



Institutt for datateknikk og informasjonsvitenskap

Eksamensoppgave i TDT4105 Informasjonsteknologi - grunnkurs

Faglig kontakt under eksamen:	Rune Sætre Anders Christensen Nils Inge Rugsveen	Mobil: 452 18103 Mobil: 918 97181 Mobil: 958 76417
--------------------------------------	--	--

Eksamensdato:	2014-12-06
Eksamenstid (fra-til):	09:00 – 13:00
Hjelpemiddelkode/Tillatte hjelpemidler:	Godkjent kalkulator

Annen informasjon:

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 løse oppgavene. 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.

Målform/språk:	Bokmål
Antall sider:	17 (inkl. Forside, svarark og appendiks)

Innhold:

- Oppgave 1: Flervalgsoppgave (25%)
- Oppgave 2: Programmering: Fallskjerm (25%)
- Oppgave 3: Programmering: Værstasjon (30%)
- Oppgave 4: Kodeforståelse (20%)
- Appendiks: Nyttige funksjoner
- Svarark til Flervalgsoppgave (2 eksemplarer)

Kontrollert av:

26/11-2014, Benjamin Bjørnseth

Dato

Sign

Oppgave 1: Flervalgsoppgave (25%)

Bruk de to vedlagte svarskjemaene for å svare på denne oppgaven (ta vare på det ene selv). 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. Hvilken fundamental aktivitet innen programvareutviklingsprosessen fokuserer på å endre programvaren for å møte endrede kunde- og markedskrav?
 - a. Programvarespesifikasjon.
 - b. Programvareutvikling.
 - c. Programvarevalidering.
 - d. Programvareevolusjon.
2. Hva er et analogt signal?
 - a. Et kontinuerlig signal hvor den variable egenskap er gitt av en diskret funksjon, som gir verdier fra et definert og begrenset område.
 - b. Et kontinuerlig signal hvor en variabel egenskap (f.eks. amplitude eller frekvens) representerer informasjonen som overføres.
 - c. Et diskret signal som representeres ved hjelp av nuller og enere.
 - d. En kombinasjon av alternativ a og b.
3. Hvilken type løkkestruktur er garantert å utføre handlingen minst en gang?
 - a. pre-test løkke (pretest loop).
 - b. post-test løkke (posttest loop).
 - c. begge typer.
 - d. ingen av typene.
4. Omtrent hvor mange ganger raskere er en 1 GHz - prosessor i forhold til en på 2 MHz.
 - a. Halvparten så rask.
 - b. Like rask.
 - c. Dobbelt så rask.
 - d. 500 ganger så rask.
5. Hva sier Nyquist-regelen?
 - a. at samplingsrate ved lyd må være minst det dobbelte i forhold til høyeste frekvensen.
 - b. at lyd over 20000Hz ikke kan høres av det menneskelige øret.
 - c. at tapsfri komprimering ikke er mulig for lyd.
 - d. at lyddata tapsfritt kan komprimeres med maksimalt en faktor 2π .
6. Hvilken programvareprosessmodell bør velges for et prosjekt der det skal utvikles et helt nytt system hvor eksisterende komponenter ikke finnes og kunden er usikker på hvordan systemet skal være?
 - a. Vannfallsmodellen.
 - b. Inkrementell utvikling.
 - c. Gjenbruksorientert systemutvikling.
 - d. Havmodellen.
7. Hva er hensikten med et paritetsbit i digitale signaler?
 - a. Forteller hvor meldingen skal sendes.
 - b. Gjør meldingene raskere å overføre (komprimering).
 - c. Bidrar til å detektere feil i digitale signaler.
 - d. Krypterer signaler så overføringen av data blir sikrere.

8. Kompleksiteten til sortering ved innsetting (insertion sort) er
 - a. $\Theta(n)$.
 - b. $\Theta(n \log n)$.
 - c. $\Theta(n^2)$.
 - d. $\Theta(2n)$.
9. En moderne prosessor er typisk bygd opp av mange millioner små...
 - a. Dioder.
 - b. Magneter.
 - c. Transistorer.
 - d. Kondensatorer.
10. En byte med minne i datamaskinen kan lagre hvor mye?
 - a. 16 bits.
 - b. 8 flyttall.
 - c. fire ASCII-tegn.
 - d. en heltallsverdi mellom 0 og 255.
11. Hvilken av de følgende er en kjent fordel med vannfallsmodellen?
 - a. Tar hensyn til brukerkrav som endrer seg i løpet av prosjektet.
 - b. Gjør prosessen synlig og enklere å monitorere for prosjektlederen.
 - c. Får tidlige versjoner av systemet raskt ut til kunden.
 - d. Åpner for kontinuerlig tilbakemelding fra brukerne av systemet.
12. Morsekode representerer bokstaver som sekvenser av prikk og strek som er
 - a. like lange for alle bokstaver i alfabetet.
 - b. kortere for bokstaver tidlig i alfabetet, lenger for bokstaver sist i alfabetet.
 - c. kortere for vokaler, lenger for konsonanter.
 - d. kortere for bokstaver som forekommer hyppig i vanlig tekst, lenger for sjeldnere bokstaver.
13. Hvilken av disse er en korrekt gjengivelse av teoribokas definisjon av en algoritme? "En algoritme er et ordnet sett av..."
 - a. "... entydige, utførbare skritt som definerer en terminerende prosess" (unambiguous, executable steps that defines a terminating process).
 - b. "... entydige, effektive skritt som definerer en utførbar prosess" (unambiguous, efficient steps that defines an executable process).
 - c. "... velformede, effektive skritt som definerer en terminerende prosess" (well-formed, efficient steps that defines a terminating process).
 - d. "... velformede, utførbare skritt som definerer en effektiv prosess" (well-formed, efficient steps that defines an efficient process).
14. En datamaskin går i en uendelig løkke som kalles
 - a. Det naturlige kretsløpet.
 - b. Hent-Utfør kretsløpet.
 - c. Det evige kretsløpet.
 - d. Beregnings-kretsløpet.
15. Hva er korrekt binær representasjon av 'NTNU' i 8 bits ASCII?
 - a. 01001110 01010100 01001110 01010101.
 - b. 01100001 01100100 01110011 01100110.
 - c. 01101110 01110101 01110100 01001110.
 - d. 01100010 01010101 01010010 01010000.

16. I hvilket tilfelle er det mest nyttig å bruke gjenbruksorientert systemutvikling?
- Når det finnes tilgjengelig programvare som kan gjøre oppgaver systemet skal utføre.
 - Når det skal lages programvare for å håndtere resirkulering av søppel eller lignende systemer.
 - Når det skal gjenbrukes ideer fra tidligere prosjekter.
 - Når det skal gjenbrukes systemutviklere og systemdesignere fra tidligere prosjekter.
17. Hva står forkortelsen ISP for?
- Internet Service Provider.
 - Information Security Protocol.
 - Internet Security Protocol.
 - Information Super Pool.
18. Kompleksiteten til binær søk er
- $\Theta(n)$ hvis lista er sortert og $\Theta(n \log n)$ hvis den er usortert.
 - $\Theta(\log n)$ hvis lista er sortert og $\Theta(2 \log n)$ hvis lista er usortert.
 - $\Theta(\log n)$ hvis lista er sortert og $\Theta(n)$ hvis lista er usortert.
 - $\Theta(\log n)$ hvis lista er sortert. Binær søk er ubrukelig hvis lista er usortert.
19. RAM
- Husker alle verdiene når strømmen kuttes.
 - Er alltid inndelt i blokker på 1 kilobyte.
 - Betyr Random Access Memory.
 - Kan trygt fjernes uten at maskinen slutter å fungere.
20. Hva står bokstavene i RGB for?
- Red, Green, Blue.
 - Readable Graphics Byte.
 - Raster Grayscale Balance.
 - Realtime GPU Backlog.
21. Hva kalles aktiviteten som har fokus på å identifisere den overordnede strukturen for et system inkludert dets sub-systemer?
- Hoveddesign.
 - Arkitekturdesign.
 - Interfacedesign.
 - Komponentdesign.
22. MODEM er en forkortelse for
- MODulator / DEModulator.
 - Massive Online Digital Electric Messaging.
 - MONitored Data EMISSION.
 - Mapping Of Digital Electronic Mail.
23. ASCII-kode representerer bokstavene A til Z som sekvenser av 0 og 1 som er
- like lange for disse bokstavene i alfabetet.
 - kortere for bokstaver tidlig i alfabetet, lenger for bokstaver sist i alfabetet.
 - kortere for vokaler, lenger for konsonanter.
 - kortere for bokstaver som forekommer hyppig i vanlig tekst, lenger for sjeldnere bokstaver.

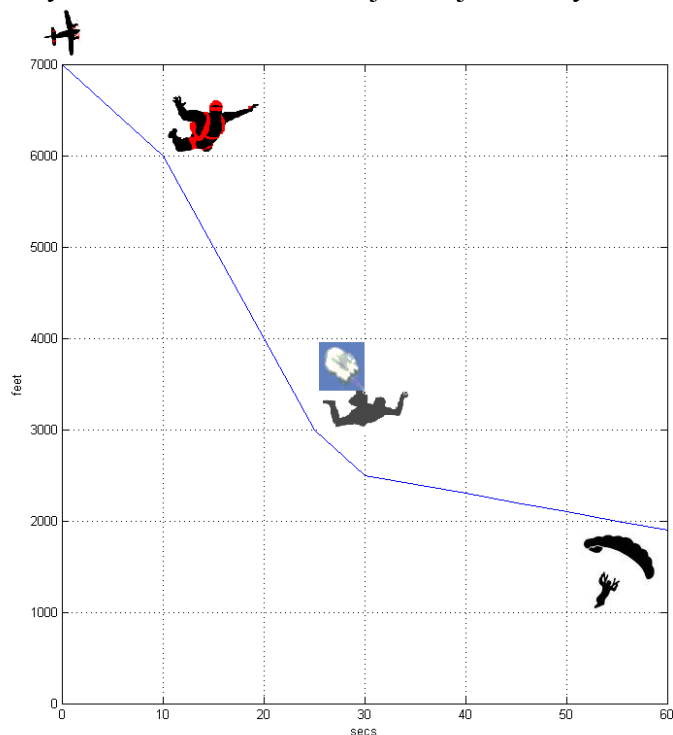
24. Et nettverk som knytter sammen datamaskiner og utstyr i et begrenset område som et kontor, bygning eller i en bolig betegnes med forkortelsen:
- LAN.
 - MAN.
 - PAN.
 - WAN.
25. VPN (Virtual Private Network) kan gi mottageren inntrykk av at en reisende ansatt sin bærbare PC befinner seg innenfor bedriftens nettverk ved at meldinger fra denne PC'en
- plasseres inni en kryptert datapakke for ekstern oversendelse.
 - sendes med en tidsforsinkelse.
 - sendes ekstra hurtig, med høy prioritet.
 - sendes med en falsk avsenderadresse som inneholder et virus.

Oppgave 2 Programmering Fallskjerm (25%)

(I denne oppgaven kan det være gunstig å kalle funksjoner som du har laget i tidligere deloppgaver. Selv om du ikke har fått til den tidligere oppgaven, kan du kalle funksjon derfra med antagelse om at den virker som spesifisert i oppgaveteksten.)

NTNU fallskjermklubb (NTNU-FSK) trenger hjelp til å lage et nytt opplærings- og administrasjonsprogram. De har bedt firmaet ditt (ITGK) om hjelp, og du har fått jobben med å programmere funksjoner som beskrevet i deloppgavene under.

Vi benytter en litt forenklet versjon av jordens fysiske lover:



Figur 1. Hopp fra 7000 fot.

En fallskjermhopper faller (med konstant/gjennomsnittlig hastighet) 100 fot pr. sekund, de 10 første sekundene, og deretter med konstant topphastighet på 200 fot pr. sekund til skjermen må åpnes i 3000 fots høyde (se figur 1). Hvis man mot normalt hopper ut under 3000 fot må skjermen utløses umiddelbart (etter 0 sekunder).

Medlemsdatabasen til NTNU-FSK ligger lagret på en fil 'members.txt', med følgende format:

NAVN;IDNR;VEKT;SKJERMST.

Eksempel på innholdet i filen:

Frank Stank;D-49334;75;120

Bjarne Stor;C-49335;95;150

Dumbo Ear;D-50105;450;750

Peter Pan;A-12345;30;100

Oppgave 2a (5%)

Lag en funksjon `inputPerson` som leser inn navn, id, vekt og skjermstørrelse fra tastaturet, og returnerer en struct `person` med riktige verdier i følgende felt: `name`, `id`, `weight`, `size`. Merk at «name» og «id» er tekststrenger, mens «weight» og «size» er tall. Du kan anta at bruker skriver inn lovlige verdier.

Eksempel på kjøring (tekst med understrekning skrives inn av brukeren):

```
>> person = inputPerson( )
Name: Fredrik Olsen
ID: B-77777
Kg: 80
Size: 240
person =
  name: 'Fredrik Olsen'
  id: 'B-77777'
  weight: 80
  size: 240
>>
```

Oppgave 2b (5%)

Lag en funksjon `readDbFile` som leser inn hele medlemsbasen i en struct-vektor, med følgende feltnavn: «name», «id», «weight» og «size». «name» og «id» skal lagres som tekst, mens «weight» og «size» skal lagres som tall. Du kan anta at filen finnes og at det ikke oppstår noen problemer ved åpning/lukking, og at filen ikke inneholder noen blanke eller ugyldige linjer. Funksjonen skal ha inn-parameter `filename` og retur-verdi `db`.

Eksempel på kjøring:

```
>> db = readDbFile( 'members.txt' )
db = 1x4 struct array with fields:
    name
    id
    weight
    size
>> db(1)
ans =
    name: 'Frank Stank'
    id: 'D-49334'
    weight: 75
    size: 120
>>
```

Oppgave 2c (5%)

Lag en funksjon `printMemberList` som skriver ut innholdet i struct-vektoren `db` (som forklart ovenfor) på skjermen med overskrifter og format som i følgende eksempel på kjøring:

```
>> db = readDbFile('members.txt');
>> printMemberList( db )
      NAVN      ID-NR  VEKT kg.  SKJERMSTØRRELSE
Frank Stank  D-49334   75 kg.   120 kvadratfot
Bjarne Stor  C-49335   95 kg.   150 kvadratfot
  Dumbo Ear  D-50105  450 kg.   750 kvadratfot
  Peter Pan  A-12345   30 kg.   100 kvadratfot
>>
```

Det skal være avsatt 15 tegn til NAVN, 9 tegn til ID-NR, 5 tegn/siffer til VEKT og 4 tegn/siffer til SKJERMSTØRRELSE. Du kan anta at databasen ikke har innhold som går utover de avsatte antall tegn for hvert felt. Feltene skal være høyrejusterte. Hvis verdier trenger mindre plass en antallet avsatte tegn, skal resten av feltbredden fylles av blanke.

Inn-parameter `db`, ingen retur-verdi.

Oppgave 2d (5%)

Lag en funksjon `addPerson` med inn-parametere `filename` (der databasen er lagret). Funksjonen skal be bruker skrive inn informasjon om en person beskrevet med navn, id, vekt og skjermstørrelse, og lagre dette i variabelen `person`. Funksjonen skal så lese inn databasen som ligger lagret i filen `filename` til datastrukturen (struct-vektor `db`). Deretter skal opplysningene om den nye person legges til i `db` og i filen `filename`. Bruk riktig format: se «Eksempel på innholdet i filen» ved figur 1 (se side 6). Returverdi `db` skal inneholde den oppdaterte databasen.

Du trenger ikke å skrive kode for feilhåndtering i forbindelse med å lese eller skrive fra/til fil. Eksempel på kjøring (Endringen er markert med **fet skrift**):

```
>> db = addPerson( 'members.txt' )
Name: Santa Klaus
ID: H-12345
Kg: 155
Size: 380
db =
1x5 struct array with fields:
    name
    id
    weight
    size
>> db(5)
ans =
    name: 'Santa Klaus'
    id: 'H-12345'
    weight: 155
    size: 380
>>
```

Oppgave 2e (5%)

For en fallskjermhopper er det veldig viktig å være klar over hvor mange sekunder man kan vente før man må åpne fallskjermen (Se figur 1). Lag en funksjon `feet2seconds` som regner ut hvor mange sekunder det tar å falle fra et oppgitt antall fot ned til 3000 fot (inn-parameter `feet`, og retur-verdi `seconds`). Bruk informasjon gitt i starten av oppgave 2 (forklaringen til figur 1) til å beregne riktig tid. Hvis antall fot er under 3000 skal funksjonen returnere 0.

Eksempler på bruk:

```
>> feet2seconds( 12500 )
ans = 52.5000
>> feet2seconds( 7000 )
ans = 25
>> feet2seconds( 2000 )
ans = 0
>>
```


Oppgave 3 Programmering Værstasjon (30%)

Du skal behandle data fra en værstasjon for et større antall dager. Dataene ligger lagret som tall i en tabell som heter `weatherData`. Denne tabellen har tre kolonner, én for hver type måledata, og den har en rad for hver dag. De tre typene med måledata er maksimumstemperatur, minimumstemperatur og nedbørmengde. Vi refererer til dagene slik at dataene på første rad er for dag nr 1, dataene på andre rad er for dag nr 2 osv. Eksempel på utskrift av `weatherData` kan være:

```
>> weatherData
ans =
    12.0000    2.4000    8.2000
     6.1000    0.6000   11.9000
     8.3000   -3.5000    0.0000
    11.6000   -5.2000    0.0000
    15.3000    2.8000   14.3000
>>
```

Oppgave 3a (10%)

Skriv en funksjon som du kaller `weatherStats` og som tar variabelen `weatherData` som parameter. Funksjonen skal gå igjennom dataene og utfra det skrive et sammendrag som vist under. Det vil si at den skal skrive ut antall dager i perioden, totalmengden av nedbør i hele perioden, samt at den skal liste den aller laveste og aller høyeste temperaturen sammen med nummeret på dagene for disse temperaturene.

Eksempel på kjøring for dataene vist i den grå boksen ovenfor:

```
>> weatherStats(weatherData)
There are 5 days in the period.
The highest temperature was 15.3C, on day number 5.
The lowest temperature was -5.2C, on day number 4.
There was a total of 34.4mm rain in the period.
>>
```

Oppgave 3b (10%)

Skriv en funksjon `coldestThreeDays` som tar som parameter variabelen `weatherData`. Funksjonen skal finne den perioden av tre sammenhengende dager som hadde den laveste gjennomsnittlige minimumstemperaturen. Den skal returnere nummeret på første dagen i denne tredagersperioden. Dersom det er flere perioder som er like kalde, så skal den returnere kun den siste av disse periodene. Et eksempel på en kjøring av denne funksjonen for `weatherData` som definert tidligere i oppgaven gir:

```
>> coldestThreeDays(weatherData)
ans =
     2
>>
```

Oppgave 3c (10%)

Værstasjonen har nettopp rapportert data for ytterligere en dag. Den kommer som en tekststreng lagret i variabelen `extraData` som har formatet som vist under:

```
>> extraData = 'max=23.5, min=9.3, 5.1mm';  
>>
```

Skriv en funksjon `addNewDay` som tar `extraData` samt `weatherData` som parametere, og som returnerer en ny versjon av `weatherData` som er oppdatert med de nye dataene på slutten av tabellen. Under er vist et eksempel på kjøring (endring er vist i **fet skrift**):

```
>> updated = addNewDay( extraData, weatherData ) ;  
>> updated(end-2:end, :)  
ans =  
  11.6000   -5.2000    0.0000  
  15.3000    2.8000   14.3000  
  23.5000    9.3000    5.1000  
>>
```

Oppgave 4 Kodeforståelse (20%)**Oppgave 4a (5%)**

Hva returneres ved kjøring av funksjonen `myst([1, 2, 3, 3, 2, 1])` med kode som vist under? (3 %)

Forklar med en setning hva funksjonen `myst` gjør? (2 %)

```
function out = myst(A)
    out = 0;
    L = length(A);
    if ( L > 1 )
        B = A(1) * A(L);
        out = B + myst( A(2:L-1) );
    end %if
end %func
```

Oppgave 4b (5%)

Hva blir skrevet ut når man kjører funksjonen `myst_b()` nedenfor? (3 %)

Forklar med en setning hva funksjonen `mystery` gjør? (2 %)

```
function myst_b( )
    disp( mystery(4) )
end
```

```
function out = mystery( W )
    %Creating a Matrix with W x W zeros
    table = zeros( W );
    for a = 1:W
        table(a,a)=1;
        b=0;
        while a-b > 1 && a+b < W
            b = b+1;
            table( a-b, a+b ) = 1;
            table( a+b, a-b ) = 1;
        end
    end
    out = table;
end
```

Oppgave 4c (5%)

Hva returneres ved kjøring av funksjonen `myst_c('RBHOOASDUEØGNGBLSOIURMNGTD')` med kode som vist under? (3 %)

Forklar med en setning hva funksjonen `myst_c` gjør? (2 %)

```
function B = myst_c(A)
    B='';
    for x = 1:3:length(A)
        B = [B A(x)];
    end %for
end %func
```

Oppgave 4d (5%)

I et program som skal trene ungdomsskoleelever i matematikk, trenger vi en funksjon for å sjekke korrekt nøsting av parenteser i uttrykk der tre ulike parentestyper er tillatt. Funksjonen trenger ikke å sjekke at uttrykket ellers er fornuftig, kun at parenteser kommer i lovlig rekkefølge og går opp mtp antall og plassering av alle start- og sluttparenteser. Vi har også skrevet tre setninger som kaller funksjonen for å teste om den virker. Koden er vist her (linjenummer til venstre ikke del av koden men er tatt med så du lettere kan vise til spesifikke kodelinjer i svaret ditt):

```

1 function check_main( )
2     A = check( '{a+4*[b-2*(c+5)]/11}' ) % should be true
3     B = check( '{a+4*[b-2*(c+5)]/11}' ) % should be false
4     C = check( '{a+4*[b-2*(c+5)]/11}}' ) % should be false
5 end %function check_main

6 function out = check( expression )
7     PAREN = '([{}])';
8     parenthesis_list = [];
9     for char = expression
10        if find( char == PAREN(1:3) ) %start-parenthesis found
11            %add corresponding end-parenthesis to the end of the list!
12            parenthesis_list = [ parenthesis_list...
13                                PAREN( find( PAREN == char )+3 ) ];
14        elseif find( char == PAREN(4:6) ) % end-parenthesis found
15            if find( char == parenthesis_list ) % matched start/end-parenthesis
16                parenthesis_list( char == parenthesis_list ) = []; %remove it
17            else
18                out = false;
19                return
20            end
21        end
22    end
23    out = parenthesis_list;
24 end

```

Som kommentarene sier skulle utskriften fra de tre setningene ha blitt hhv true (1), false (0) og false (0). Når vi kjører check_main-funksjonen får vi imidlertid:

```

>> check_main()
A = Empty string: 1-by-0
B = Empty string: 1-by-0
C = 0

```

Det er to feil i koden:

- 1) funksjonen returnerer Empty string: 1-by-0 for A: og B: i stedet for boolske verdier. Hvis denne feilen blir rettet, vil kjøring av programmet indikere den andre feilen:

```

>> check_main()
A = 1
B = 1
C = 0

```

- 2) som man ser over: funksjonen returnerer true (1) for noen uttrykk som skulle gitt false (0), som for linje B over.

Spørsmål: Forklar hvilke kodelinjer som forårsaker feil 1) og feil 2) og hvordan de enklest kan rettes. I begge tilfeller skal det være mulig å rette feilen bare ved å endre noe i eksisterende kodelinjer, det skal ikke være nødvendig å legge til nye kodelinjer.

Appendix: Some useful functions

FIX Round towards zero.

FIX(X) rounds the elements of X to the nearest integers towards zero.

FLOOR Round towards minus infinity.

FLOOR(X) rounds the elements of X to the nearest integers towards minus infinity.

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 FGETS 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.

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 are 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.

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 $x - n \cdot y$ where $n = \text{floor}(x./y)$ if $y \neq 0$.

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.

RANDI(IMAX,M,N) or RANDI(IMAX,[M,N]) returns an M-by-N matrix.

REM Remainder after division.

REM(x,y) is $x - n \cdot y$ where $n = \text{fix}(x./y)$ if $y \neq 0$.

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.

SQRT Square root.

SQRT(X) is the square root of the elements of X.

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.

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.

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, S=['12'; '34'] str2num(S) => [12; 34]

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.

Svarskjema flervalgsoppgave

Kandidatnummer: _____ Program: _____

Fagkode: _____ Dato: _____

Antall sider: _____ Side: _____

<i>Oppgavenr</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>
1.1				
1.2				
1.3				
1.4				
1.5				
1.6				
1.7				
1.8				
1.9				
1.10				
1.11				
1.12				
1.13				
1.14				
1.15				
1.16				
1.17				
1.18				
1.19				
1.20				
1.21				
1.22				
1.23				
1.24				
1.25				

Denne siden er med hensikt blank!

Svarskjema flervalgsoppgave

Kandidatnummer: _____ Program: _____

Fagkode: _____ Dato: _____

Antall sider: _____ Side: _____

<i>Oppgavenr</i>	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>
1.1				
1.2				
1.3				
1.4				
1.5				
1.6				
1.7				
1.8				
1.9				
1.10				
1.11				
1.12				
1.13				
1.14				
1.15				
1.16				
1.17				
1.18				
1.19				
1.20				
1.21				
1.22				
1.23				
1.24				
1.25				