Løsningsforslag for
Eksamen i TDT4105 IT Grunnkurs (Matlab)
9. desember 2019

Oppgave 1 – se løsningsforslag fra tilsvarende Python-eksamen i TDT4109.

Oppgave 2.1 Rett alternativ er: «/0023/ 24.00/25 /»

Oppgave 2.2 Rett alternativ er: «[1 2 3 1 2 3 4 8]»

Oppgave 2.3 Rett alternativ er: «[2,2,3,4,5,6,7,8,9]»

Oppgave 2.4 Rett alternativ er: «[3,4,6,7,3]»

Oppgave 2.5 Rett alternativ er: «[[1 4 8 13 5]; [2 6 11 17 6]; [3 8 14 21 7]]»

Oppgave 2.6 Rett alternativ er: «julegave»

Oppgave 2.7 Rett alternativ er: «TEIT OPPGAVE»

Oppgave 2.8 Rett alternativ er: «10»

Oppgave 2.9 Rett alternativ er: «9»

Oppgave 2.10 Rett alternativ er: funksjonen går i evig løkke

Oppgave 3.1

```
function res = leap_year( year )
  res = ~rem(year,4) ;
end
```

merk at denne som en bieffekt er vektoriserende, slik at et kall til leap_year(2009:2013) returnerer en sannhetsvekter, nærmere bestemt 0 0 0 1 0.

Oppgave 3.2

```
function res = days_in_years( start, stopp )
   res = 0 ;
   for i=start:stopp
      res = res + 365 + leap_year(i) ;
   end
end
```

En kortere – men kanskje ikke mer lesbar – versjon med den vektoreriserende versjonen av leap_year() :

```
  function res = days_in_years( start, stopp )
    res = sum( 365 + leap_year(start:stopp)) ;
  end
```

Oppgave 3.3

```
function res = days( date, month, year )
   global days_in_month ;
   res = date ;
   for i=1:month-1
      res = res+days_in_month(i) ;
      if i==2 && leap_year(year)
         res = res+1 ;
      end
   end
end
```

En litt kortere «one-liner-versjon» der løkka er implisitt er:

```
function res = days2( date, mon, year ) ;
   global days_in_month ;
   res = sum(days_in_month(1:mon-1)) + date + (leap_year(year) && mon>2) ;
end
```

Oppgave 3.4

```
function res = countdays( date, month, year )
  res = days(date, month, year) ;
  if (year>1901)
     res = res + days_in_years(1901,year-1) ;
  end
end
```

Merk en liten snubletråd her, siden dersom den kalles med year==1901, så vil den kalle days_in_years med parametrene (1901,1900), og avhengig av implementasjonen vil være lett å tenke seg en bug slik at det returneres -365. Slik det er gjort i fasitløsningen for oppgave 3.2 vil det returneres 0, så det går greit. Men gitt at man ikke tabber på det, kunne det forkortes til en one-liner:

```
function res = countdays2( date, month, year )
  res = days(date, month, year) + days_in_years(1901,year-1) ;
end
```

Oppgave 3.5

Her deler vi oppgaven i selve funksjonen count_text_days() og en hjelpefunksjon parse_date(). Vi ser nemlig at i senere deloppgaver trengs det funksjonalitet som ligner på, men ikke er helt lik det som count_text_days() gjør. Dessuten blir koden litt mer oversiktlig. Det har selvfølgelig ingen betydning for sensuren av denne deloppgaven hvorvidt man har gjort det eller ikke – men koden blir totalt sett kortere. Det er selvfølgelig også mulig å duplisere kode i de ulike deloppgavene, uten at det trekker på sensuren.

```
% HJELPEFUNKSJON
function [date, month, year] = parse_date( tdate )
  global monthnames ;
  global days_in_month ;
  word = strsplit( tdate, '-' ) ;

  date = str2double( word{1} ) ;
  year = str2double( word{3} ) ;
  monthname = lower( word{2} ) ;
  month = 0 ;
```

```
  for i=1:size(monthnames,1)
      if strcmp(monthname,monthnames(i,:))
          month = i ;
          break ;
      end
  end

  if month>0 && year>=1901 && year<=2099 && ...
          ((date>0 && date<=days_in_month(month)) || ...
           (leap_year(year) && month==2 && date==29))
       return
  else
     date = -1 ;
     fprintf('"%s" er ikke en gyldig datospesifikasjon\n', tdate) ;
  end
end

% HOVEDFUNKSJON
function res = count_text_days( tdate )
   [date, mon, year] = parse_date( tdate ) ;
   if date>0
       res = countdays( date, mon, year ) ;
   else
       res = -1 ;
   end
end
```

Her er det vanskeligere å skrive korte løsninger, fordi det er flere ulike ting som skal gjøres. Det viktigste er at man konverterer tekststrengene korrekt, og sjekker for om alle verdier er innenfor de lovlige intervaller. Teknisk sett kunne man også detektert gale strenger som '0003-jun-1901' eller '3.5-jun-1901' eller '3-jun-1901-blabla' osv, men det ikke lagt så mye vekt på det i vurderingen. Men løsningen må sjekke at månedsangivelsen er korrekt, at datoen er et positivt tall som ikke er større enn antall dager i den måneden, og at årstallet er innen forgyldig verdiområde.

Oppgave 3.6

```
function res = age_in_days( bdate, date2 )
   bdate_num = count_text_days( bdate ) ;
   date2_num = count_text_days( date2 ) ;

   res = -1 ;
   if bdate_num == -1 || date2_num == -1
       fprintf( "ugyldig datospesifikasjon til age_in_days\n" ) ;
   elseif bdate_num > date2_num
       fprintf( "første data kan ikke komme etter andre dato\n" ) ;
   else
       res = date2_num - bdate_num ;
   end
end
```

Her benytter vi oss av at count_text_days både konverterer og gir feilmelding, slik at vi ikke trenger å gjøre disse operasjonene om igjen.

Oppgave 3.7

```
function store_birthdate( tdate )
   if count_text_days( tdate ) == -1
       return ;
   end
```

```
    fid = fopen( 'birthdate.txt', 'w' ) ;
    if fid == -1
        fprintf( 'Kunne ikke åpne fila\n' ) ;
    else
        fprintf( fid, '%s\n', tdate ) ;
        res = fclose( fid ) ;
        if res == -1
            fprintf( 'Kunne ikke lukke fila\n' ) ;
        end
    end
end
```

Merk at i denne løsningen så sjekkes formatet ved å kalle count_text_days() som sjekker og gir feilmelding. Merk også at vi her lagres unna tekstversjonen. Det ville vært mulig å lagre unna tre tall for dato, måned og år, eller for så vidt også antall dager fra 1. jan 1901. Det siste ville være det minst taktiske, fordi det er vanskeligere å konvertere fra et antall dager til en dato enn det er å gå motsatt vei.

Oppgave 3.8

Leser vi videre på neste deloppgave, ser vi at det kan være nyttig å introdusere en hjelpefunksjon som leser inn og returnerer strengen som ligger lagret i fila, siden den vil kunne gjenbrukes der. Dessuten øker det lesbarheten når vi skal gjøre to ting samtidig: lese fra fil og regne ut antall dager. Vi introduserer derfor hjelpefunksjonen read_filedate()

```
% HJELPEFUNKSJON
function res = read_filedate()
    fid = fopen( 'birthdate.txt', 'r' ) ;
    res = '' ;
    if fid ~= -1 && ~feof( fid )
        res = fgetl( fid ) ;
        if fclose( fid ) == -1
            fprintf( 'Kunne ikke lukke fila\n' ) ;
        end
    else
        fprintf( 'Kunne ikke åpne fila\n' ) ;
    end
end

% HOVEDFUNKSJON
function res = my_age_in_days( tdate )
    res = -1 ;
    date2 = count_text_days( tdate ) ;
    bdate = count_text_days( read_filedate()) ) ;
    if date2>0 && bdate>0
       if bdate > date2
          fprintf( 'datoen må komme etter fødselsdatoen\n' ) ;
       else
          res = date2 - bdate ;
       end
    end
end
```

Det har ingen innvirkning på sensuren om du her skriver alt i en funksjon og/eller dupliserer kode fra tidligere deloppgaver fremfor å benytte hjelpefunksjoner.

Oppgave 3.9

Gitt at vi introduserte hjelpefunksjonene parse_date() og read_filedate() i tidligere deloppgaver, så blir birthdate() som følger:

```
function birthdate( tdate )
   [date, mon, year] = parse_date( tdate ) ;
   [bdate, bmon, byear] = parse_date(read_filedate()) ;
   if date>0 && bdate>0
       next_byear = year ; % årstall for neste fødselsdag
       if (mon>bmon || mon==bmon && date>bdate)
           % neste fødselsdag kommer i påfølgende år
           next_byear = next_byear + 1 ;
       end

       if countdays(bdate,bmon,byear)>countdays(date,mon,year)
           fprintf( 'oppgitt dato er før vedkommende er født\n' ) ;
       else
          fprintf( 'Du er født på dag %d i måned %d i året %d.\n', ...
                 bdate, bmon, byear ) ;
          days = countdays( bdate, bmon, next_byear ) - ...
                    countdays( date, mon, year) ;
          fprintf( 'Det er %d dager til fødselsdagen din, ', days ) ;
          fprintf( 'og du blir %d år.\n', next_byear - byear ) ;
       end
   end
end
```

For sensurens del er det selvfølgelig helt greit om du dupliserer kode fra tidligere deloppgaver her, fremfor å bruke tidligere definerte hjelpeoppgaver.

Oppgave 3.10

Her skal vi telle antall skuddårsdager mellom to datoer, som er antall skuddår i de mellomliggende hele årene, samt muliger skuddårdager i startåret og sluttåret. Underforutsetning av at vi i tidligere deloppgaver har laget hjelpefunksjonene parse_date() og read_filedate()

```
function res = leap_days( tdate )
   [date, mon, year] = parse_date( tdate ) ;
   [bdate, bmon, byear] = parse_date(read_filedate()) ;

   leaps = 0 ;
   if date>0 && bdate>0
      if countdays(bdate, bmon, byear)<=countdays(date, mon, year)
         for i=byear+1:year-1
            if leap_year(i)
               leaps = leaps + 1 ;
            end
         end

         if year==byear
             if leap_year(year) && bmon<=2 && mon>2
                 leaps = leaps + 1 ;
             end
         else
             if leap_year(byear) && bmon<=2
                leaps = leaps + 1 ;
             end
             if leap_year(year) && mon>2
```

```
                leaps = leaps + 1 ;
            end
        end
        fprintf( 'Du har opplevd %d skuddårsdager.\n', leaps ) ;
    else
        fprintf( 'Dato kan ikke være før fødselsdato\n') ;
    end
  end
end
```

For sensurens del er det helt greit om du dupliserer kode fra tidligere deloppgaver fremfor å bruke hjelpefunksjonene.

Dersom du laget en vektoriserende versjon av leap_year(), kan initialisering av leaps og for-løkka byttes ut med

```
leaps = sum(leap_years(byear+1:year-1)) ;
```