i

Framside

Institutt for datateknologi og informatikk

Eksamensoppgave i TDT4105 - Informasjonsteknologi, grunnkurs (matlab)

Målform: Bokmål

Faglige kontakter under eksamen:

- Anders Christensen (tlf.: 918 97 181)
- Rune Sætre (tlf.: 45 21 81 03)

Eksamensdato: 9. desember 2019

Eksamenstid (fra-til): 09:00 – 13:00

Hjelpemiddelkode/Tillatte hjelpemidler:

D Ingen trykte eller håndskrevne hjelpemidler tillatt. Bestemt, enkel kalkulator tillatt.

Annen informasjon:

Det er angitt i prosent hvor mye hver deloppgave i eksamenssettet teller ved sensur. Maksimum antall poeng man kan få på denne oppgaven er 200 poeng. Les gjennom hele oppgavesettet før du begynner å løse oppgaven. Disponer tiden godt! Gjør rimelige antagelser der du mener oppgaveteksten er ufullstendig hvis mulig.

Merk! Studenter finner sensur i Studentweb. Har du spørsmål om din sensur må du kontakte instituttet ditt. Eksamenskontoret vil ikke kunne svare på slike spørsmål.

1

Oppgave 1 Flersvar (20%)

Velg det svaret du mener er mest riktig av alternativene. For hvert spørsmål gis det poeng på følgende måte:

- Korrekt avkrysning 2 poeng
- Feil avkrysning -1
- Ingen avkrysning -1 poeng

Det er ikke mulig å få under 0 poeng totalt.

Der det er spesielle uttrykk står den engelske oversettelsen i parentes.

Oppgave 1.1

Hva er *Photolithography* (fotolitografi)?

Velg ett alternativ

- En digitalisering av hullkort (Punch Cards) som har gjort det mulig for en datamaskin å lese digital informasjon.
- En teknikk som har gjort at integrerte kretser i transistorer har blitt mer suksessfulle, hvor ledninger og andre deler printes på chipene i flere lag.
- En type RAM som lagrer data permanent.

Oppgave 1.2

Hva er *ENIAC*?

Velg ett alternativ

- Komponenten i en datamaskin som er ansvarlig for regneoperasjoner.
- Navnet på den første transistoren som ble laget.
- Verdens første programmerbare elektroniske datamaskin.

Oppgave 1.3

Hva sier Moores lov?

Velg ett alternativ

- Loven sier at ting vil gå galt uavhengig av situasjon hvis gitt muligheten.
- Loven sier at antall transistorer i en integrert krets dobles hvert 2. år
- Loven sier at samplingsfrekvensen må være minst dobbelt så rask som den raskeste frekvensen.

Oppgave 1.4

Hvilket steg i “Fetch/Execute Cycle” flytter verdiene fra RAM (minnet) til ALU?

Velg ett alternativ

- Instruction Execute
- Data Fetch
- Instruction Fetch

Oppgave 1.5

Hva er hovedoppgaven til *ALU*?

Velg ett alternativ

- Utføre regneoperasjoner
- Peke på riktig minneadresse
- Hente data fra minnet

Oppgave 1.6

Hva er sant om *run-length-koding*?

Velg ett alternativ

- Run-length-koding er tapsløs komprimering, dvs. at den originale representasjonen av 0ere og 1ere kan bli rekonstruert perfekt fra den komprimerte versjonen.
- Run-length-koding er taps-komprimering, dvs. at den originale representasjonen ikke kan rekonstrueres eksakt fra den komprimerte versjonen.
- Run-length-koding vil alltid lønne seg, siden den reduserer antall bits som trengs for å lagre informasjonen til en tidel av det den var.

Oppgave 1.7

Hvilket binærtall representerer det heksadesimale tallet D52F?

Velg ett alternativ

- 1101 0101 0010 1111
- 1111 1101 0101 0010
- 0101 1111 1101 0010

Oppgave 1.8

Et bilde har 1920x1080 piksler i 32bit fargeformat. Hvor mye plass trenger det ukomprimert?
Velg ett alternativ

- Omtrent 4MB
- Omtrent 8MB
- Omtrent 66MB

Oppgave 1.9
Hva blir -85 i 2ers-kompliment representert med 8 bit?
Velg ett alternativ

- 1010 1010
- 1010 1011
- 1101 0101

Oppgave 1.10
Hvor mange symboler kan representeres med 9 bit?
Velg ett alternativ

- 256
- 511
- 512

Oppgave 1.11
Hva er *DoS* (Denial-of-Service)?
Velg ett alternativ

- Å lure en bruker til å oppgi personlige opplysninger som blir utnyttet av angriperen ved å gi brukeren feilaktig informasjon.
- Å gjøre en tjener (server) utilgjengelig for brukerne ved å sende enorme mengder datapakker.
- At angriper nekter å stå bak et innbrudd på en nettverkstjener (server).

Oppgave 1.12
Hva er *spoofing*?
Velg ett alternativ

- Å spionere på en bruker ved å overstyre webkamera på en datamaskin.
- Å spionere på en bruker ved å ta opp lyd ved hjelp av datamaskinens lydkort.
- Bruk av falsk IP-kildeadresse for å lure mottaker til å prosessere pakken, ved å f.eks. benytte seg av falske e-postavsendere eller å misbruke domenenavn.

Oppgave 1.13
Hvordan vurderer man *dataintegritet* (hva vurderes ved dataintegritet)?
Velg ett alternativ

- Sjekker om dataene er beskyttet mot uautorisert tilgang.
- Sjekker om dataene som sendes er identiske med dataene som mottas.
- Sjekker i hvilken grad anonymitet hos senderen blir ivaretatt.

Oppgave 1.14

Hva er **IKKE** sant om *hashing*?
Velg ett alternativ

- Det genereres en hash-kode av meldingen ved hjelp av en krypteringsnøkkel som avsender merker meldingen med.
- Mottaker bruker en krypteringsnøkkel for å undersøke om samme hash-kode kan genereres fra meldingen.
- Hashing avdekker det meste av endringer gjort med en melding, men kan ikke garantere 100% at en endring har skjedd.

Oppgave 1.15
Hvordan fungerer kryptering med *offentlig nøkkel* (public key encryption)?
Velg ett alternativ

- En offentlig nøkkel brukes både til å kryptere og dekryptere en melding.
- Partene deler en hemmelig nøkkel som brukes både for kryptering og dekryptering.
- Hver part får en hemmelig og en offentlig nøkkel. En melding kryptert med en offentlig nøkkel kan kun dekrypteres med den korresponderende private nøkkelen.

Oppgave 1.16
Hvilken form for *kanalkoding* (channel coding) bruker Internettet vanligvis?
Velg ett alternativ

- Hamming Coding
- 16-bit sjekksum (16-bit checksum)
- Single Parity Checking (SPC)

Oppgave 1.17
Hva er **IKKE** sant om *nettverks-topologier*?
Velg ett alternativ

- I en ring-topologi sendes pakkene gjennom ringen til de når rett mottaker, noe som gjør det enklere å koordinere aksess og detektere nettverksproblemer.
- I en buss-topologi er alle datamaskiner i nettverket koblet til ett felles medium (kabel), og hver datamaskin kan sende data til hvilken som helst annen datamaskin i samme lokalnettverk til samme tid.
- I en mesh-topologi er det en direkte kobling mellom hver datamaskin i nettverket, og dermed blir ikke resten av nettverket påvirket om en node skulle slutte å virke.

Oppgave 1.18
Hva er sant om *MAC-adresser*?
Velg ett alternativ

- Alt utstyr koplet til internett får utdelt en MAC-adresse fra en Domain Name Server (DNS).
- En stor ulempe er at MAC-adresser ikke fungerer for datamaskiner som kjører Linux eller Windows.
- Hver datamaskin i et pakkesvitsjet lokalnettverk har en unik MAC-adresse.

Oppgave 1.19
Hva er **IKKE** sant om *rutere*?

Velg ett alternativ

- Rutere er maskinvare med prosessor, minne og I/O-grensesnitt som brukes for å koble sammen ulike typer nettverk og nettverksteknologier.
- Rutere bruker protokoller for å støtte kommunikasjon mellom ulike typer nettverk.
- Rutere er en teknologi for å knytte sammen ulike typer lokalnettverk, men fungerer ikke på Wide Area Networks (WANs).

Oppgave 1.20

Hvilken av følgende standarder sørger for pålitelig overføring av pakker over internettet?

Velg ett alternativ

- TCP
- IP
- UDP

Maks poeng: 40

Oppgave 2 Kodeforståelse (30%)

Velg det svaret du mener er mest riktig av alternativene. For hvert spørsmål gis det poeng på følgende måte:

- Korrekt avkryssing 6 poeng
- Feil avkryssing 0 poneg
- Ingen avkryssing 0 poeng

2

Oppgave 2.1 (3%)

Hva blir skrevet til terminal når funksjonen myst1 kalles som:

myst1(23, 24, '25')

```
function myst1(b,c,d)
   fprintf( '/%04d/%6.2f/%-3s/\n', b, c, d ) ;
end
```

Velg ett alternativ

- /23.0/24.00/25/
- (det gis en feilmelding)
- /23/24/25/
- /0023/ 24.00/25 /
- /+23/2.4E1/ 25 /
- /23 /+24.00/ 25/

Maks poeng: 6

3

Oppgave 2.2 (3%)

Hva blir returnert dersom funksjonen myst2 kalles som:

myst2([1:8], [1:5])

```
function a=myst2(a,b)
   a(end-(length(b)-1):(end-1)) = b(1:(end-1)) ;
end
```

Velg ett alternativ

- [1 2 3 4 2 3 4 8]
- [1 2 3 1 2 3 4 8]
- [1 2 5 5 5 5 7 8]
- [1 2 3 4 5]
- [1 2 5 4 3 2 1 7 8]

Maks poeng: 6

4 Oppgave 2.3 (3%)

Hva blir skrevet ut til terminal når følgende funksjon blir kjørt som **myst3([1,2,3,4,5,6,7,8,9])** ?

```
function a=myst3(a)
   for i=2:length(a)
      a(i-1) = a(i) ;
      return ;
   end
end
```

Velg ett alternativ

- [2,2,3,4,5,6,7,8,9]
- [1,1,1,1,1,1,1,1,1]
- [9,8,7,6,5,4,3,2,1]
- [2,3,4,5,6,7,8,9,9]
- [0,1,2,3,4,5,6,7,8]

Maks poeng: 6

5

Oppgave 2.4 (3%)

Hva blir skrevet ut til terminal når følgende kode blir kalt som **myst4([1,2,3,4,5])** ?

```
function a=myst4(a)
   for i=1:length(a)-1
      if a(i) < a(i+1)
         t = a(i+1) ;
         a(i+1) = a(i+1) - a(i) ;
         a(i) = a(i) + t ;
      end
   end
end
```

Velg ett alternativ

- [1,2,3,4,5]
- [3,4,7,8,9]
- [3,5,7,9,11]
- [4,3,6,3,1]
- [5,4,3,2,1]
- [3,4,6,7,3]

Maks poeng: 6

6

Oppgave 2.5 (3%)

Hva blir skrevet til terminal når følgende funksjon blir kalt som **myst5([[1 2 3 4 5]; [2 3 4 5 6]; [3 4 5 6 7]])** ?

```
function a=myst5(a)
   for i=1:size(a,1)
      for j=2:size(a,2)-1
         a(i,j) = a(i,j-1) + a(i,j+1) ;
      end
   end
end
```

Velg ett alternativ

- [[1 10 8 6 5]; [2 12 10 8 6]; [3 14 12 10 7]]
- [[1 4 8 13 5]; [2 6 11 17 6]; [3 8 14 21 7]]
- [[1 16 14 12 5]; [2 16 14 12 6]; [3 8 6 4 7]]
- [[1 3 3 3 5]; [2 4 4 4 6]; [3 5 5 5 7]]
- [[1 2 3 4 5]; [2 3 4 5 6]; [3 4 5 6 7]]
- [[1 16 14 12 5]; [2 12 10 8 6]; [3 8 6 4 7]]

Maks poeng: 6

7

Oppgave 2.6 (3%)

Hva blir returnert når følgende funksjon blir kalt som:
myst6('peleadlspvpsvasoasgaspgtsueetrtlisuoj')

```
function r=myst6(s)
   r = '' ;
   y = 1 ;
   x = length(s) ;
   while x>0
      r(end+1) = s(x) ;
      y = y + 1 ;
      x = x - y ;
   end
end
```

Velg ett alternativ

- ostepopp
- julegave
- jousilt
- plassgul
- ostegass

Maks poeng: 6

8

Oppgave 2.7 (3%)

Hva blir returnert dersom følgende funksjon blir kalt som:

myst7('KTUELITTLPL', 'SDOOSPPJPPDGLGAESVGSEGV')

```
function s=myst7(a,b)
   s=[a(2:2:end-2), ' ', b(3:3:end)] ;
end
```

Velg ett alternativ

- KTUE SDOOSPPJ
- TEIT OPPGAVE
- KULT OPPLEGG
- ULTL SOPPLEGG
- KUTT SOLDUGG

Maks poeng: 6

9

Oppgave 2.8 (3%)

Hva blir returnert dersom funksjonen myst8 blir kalt som:

myst8(4)

```
function d=myst8(a)
   if a==1
      d = a ;
   else
      b = a-1 ;
      c = myst8(b) ;
      d = c+a ;
   end
end
```

Velg ett alternativ

- 10
- 6
- 16
- 1
- 3
- -1

Maks poeng: 6

10

Oppgave 2.9 (3%)

Hva blir returnert dersom funksjonen myst9 blir kalt:

myst9([1,2,6], [1,2,6,3,2,2,3,5,1,2,6,4,7,2])

```
function x=myst9(a,b)
   x = 0 ;
   for y = a
      for z = b
         if y==z
            x = x + 1 ;
         end
      end
   end
end
```

Velg ett alternativ

- 6
- 12
- 1
- 9
- 2

Maks poeng: 6

11

Oppgave 2.10 (3%)

Hva blir returnert fra funksjonen myst10 når den kalles som:

myst10(-3)

```
function s=myst10(a)
   x = 20 ;
   s = 0 ;
   while x>0
      s = x+a ;
      x = x-a ;
   end
end
```

Velg ett alternativ

- 7
- (funksjonen går i evig løkke)
- 2
- -2
- 20

Maks poeng: 6

Useful Matlab functions and commands

blanks - String of blanks. *blanks(n)* is a string of n blanks. Use with *disp()*, e.g. *disp(['xxx' blanks(20) 'yyy'])*

cell2mat - Converts a cell array into an ordinary array. The elements of the cell array must all contain the same data type, and the resulting array is of that data type.

fix - Round towards zero. *fix(x)* rounds the elements of *x* to the nearest integers towards zero.

fclose - Close file. *st = fclose(fid)* closes the file associated with file identifier *fid*, which is an integer value obtained from an earlier call to *fopen()*. *fclose()* returns 0 if successful or -1 if not.

feof - Test for end-of-file. *st = feof(fid)* returns 1 if the end-of-file indicator for the file with file identifier *fid* has been set, and 0 otherwise. The end-of-file indicator is set when a read operation on the file associated with the *fid* attempts to read past the end of the file.

fgetl - read line from file, discard newline character. *tline = fgetl(fid)* returns the next line of a file associated with file identifier *fid* as a MATLAB string. The line terminator is NOT included. Use *fget*s() to get the next line with the line terminator INCLUDED. If just an end-of-file is encountered, -1 is returned.

find - Returns the linear indexes of non-zero elements in a matrix. *find([0 1 0 1 0])* returns [2 4]. If the first parameter has more than one row, a column vector containing the linear indexes of non-zero elements are returned. An optional second parameter set the maximum number of indexes to return.

fopen - Open file. *fid = fopen(filename,permission)* opens the file *filename* in the mode specified by PERMISSION:

'r' - open file for reading
'w' - open file for writing; discard existing contents
'a' - open or create file for writing; append data to end of file
'r+' - open (do not create) file for reading and writing
'w+' - open or create file for reading and writing; discard existing contents
'a+' - open or create file for reading and writing; append data to end of file

fprintf - Write formatted data to file. *count = fprintf(fid,format,A,...)* formats the data in the real part of array A (and in any additional array arguments), under control of the specified *format* string, and writes it to the file associated with file identifier *fid*. *count* is the number of bytes successfully written. *fid* is an integer file identifier obtained from *fopen*(). It can also be 1 for standard output (the screen) or 2 for standard error. If *fid* is
omitted, output goes to the screen. *format* is a string containing ordinary characters and/or C language conversion specifications. Conversion specifications involve the character %, optional flags, optional width and precision fields, optional subtype
specifier, and conversion characters d, i, o, u, x, X, f, e, E, g, G, c, and s. The special formats \n, \r, \t, \b, \f can be used to produce linefeed, carriage return, tab, backspace, and formfeed characters respectively. Use \\ to produce a backslash character and %% to produce the percent character.

global - Define global variable. *global X Y Z* defines *X*, *Y*, and *Z* as global in scope (scope can be functions/programs).

input - Read a value from the keyboard and into a variable. *answer=input(str)* prints str as a prompt, reads a number and assigns it to *answer*. If character string is to be read, use the optional second parameter ‘s’.

isempty - Determine whether array is empty This MATLAB function returns logical 1 (true) if A is an empty array and logical 0 (false) otherwise. *TF = isempty(A)*

length - The length of vector. *length(X)* returns the length of vector X. It is equivalent to *max(size(X))* for non-empty arrays and 0 for empty ones. use size(X,1) and size(X,2) etc to get the size in specific dimensions.

load - Loads data from filename. *load(filename)* loads data from filename. If filename is a MAT-file, then *load(filename)* loads variables in the MAT-File into the MATLAB® workspace. If filename is an ASCII file, then *load(filename)* creates a double-precision array containing data from the file.

max - finds the highest element in a vector, or the highest element in each column of a matrix.

min - finds the lowest element in a vector, or the lowest element in each column of a matrix.

mod - Modulus after division. *mod(x,y)* is x - n.*y where *n = floor(x./y)* if y ~= 0.

num2str - Convert numbers to a string.

randi - Pseudorandom integers from a uniform discrete distribution. *R = randi(IMAX,N)* returns an N-by-N matrix containing pseudorandom integer values drawn from the discrete uniform distribution on *1:IMAX*. Further: *randi(IMAX,M,N)* or *randi(IMAX,[M,N])* returns an M-by-N matrix.

rem - Remainder after division. *rem(x,y)* is *x - n.*y* where *n = fix(x./y)* if y ~= 0.

repmat - returns a matrix constructed from concatenating its first parameter multiple times. For instance *repmat(foo, 3, 4)* replicates *foo* three times vertically and four times horisontally. The function *repmat('ab',1,3)* returns 'ababab'.

round - Rounds to nearest decimal or integer. *Y = round(X)* rounds each element of *X* to the nearest integer. If an element is exactly between two integers, the round function rounds away from zero to the integer with larger magnitude. *Y = round(X,N)* rounds to N digits

size - The size of array. *D = size(X)*, for M-by-N matrix X, returns the two-element row vector. D = [M,N] containing the number of rows and columns in the matrix. Further, *size(X,D)* gives the size of *X* in dimention *D*, where 1 means number of rows and 2 means number of columns.

sortrows - Sort array rows. This MATLAB function sorts the rows of a matrix, table or cell arry in ascending order, based on values in first column: *B = sortrows(A)*. More generally: *B = sortrows(A [, column [, order]])*. Here, *column* can be a vector of integers, where the second (third etc) element gives the secondary (tertiary etc) column that determine the sort order for elements that have the same value in the primary sort colum. If *order* is present, it must be either 'ascending' or 'descending' to specify the sorting order.

sprintf - Basically the same as *fprintf()* without the first, optional parameter fid, but instead of printing to the standard output, it returns the constructed string value.

sscanf - Extracts values from a string according to a format string. Opposite of *fprintf()*.
A=sscanf('12/11-2014','%d/%d-%d') returns a column vector containing the values 12, 11, and 2014.

strcmp - Compare strings. *TF = strcmp(S1, S2)* compares the strings *S1* and *S2* and returns logical 1 (true) if they are identical, and returns logical 0 (false) otherwise.

strsplit - Splits the first (string) parameter into a cell array of substrings, according to the delimiter string given as the second parameter. *strsplit('one, two, three', ', ')* results in {'one', 'two', 'three'}. Multiple alternative delimiters can be specified using a cell array as the second parameter.

strtok - separates the first token of a string from the rest of that string.
[token, rest] =strtok (' first second', delim) sets *token* to 'first' and *rest* to ' second'. The optional parameter *delim* contains a list of delimiter characters – where the space character is default. Any delimiter characters before the first token are ignored, but delimitors between first token and the rest of the string is preserved at the front of *rest*.

str2num - Convert string matrix to numeric array. *X = str2num(S)* converts a character array representation of a matrix of numbers to a numeric matrix. For example, if *S=['12'; '34']* then *str2num(S)* => [12; 34]. Further, if *S='abc'* then *str2num(S)=> []*

sum - The sum of elements. *S = sum(X)* is the sum of the elements of the vector *X*. If *X* is a matrix, *S* is a row vector with the sum over each column.

i

Oppgave 3 Kalender (50%)

I denne programmeringsoppgavene skal du lage ulike funksjoner som skal regne på kalender. Det skal tas hensyn til skuddår, som er år som har 29. februar som en ekstra dag. For å gjør det enkelt opererer vi kun med årene fra og med 1901 til og med 2099, der skuddår er alle år som er delelige med 4. (Merk at også år 2000 var et skuddår)

Husk at du kan gjenbruke funksjoner fra tidligere deloppgaver i de senere deloppgavene. Selv om du svarer feil eller ikke svarer på en deloppgave, kan du allikevel bruke denne funksjonen uten at du får trekk i senere deloppgaver - under forutsetning av at du kaller den på korrekt måte.

Alle oppgavene skal løse på en generell måte og det skal unngås å bruke unødig repetitiv kode.

12

3.1 Skuddår (5%)

Skriv en funksjon **leap_year(year)** som tar et årstall i tidsrommet 1901-2099 som parameteren **year**, som er et heltall, og returnerer en boolsk sannhetsverdi som retur-verdi. Denne returverdien er sann dersom året som er angitt som parameter er et skuddår. I intervallet 1901-2099 vil alle år som er delelige med 4 være skuddår, dvs 2004, 2008, 2012 osv.

Du kan anta at inn-parameter er et år i det gyldige intervallet, slik at du slipper å sjekke for ugyldige verdier av parameteren **year**.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

13

3.2 Dager i årene (5%)

Skriv en funksjon **days_in_years(startyear, stopyear)** som returnerer det totale antallet dager i de hele årene som starter med år **startyear** og ender med **stopyear**. Alle dagene i både **startyear** og **stopyear** skal være med. Du kan anta at **stopyear** ikke er lavere (dvs før) **startyear**, slik at du slipper å sjekke for dette. Du kan også anta at begge parametere er innen de gyldige området 1901-2099. Funksjonen skal ta hensyn til skuddår.

For eksempel vil **days_in_years(2009,2011)** returnere 1095, som er 3x365. Men **days_in_years(2008,2009)** returner 731 som er 365+366 siden 2008 er skuddår.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

14

3.3 Dager frem til dato (5%)

Skriv en funksjon **days(date, month, year)** som teller antallet dager fra starten av et år som er angitt ved **year**, og frem til den datoen som er angitt ved dato **date** i måneden **month** i det samme året.

Både **date** og **month** og **year** er heltall. Du skal ta hensyn til om året er skuddår.

Du kan anta at det finnes en global variabel som du velge å importere. Den heter **days_in_month**, og er av type vektor (dvs endimensjonal matrise) av heltall, og den angir antallet dager i hver måned for et år som ikke er skuddår. De fire første verdiene i den vil være 31, 28, 31 og 30 for antall dager i januar, februar (uten skuddår), mars og april.

Dagen som er angitt av parameterne til funksjonen skal telles med.

For eksempel vil **days(15,2,2009)** returnere 46, **days(15,3,2009)** skal returnere 74, mens **days(15,3,2008)** skal returnere 75.

Du kan anta at inn-parameterne angir en gyldig dato, slik at du slipper å sjekke for ugyldige verdier og år.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

15

3.4 Dager før dato (5%)

Skriv en funksjon **countdays(date, month, year)** som teller antall dager fra og med 1. januar 1901 og frem til og med dagen som er angitt av dato **date** i måneden **month** i året **year**. De tre parameterne **date**, **month** og **year** er alle heltall.

Du kan anta at alle inn-parameterne angir en gyldig dato, så du slipper å sjekke for ugyldige parametere. Du skal ta hensyn til skuddår.

For eksempel vil **countdays(15,2,1902)** returnere 411, som er 365 dager i 1901 og 46 dager frem til 15. februar i 1902.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

16

3.5 Dato som tekststreng (5%)

Brukerne elsker funksjonen for å regne ut antall dager fra og med 1. jan 1901 og frem til en oppgitt dato som du skrev i forrige deloppgave. Imidlertid vil de gjerne oppgi datoen som en tekststreng på formatet 'date-mon-year' der 'mon' er de tre første bokstavene i navnet på måneden. Det skal brukes bindestrek til å skille de tre elementene. Datoen i måneden kan godt være ett siffer dersom det er før den tiende i den måneden. Angivelsen av måneden kan være i en vilkårlig blanding av store og små bokstaver, slik at 'jan', 'Jan', 'JAN' og til og med 'jAN' alle angir januar. Eksempler på gyldige datostrenger er '1-jan-2013' og '31-JUL-2099' og '29-fEB-2020'.

Skriv en funksjon **count_text_days(date)** som tar en slik datospesifikasjon **date** som tekststreng på det formatet som er angitt over, og returnerer antall dager fra og med 1. januar 1901 og frem til og med den angitte dagen.

Når du løser denne oppgaven, skal du benytte en global variabel som heter **monthnames**, og som er et 12 x 3 char array som angir de tre første bokstavene i hver måneds navn. Vi forutsetter at variabelens verdi er tilpasset brukerens språk, slik der den norske varianten er **['jan'; 'feb'; 'mar'; 'apr'; 'mai'; 'jun'; 'jul'; 'aug'; 'sep'; 'okt'; 'nov'; 'des']**. Med andre ord, du skal ikke definere de enkelte månedsnavnene selv i funksjonen, men benytte den globale variabelen **monthnames**.

Du skal sjekke om inn-parameteren spesifiserer en gyldig dato. Husk at du kan bruke den globale variabelen **days_in_month** (fra en tidligere deloppgave) for en vektor av antall dager i hver måned for ikke-skuddår. Dersom inn-parameteren angir en ugyldig dato, skal det gis en feilmelding og det skal returneres -1.

For eksempel vil et kall til **count_text_days('15-Feb-1902')** returnere 411.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

17

3.6 Alder i dager (5%)

Skriv en funksjon **age_in_days(born, date)** som tar to datospesifikasjoner på strengformat slik som det ble angitt i forrige oppgave, der **born** er den første datoen og **date** er en senere dato. Funksjonen skal returnere det antallet dager som er mellom de to angitte datoene.

For eksempel, **age_in_days('15-jun-2015', '20-JUN-2016')** skal returnere 370, som er fem dager mer enn et helt år. Videre skal **age_in_days('15-jun-2015', '16-jun-2015')** returnere 1.

En feilmelding skal gis dersom dato-parameteren **born** er etter dato-parameteren **date**, og funksjonen skal da returnere -1. Det skal tas hensyn til skuddår. Feilmelding skal gis og det skal returneres -1 dersom minst en av datoene er ugyldige.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

18

3.7 Lagre til fil (5%)

Mange av brukerne er bare interessert i å bruke funksjonen fra forrige deloppgave på sin egen alder. De har lyst til å unngå å taste inn fødselsdatoen sin hver gang. Du skal nå lage et system som lagrer unna fødselsdatoen til en fil som heter 'mybirthdate.txt', slik at den senere kan hentes frem senere. Du kan selv bestemme formatet på hvordan dataene lagres i denne fila, men husk at du senere skal lese inn fila og tolke datoene.

Skriv funksjonen **store_birthdate(date)** som lagrer en dato, angitt ved parametereren **date**, i fila mybirtdate.txt. Funksjonen skal ikke returnere noe. Parameteren **date** er på det samme tekstformatet som ble brukt i forrige deloppgave.

Dersom fila mybirthdate.txt finnes fra før, skal den overskrives, og samtidig skal du gi en advarsel om at gammel dato i fila er overskrevet. Ved feil på åpning eller lukking av fila skal det gis en feilmelding.

Du skal verifisere at datoen angitt som parameter til store_birthdata() er en gyldig dato, og dersom den ikke er det, skal det ikke lagres noe til fil, men en feilmelding skal skrives ut.

Et eksempel på kall til funksjonen kan være **store_birthdate('2-aug-2001')** som lagrer informasjon om at 2. august 2001 er brukerens fødselsdag.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

19

3.8 Hente fra fil (5%)

Skriv nå funksjonen **my_age_in_days(date)** som forutsetter at det finnes en fil som heter mybirthdate.txt, slik som ble skrevet til i forrige oppgave. Funksjonen tar parameteren **date**, som er en datospesifikasjon på strengformat som brukt i tidligere deloppgaver. Den skal regne ut og returnere alderen i antall dager utfra datospesifikasjonen i parameteren **date** og fødselsdatoen som er lagret på fila mybirthdate.txt.

Dersom man ikke får lest fila mybirthdate.txt eller dersom **date** er tidligere enn fødselsdatoen på mybirthdate.txt, eller dersom datospesifikasjonen **date** er ugyldig, så skal det gis en beskrivende feilmelding, og det skal returneres -1.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

20

3.9 Skriv ut fødelsdato (5%)

Skriv en funksjon **birthdate(date)** som leser fila mybirthdate.txt - dersom den finnes. Parameteren **date** er en tekststreng som angir en dato, f.eks '9-des-2019'. Funksjonen skal skrive ut to linjer med tekst på følgende format:

Du er født på dag nr x i måned nr y i året zzzz.
Det er n dager til du har fødselsdag og da blir du m år.

Her er x, y, zzzz, n og m tall, der du skal bruke riktig verdi i utskriften. Dersom **date** angir fødselsdagen, skal utskriften angi at det er 0 dager til fødselsdagen.

Et eksempel på utskrift er gitt under, gitt at mybirthdate.txt angir at du er født 13. desember 2001:

Du er født på dag 13 i måned 12 i året 2001.
Det er 4 dager til fødselsdagen din, og du blir 18 år.

Dersom mybirthdate.txt ikke kan åpnes, eller dersom datoen som er gitt av **date** er ugyldig skal det også gis en beskrivende feilmelding, men det skal ikke skrives ut noe utover feilmeldingen.

Skriv ditt svar her...

```
1
```

Maks poeng: 10

21

3.10 Skuddårsdager (5%)

Du skal nå skrive en funksjon **leap_days(date)** som teller antall skuddårsdager (dvs antall dager som faller på 29. februar) som brukeren vil ha opplevd fra fødselsdatoen spesifisert i mybirthday.txt og til og med den datoen som er oppgitt med parameteren **date**. Denne parameteren er en datospesifikasjon i tekstformat som brukt i tidligere deloppgaver. Funksjonen skal skrive ut en tekst på følgende format:

Du har opplevd x skuddårsdager.

Her skal x byttes ut med det relevante antallet skuddårsdager. Dersom ikke mybirthdate.txt kan åpnes eller dersom datospesifikasjonen i parameteren **date** er ugyldig, skal det gis en beskrivende feilmelding, og ingenting annet skal skrives ut.

Merk at det ikke er antall skuddår som skal telles, men antall skuddårsdager - det vil si hvor mange ganger skuddårsdagen 29. februar opptrer mellom de to datospesifikasjonene som er gitt ved mybirthdate.txt og parameteren **date**. Skuddårsdager som faller på datoen gitt av mybirthdate.txt eller av parameteren **date** skal også telles med.

For eksempel, dersom fødselsdagen er 13. desember 2001, og **date** angir 9. desember 2019, så skal et kall til **leap_days('13-12-2019')** skrive ut "Du har opplevd 4 skuddårsdager", og det er 2004, 2008, 2012 og 2016.

Skriv ditt svar her...

```
1
```

Maks poeng: 10