

1 februari 2012

DN1212 NumProg för T1 VT2012

Föreläsning 10: Datastrukturer & filer

Kap 6-8 i PEng

Idag:

- Komplexa tal
- Strängar
- Glesa matriser
- Posttabeller
- Biblioteket igen, nu med funktioner och data
- Celltabeller
- Celltabell som parameter

Komplexa tal

- Den imaginära enheten skrivs i eller j
- Funktioner kan ha komplexa parametrar
- real och imag ger real resp imaginärdel
- Exempel:

```
>> ko = sqrt(-1)+5
      ko = 5.0000 + 1.0000i
```

Strängar

- En sträng lagras som en vektor med tecken.
- Använd strcmp för att jämföra två strängar. Fler strängfunktioner finns i Tabell 6.3 i PEng (sid 284 i upplaga 3).

```
if strcmp(namn1,namn2)
    disp('Namnen är lika')
end
```

Posttabeller (structure arrays)

- I vektorer når man element med index.
- I posttabeller använder man namn.
- Exempel:

```
>>vara.namn='potatis';
>>vara.pris=5.90;
```

```

function bibliotek;
% Biblioteksprogrammet , nu med funktioner och posttabeller

global boklista;
lasInBocker;
fortsatt = 1;
disp('Välkommen! ');
anvandare = input('Skriv ditt användarnamn: ','s');

while fortsatt
    visaMeny;
    val = input('Vad vill du göra? ','s');
    valbokstav = upper(val(1));
    switch(valbokstav)
        case 'V'
            visa;
        case 'S'
            sok();
        case 'L'
            lana(anvandare);
        case 'R'
            returnera;
        case 'A'
            fortsatt=0;
            avsluta;
        otherwise
            disp('Felaktigt val, försök igen.');

```

```

    disp(boklista(i))
end;

function plats = sok();
% Söker i fälten titel och författare
% Utparameter är platsen där boken hittades, noll annars
global boklista;
plats = 0;
sokStrang = input('Ange titel eller författare: ','s');
for nr = 1:length(boklista)
    if strncmp(sokStrang,boklista(nr).titel,5)
        disp('Sökt titel funnen:');
        disp(boklista(nr));
        plats = nr;
    elseif strncmp(sokStrang,boklista(nr).författare,5)
        disp('Sökt författare funnen:');
        disp(boklista(nr));
        plats = nr;
    end;
end;
if plats == 0
    disp('Tyvärr, hittade ingen sådan bok.');
```

```

end;

function lana(anvandare);
% Hanterar utlåning av bok
% Ändrar boklista
global boklista;
disp('Vilken bok vill du låna?');
boknr = sok();
if boknr ~= 0
    namn = boklista(boknr).lanadAv;
    if strcmp(namn,'ingen') == 1
        boklista(boknr).lanadAv = anvandare;
        disp('Du har nu lånat boken.');
```

```

    else
        disp(['Boken är redan utlånad till ' namn]);
    end;
end;

function returnera();
% Hanterar återlämning av bok
% Ändrar boklista
global boklista;
disp('Vilken bok vill du återlämna?');
boknr = sok();
if boknr ~= 0
    boklista(boknr).lanadAv = 'ingen';
    disp('Du har nu återlämnat boken.');
```

```

end;

function avsluta();
% Inte klar - ska spara boklistan på fil
```

Glesa matriser

- Glesa matriser (sparse matrices) innehåller mest nollor
- Vanliga i tillämpningar, se t ex randvärdesproblemet i labb 8.
- Effektivare lagring av en gles matris fås med sparse
- Omvandla tillbaka med full
- Undersök om matrisen är gles med issparse

```
>> a=17*eye(1000);
>> glesa = sparse(a);
>> whos
  Name      Size      Bytes  Class
  a         1000x1000  8000000  double array
  glesa     1000x1000   16004    sparse array
```

- Vanliga matrisoperationer fungerar även för sparse-lagrade matriser

Celltabeller

- I vanliga matriser och vektorer har alla element samma typ
- I en celltabell kan cellerna innehålla värden av olika typ, t ex tal och tecken
- En cell kan även innehålla sammansatta data, t ex en vektor

```
>> tabell = {'krokodil',[1 0 0],pi,sqrt(-1)}
tabell =
    'krokodil'    [1x3 double]    [3.1416]    [0+ 1.0000i]
```

- Måsvingeparenteser används för att hänvisa till innehållet i en cell

```
>> tabell{2}
ans =     1     0     0
```

- För att komma åt element i en vektor i en cell sätter man vanliga indexparenteser sist:

```
>> tabell{2}(3)=5;
>> tabell{2}
ans =     1     0     5
```

Celltabell som parameter

- Om man som inparameter i en funktion anger varargin får man en funktion som kan ta emot ett godtyckligt antal anropsinparametrar
- ```
function persondata(varargin)
disp('Dina persondata är: ');
for nr = 1:nargin
 disp(varargin{nr});
 if ~ischar(varargin{nr})
 disp('Är det här din ålder eller ditt skonummer? ');
 end;
end;
```

## Enklaste filhanteringen i Matlab

- Kommandot `save choklad` sparar rubbet på filen `choklad`
- `save choklad x y` sparar bara variablerna `x` och `y`
- Kommandot `load choklad` laddar in all information från filen `choklad`
- Filerna går att flytta mellan olika datorer men inte till andra program än Matlab (och är svårlästa för mänskliga ögon).
- `save choklad -ascii` skriver ut på textformat
- `load choklad -ascii` läser in och tolkar filen som en textfil

## Öppna och stänga filer

- Öppna filen med `filnummer = fopen(filnamn)`
- Om nåt fel uppstår blir `filnummer = -1`
- Variant för felhantering: `[filnummer, felmeddelande] = fopen(filnamn)`
- Skapa ny utfil med: `filnummer = fopen(filnamn, 'w')`
- Utfil på textformat: `filnummer = fopen(filnamn, 'wt')`
- Kommandot `uigetfile` öppnar ett fönster där användaren kan välja fil
- Stäng filen med `fclose(filnummer)` och alla filer med `fclose('all')`

## Binärfiler

- Talen lagras med den binära representation som används i Matlab
- Fördelar: Effektivt minnesutnyttjande, snabbt att läsa in och skriva ut
- Nackdelar: Går inte att titta på, överföringsproblem mellan datorer och program som lagrar på olika sätt
- `fwrite(filnummer, skalbagge)` skriver ut värdet av variabeln `skalbagge`
- Olika lagringsformat kan användas, t ex `fwrite(filnummer, skalbagge, 'float32')`
- `elefant = fread(filnummer)` läser in och lagrar i variabeln `elefant`
- `elefant = fread(filnummer, 5)` läser fem värden (till `elefant`)
- `elefant = fread(filnummer, Inf)` läser alla värden (till `elefant`)

## Textfiler

- Textfiler kallas i boken formatted files
- Varje siffra i ett tal lagras som ett tecken, vilket innebär att ...
- ... ett 64-bitars flyttal lagras med 21 tecken, dvs i  $21 \cdot 16 = 336$  bitar
- Fördelar: Människoläsbart, går att flytta mellan olika program och olika datorer
- Nackdelar: Tar mer plats, tar längre tid att läsa in och skriva ut
- `fprintf(filnummer, format, variabler)` skriver ut precis som `printf`
- `lokomotiv = fscanf(filnummer, format)` läser in till variabeln `lokomotiv`
- Se formatkoder i tabell 8.7 (sid 374) och 8.10 (sid 381) i PEng
- `rad = fgetl(filnummer)` läser in en hel rad, `fgets` returnerar även radslutstecknet
- `feof(filnummer)` undersöker om filen är slutläst

## Bibliotekets filhantering

```
function lasInBocker(filnamn);
% Läser in böckerna från fil
global boklista;
infil = fopen(filnamn,'r');
nr = 1;
while ~feof(infil)
 boklista(nr).titel = fgetl(infil);
 boklista(nr).forfattare = fgetl(infil);
 boklista(nr).lanadAv = fgetl(infil);
 fgetl(infil);
 nr = nr + 1;
end;
fclose(infil);

function sparaBocker(filnamn);
% Sparar den ändrade boklistan på fil
global boklista;
utfil = fopen(filnamn,'w');
for nr = 1:length(boklista)
 fprintf(utfil, '%s\n', boklista(nr).titel);
 fprintf(utfil, '%s\n', boklista(nr).forfattare);
 fprintf(utfil, '%s\n', boklista(nr).lanadAv);
 fprintf(utfil, '\n');
end;
fclose(utfil);
```