjensidig kontrakt med krav til kunde og leverandør relatert til kvalitet på nettverktjenester
    c) En kontrakt som formelt beskriver kvalitetskrav fra leverandør på nettverkstjenester

18. Hvor mange bits trengs for å kode det norske alfabetet inkludert mellomrom med enkel koding (hver bokstav/tegn er representert med like mange bit)?

    a) 4
    b) 5
    c) 6

19. En funksjon inneholder en for-løkke som går fra 0 til N-1 i trinn på 2, inne i en while-løkke som går fra 0 til N/2. Hvilken orden (hvor mye arbeid) har funksjonen ?

    a) $\mathcal{O}(N/2)$
    b) $\mathcal{O}(N\log N)$
    c) $\mathcal{O}(N^2)$

20. Hva blir Huffmann-koden til symbolene A-E når frekvensfordelingen i en melding på 100 symboler er som vist i tabell 1 ?

    a) A:1, B:01, C:001, D:0001, E:0000
    b) A:1, B:01, C:011, D:0011, E:0001
    c) A:1, B:01, C:001, D:0010, E:0001

| **Symbol** | **Antall** |
|---|---|
| A | 50 |
| B | 20 |
| C | 15 |
| D | 10 |
| E | 5 |

Tabell 1: Frekvensfordeling av symbolene A-E i en melding på 100 symboler

**Løsning:**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| b | c | a | c | a | b | c | a | b | c | b | a | b | a | c | c | b | b | c | a |

## Oppgave 2 – HTML (10 %)

Bruk informasjonen oppgitt under til å skissere hvordan filen `side.html` vil vises i en nettleser.

Anta nettleseren bruker standardvisning, vertikalt og horisontalt midtjustert med fet skrift for attributten `<th>` og venstrejustert for attributten `<td>`.

Innhold av filen `side.html`, plassert i folderen `websider`:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Eksamen 2009</title>
<link rel="stylesheet" type="text/css" href="stilark.css">
</head>
<body>
<h1>Overskrift</h1>
<table border="1">
   <tr>
      <th rowspan="2" colspan="3">A</th>
      <td>B</td>
   </tr>
   <tr>
      <td>C</td>
   </tr>
   <tr>
      <td>D</td>
      <th rowspan="2" colspan="2">E</th>
      <td>F</td>
   </tr>
   <tr>
      <td>G</td>
      <td>H</td>
   </tr>
   <tr>
      <td>I</td>
      <td>J</td>
      <td>K</td>
      <td>L</td>
   </tr>
</table>
</body>
</html>
```

*(oppgaven fortsetter på neste side)*

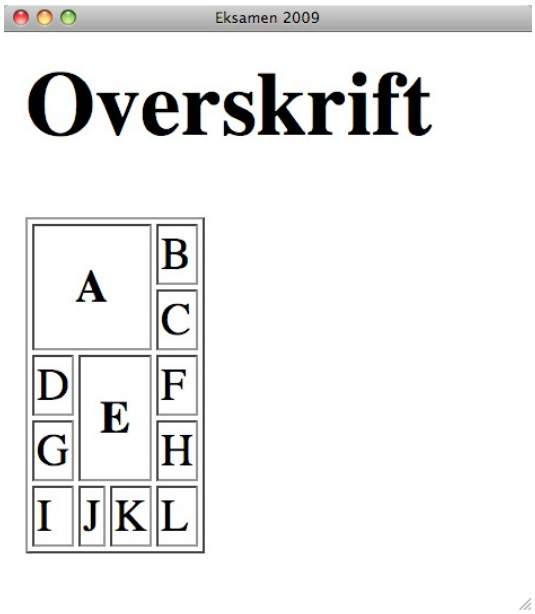
Innhold av filen `stilark.css`, plassert i folderen `cssfiler`:

```
td {text-decoration:underline;text-align:right;}
th {text-align:left;text-decoration:line-through;}
h1 {text-decoration:underline;text-align:right;}
```

### Oppgitte CSS tekstattributter

| | | |
|---|---|---|
| `text-align` | horisontal justering | |
| `text-decoration:line-through` | overstryking | ~~eksempel~~ |
| `text-decoration:underline` | understreking | eksempel |

**Løsning:**




Det blir ikke tatt hensyn til CSS-filen siden den ligger i en annen katalog.

## Oppgave 3 – Grunnleggende Matlab programmering (30 %)

Se vedlegget "List of built-in functions". I hver deloppgave er det angitt dersom det **ikke** er tillatt å bruke innebygde funksjoner, eller om navngitte innebygde funksjoner ikke er tillatt for å løse problemet. Hvis det ikke er tatt noen forbehold kan du bruke hvilken som helst av funksjonene i vedlegget.

**a)** (4%) Lag en funksjon `max2` som finner det største tallet i en liste med heltall. Bruk den innebygde funksjonen `intmax` for å løse problemet. Hvis det sendes inn en tom liste skal `max2` returnere `-intmax`. Problemet skal løses uten å bruke andre innebygde funksjoner enn `intmax` og `length`.

**Løsning:**

```
function mx = max2(L)
   mx = -intmax;
   n = length(L);

   for i = 1:n
      if (L(i) > mx)
         mx = L(i);
      end
   end
end
```

**b)** (5%) Lag en funksjon `mxcomp` som tar inn to like store heltallslister. Funksjonen skal returnere **to** parametere, en liste med det største elementet i hver posisjon (indeks), og en teller med totalt antall posisjoner hvor elementene er like store. Returverdien settes lik 0 (null) i posisjoner hvor elementene er like store.

**Eksempel:**

```
>> A = [1  5 -2  7  0 -1]
>> B = [3  5  1  4  0 -1]
>> [list num] = mxcomp(A,B)
list =

   3   0   1   7   0   0

num = 3
```

**Løsning:**

```
function [rlist rcount] = mxcomp(A,B)
   rcount = 0;
   n = length(A);

   for i = 1:n
      if (A(i) ~= B(i))
         rlist(i) = max(A(i),B(i));
      else
         rlist(i) = 0;
         rcount = rcount + 1;
      end
   end
end
```

**c)** (7%) Skriv en funksjon `trans` som tar inn en tabell `A` med $n$ rader og $n$ kolonner og returnerer den transponerte av tabellen. Transponert betyr "speilet" om hoveddiagonalen $A_{1,1} - A_{n,n}$, skrevet med fet skrift i figur 1.

| | 1 | 2 | 3 | 4 | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1 | **1** | 2 | 3 | 4 | | | **1** | 5 | 9 | 13 |
| 2 | 5 | **6** | 7 | 8 | $A^T$ | | 2 | **6** | 10 | 14 |
| 3 | 9 | 10 | **11** | 12 | $\longrightarrow$ | | 3 | 7 | **11** | 15 |
| 4 | 13 | 14 | 15 | **16** | | | 4 | 8 | 12 | **16** |

Figur 1: en transponert $4\times4$-tabell

Funksjonen skal returnere en tom tabell hvis inngangstabellen ikke er kvadratisk. Problemet skal løses **uten** å bruke transponert-operatoren. **Hint:** bytt om rader og kolonner.

**Løsning:**

```
function T = trans(A)
   [m n] = size(A);
   if (m ~= n)
      T = [];
      return;
   end

   for i = 1:n
      T(i,:) = A(:,i);
   end
end
```

**d)** (4%) Gitt funksjonen `secret`

```
function ret = secret(A,f)
    n = length(A);
    ret = 0;
    for i = 1:n
        ret = ret + f(A(i));
    end
end
```

På kommandolinjen skrives det inn

```
>> f = @(x) f = x^2 + 2*x;
>> M = [1  3  4];
>> R = secret(M,f)
```

Hvilke(t) tall skrives ut til skjermen ?

**Løsning:**

```
R = 42
```

**e)** (3%) Skriv om tilordningen `f` i forrige deloppgave og bruk den innebygde funksjonen `sum` (se vedlegg) til å regne ut det samme som kallet til `secret`. Det er ikke nødvendig å skrive svaret som en funksjon.

**Løsning:**

```
>> f = @(x) f = x.^2 + 2*x;
>> sum(f(M))
ans = 42
```

**f)** (3%) Skriv en funksjon `dice` som returnerer en liste med $N$ terningkast, hvert kast utført med en terning med $M$ øyne.

**Løsning:**

```
function throw = dice(n, m)
   throw = ceil(m*rand(1,n));
end
```

**g)** (4%) Skriv programsetninger som kaster N terninger med M øyne og sjekker om kastet resulterte i bare like verdier (tilsvarer yatzee for $N = 5$ og $M = 6$). Det er ikke nødvendig å skrive en funksjon.

**Løsning:**

```
>> L = sort(dice(n,m));
>> L(1) == L(n)
```

## Oppgave 4 – Avansert Matlab programmering (40 %)

Se vedlegget "List of built-in functions". I hver deloppgave er det angitt dersom det **ikke** er tillatt å bruke innebygde funksjoner, eller om navngitte innebygde funksjoner ikke er tillatt for å løse problemet. Hvis det ikke er tatt noen forbehold kan du bruke hvilken som helst av funksjonene i vedlegget.

**a)** (10%) En funksjon er definert ved formelen

$$L_n(x) = \begin{cases} 1 & \text{hvis } n \leq 0 \\ 1 - x & \text{hvis } n = 1 \\ (2n - 1 - x)L_{n-1}(x) - (n-1)^2 L_{n-2}(x) & \text{hvis } n > 1 \end{cases} \tag{1}$$

Skriv en funksjon `laguerre` som tar inn to parametere `x` og `n` og regner ut $L_n(x)$ med formelen (1).

**Løsning:**

```
function y = laguerre(x,n)
   if (n <= 0)
      y = 1;
   elseif (n == 1)
      y = 1 - x;
   else
      y = (2*n-1-x)*laguerre(x,n-1) - (n-1)^2*laguerre(x,n-2);
   end
end
```

**b)** (10%) Elementene i en tabell med 254 rader og 254 kolonner er lagret binært på filen `tab254.dat` med periodisk utvidelse som vist i figur 2. Filposisjonen til elementene i første rad, $A_{1,1} - A_{1,254}$, er gitt ved sammenhengen

$$\text{filposisjon}(A_{1,j}) = 8(1 + 256j), \tag{2}$$

for alle kolonner $j = 1, 2, \ldots, 254$. Bruk (2) til å skrive et skript som leser dataene fra `tab254.dat` inn i $254 \times 254$-tabellen $A$. Lag skriptet slik at ingen overflødige elementer leses fra fil, kun den indre rammen.

**Løsning:**

```
[fid message] = fopen('tab254.dat', 'r');
for j = 1:254
   status = fseek(fid, 8*(1+256*j), 'bof');
   A(:,j) = fread(fid, [254 1], 'double');
end
status = fclose(fid);
```

| | | | | | |
|---|---|---|---|---|---|
| $A_{254,254}$ 0 | $A_{254,1}$ 2048 | $A_{254,2}$ 4096 | • • • | $A_{254,254}$ 520192 | $A_{254,1}$ 522240 |
| $A_{1,254}$ 8 | $A_{1,1}$ 2056 | $A_{1,2}$ 4104 | • • • | $A_{1,254}$ 520200 | $A_{1,1}$ 522248 |
| $A_{2,254}$ 16 | $A_{2,1}$ 2064 | $A_{2,2}$ 4112 | • • • | $A_{2,254}$ 520208 | $A_{2,1}$ 522256 |
| ⋮ | ⋮ | ⋮ | ⋱ | ⋮ | ⋮ |
| $A_{254,254}$ 2032 | $A_{254,1}$ 4080 | $A_{254,2}$ 6128 | • • • | $A_{254,254}$ 522224 | $A_{254,1}$ 524272 |
| $A_{1,254}$ 2040 | $A_{1,1}$ 4088 | $A_{1,2}$ 6136 | • • • | $A_{1,254}$ 522232 | $A_{1,1}$ 524280 |

Figur 2: Tabelldata lagret binært på disk med 8 bytes/element, i en fil på 524288 bytes totalt. Hver rute viser verdi og filposisjon til begynnelsen av elementet.

**c)** (5%) Tverrsummen av et tall er summen av sifferverdiene, f.eks. er $\text{tverrsum}(16) = 1 + 6 = 7$. Skriv funksjonen `d2sum` som finner tverrsummen av ett tall $0 \leq n \leq 99$. Funksjonen skal returnere 0 hvis det gies inn et tall utenfor gyldig område.

**Løsning:**

```
function sum2 = d2sum(n)
   if (n < 0 || n > 99 )
      sum2 = 0;
      return
   end

   sum2 = mod(n,10) + floor(n/10);
end
```

**d)** (15%) Kredittkortnumre (f.eks. VISA eller Mastercard) er satt sammen slik at det er mulig å sjekke om et feilaktig nummer blir tastet inn (alle enkle og 7 av 10 doble feilinnslag kan detekteres). Nummeret sjekkes med *Luhns algoritme* :

> **Sjekk** at nummeret har minimum 13 og maksimum 19 sifre
> **Utfør** for nest siste siffer og annethvert siffer tilbake inntil første (mest signifikante) siffer:
> doble tallverdien av sifferet
> **Summer** *tverrsummen* av alle tallverdiene
> **Sjekk** om summen går opp i 10

**Eksempel:**

For det fiktive kredittkortnummeret $3\,9\,0\,6\,5\,7\,1\,1\,8\,2\,3\,2\,4\,2\,6\,8$ blir Luhns sum

$$\begin{aligned} L &= \text{sum\_av\_tverrsummer}(2{\cdot}3\ 9\ 2{\cdot}0\ 6\ 2{\cdot}5\ 7\ 2{\cdot}1\ 1\ 2{\cdot}8\ 2\ 2{\cdot}3\ 2\ 2{\cdot}4\ 2\ 2{\cdot}6\ 8) \\ &= 6+9+0+6+1+7+2+1+7+2+6+2+8+2+3+8 \\ &= 70, \end{aligned}$$

så dette er gyldig som kredittkortnummer.

Skriv funksjonen `luhn` som tar inn et kredittkortnummer som en tekststreng og returnerer **sann** hvis nummeret er et gyldig kredittkortnummer, ellers **usann**. Bruk `d2sum` fra forrige deloppgave til å summere tverrsummene.

**Løsning:**

```
function validates = luhn(string)
  len = length(string);
  if (len < 13 || len > 19)
    validates = false;
    return;
  end

  twice = false;
  lsum = 0;
  for i = len:-1:1
    n = str2double(string(i));  % or str2num(i) or string(i) - '0'
    if (twice)
      n = 2*n;
    end
    twice = ~twice;

    lsum = lsum + d2sum(n);
  end

  validates = mod(lsum,10) == 0;
end
```

# List of built-in functions

**abs(x)**
absolute value of x, $|x|$

**acos(x)**
inverse cosinus of x, $\cos^{-1}(x)$

**acosd(x)**
inverse cosinus of x with angle in degrees

**acot(x)**
inverse cotangent of x, $\cot^{-1}(x)$

**asin(x)**
inverse sinus of x, $\sin^{-1}(x)$

**asind(x)**
inverse sinus of x with angle in degrees

**atan(x)**
inverse tangent of x, $\tan^{-1}(x)$

**atand(x)**
inverse tangent of x with angle in degrees

**atan2(x,y)**
element-wise inverse tangent of x/y

**bar(x,y), bar(y)**
produce bar graph of two vectors x and y
use indices of y as x-axis if one argument

**cd *dir***
change current working directory

**ceil(x)**
return smallest integer $\geq$ x

**char(x)**
convert integer(s) into character(s)

**clear var ...**
delete variable(s) from the symbol table

**cos(x)**
cosinus of x with argument in radians

**cosd(x)**
cosinus of x with argument in degrees

**cot(x)**
cotangent of x with argument in radians

**cotd(x)**
cotangent of x with argument in degrees

**cputime()**
return total CPU time spent so far

**disp(x)**
print the value of x with a newline

**eps, eps(N,M)**
machine precision as a number or a N×M-matrix

**exp(x)**
compute the exponential of x, $e^x$

**eye(N), eye(N,M)**
return $N \times N$ or $N \times M$ identity matrix

**false**
return logical 0 (false)

**status = fclose(fid)**
close file with file-id *fid*
return 0 on success, -1 on failure

**fix(x)**
round x to nearest integer towards zero

**floor(x)**
return largest integer $\leq$ x

**fid *or* [fid msg] = fopen(name,mode)**
open file with pathname *name*, mode character is 'r':read, 'w':write, 'a':append, 'r+':read and write
return file-id *fid* $> 0$ on success, -1 on error
optionally return error message *msg*

**fprintf(fid,format,variable,...)**
print variable(s) with specified formatting
%mf, %m.nf or %f:fixed-form float-point number
%me, %m.ne or %e:float-point with exponent
%md or %d:integer, %s:string, %%:%, \n:newline
m is field width, n is number of digits in fraction

**val *or* [val count] = fread(fid,sz,'double')**
read *sz* elements from file *fid* to *val*
optionally return number of elements read in *count*
*sz* is any of inf:as much as possible, N:n elements, [N M]:N×M matrix, [N inf]:M as large as possible

**status = fseek(fid,offset,origin)**
set file position to *offset* within open file *fid*
origin is 'bof', 'cof' or 'eof'
return 0 on success, -1 on failure

**position = ftell(fid)**
return file-pointer position of *fid*, -1 on failure

**count = fwrite(fid,var,'double')**
write variable *var* to file *fid*
return number of elements written in *count*

**grid on *or* off**
turns grid on or off on a 2D-plot

**hold off** *or* **on**
do or do not erase previos plot before plotting next

**val = input(msg)** ***or*** **input(msg, 's')**
output *msg*, then read keyboard input to *val*
last form reads input as a string (does not evaluate)

**intmax**
return largest (32-bit) integer available

**intmin**
return smallest (32-bit) integer available

**$A_{inv}$ = inv(A)**
return inverse of matrix A, the matrix $A^{-1}$

**isfinite(x)**
return 1 if x is a finite number, 0 otherwise

**length(A)**
return largest dimension of matrix A

**val = load('-ascii', name)**
load contents of text file *name* into *val*

**log(x)**
compute the natural logarithm, $\ln x$

**log2(x)**
compute the base-2 logarithm, $\log_2 x$

**var** *or* **[var ix] = max(x)**
find largest element in x, optionally with index *ix*

**mesh(X,Y,Z)**
plot 3-D mesh grid Z = f(X,Y)
use meshgrid to compute arrays X and Y

**[X Y] = meshgrid(x,y)** *or* **meshgrid(x)**
transforms domain specified by vectors (x,y) into arrays X and Y for use with 3-D plots
meshgrid(x) equals meshgrid(x,x)

**var** *or* **[var ix] = min(x)**
find smallest element in x, optionally with index *ix*

**mod(x,y)**
remainder of x/y, $x - \lfloor x/y \rfloor \cdot y$

**nargin**
number of arguments passed to the function

**nargout**
number of values the caller expects to receive

**norm(x)**
compute the 2-norm of x, `sqrt(sum(x.^2))`

**str = num2str(x)** ***or*** **num2str(x,n)**
convert input into text and store in *str*,
last form use a maximum precision of n digits

**ones(N), ones(N,M)**
return N×N or N×M matrix of ones

**pause(secs), pause**
pause execution *secs* seconds or until any key hit

**pi, pi(N,M)**
$\pi$ as a number or a N×M-matrix

**plot(x,y)**
2-D plot of vector x versus vector y

**prod(x)**
product of elements in x, $\prod x_i$

**pwd, string = pwd**
print or return working directory as a string

**rand, rand(N), rand(N,M)**
return a random number on the open interval (0,1),
a N×N or N×M matrix of random numbers

**realmax**
return largest real (floating-point) number

**realmin**
return smallest real (floating-point) number

**round(x)**
round x towards nearest integer

**save('-ascii', name, 'var', ...)**
save variables *var, ...* on text file *name*

**sign(x)**
sign of x, -1 if negative, 0 if zero and 1 if positive

**sin(x)**
sine of x with argument in radians

**sind(x)**
sine of x with argument in degrees

**size(A), size(A,n)**
return all dimensions or $n^{th}$ dimension of A

**sort(X), sort(X,n), sort(X,n,mode)**
sort X in ascending order, sort along $n^{th}$ dimension
or sort with mode `'ascend'` or `'descend'`

**sqrt(x)**
compute square root of x, $\sqrt{x}$

**x = str2double(string)**
convert character string to (floating-point) number

**A = str2num(string)**
convert character string matrix to number
*use str2double to convert a single number*

**y = sum(x)**
compute sum of elements, $\sum x_i$

**surf(X,Y,Z)**
plot 3-D surface Z = f(X,Y)
use meshgrid to compute arrays X and Y

**tan(x)**
tangent of x with argument in radians

**tand(x)**
tangent of x with argument in degrees

**tic, toc**
set and check a wall-clock timer

**true**
return logical 1 (true)

**type name**
return the function or built-in matching *name*

**version**
return Matlab interpreter version string

**who, who var, . . .**
display all or specified variables *var,* . . .

**whos, whos var, . . .**
long form of who; more detailed listing

**xlabel(str)**
print x-axis label *str* onto 2D plot

**ylabel(str)**
print y-axis label *str* onto 2D plot

**zeros(N), ones(N,M)**
return N×N or N×M matrix of zeroes