NTNU
Det skapende universitet

Department of Computer and Information Science

Examination paper - TDT4105 Information technology, Introduction

Academic contact during examination:	**Terje Rydland**
Phone:	**95 77 34 63**
Examination date:	**10/12-2013**
Examination time (from-to):	**09:00 – 13:00**
Permitted examination support material:	**Approved calculator**

Other information:

The examination text contains 4 problems. A percentage is given to show how much every problem and sub-problem counts when the examination is graded. Read through all the problems before you start to solve them. Be smart and make good use of your time! If you feel that the problems are not fully specified, please write your assumptions explicitly.

Answer briefly and clearly, and write so the text is easy to read. If your text is ambiguous or longer than necessary, points will be subtracted.

Language:	**English**
Number of pages:	**14 (incl. front page)**

Table of contents:

Checked by:

_____

Date Signature

Problem 1: Multiple-Choice (25 %)

Use the enclosed forms to solve this exercise (take one home). You can get a new form if you need it. **Only one answer is completely correct.** For each question a correct answer counts 1 point. Wrong answer or more than one answer counts -½ point. Mo answer counts 0 points. You will not get less than 0 points total for this problem.

1) We want to represent 1750 different states. How many bits do we have to use?

a) One byte (8 bit).
b) 11 bit.
c) 12 bit.
d) 2 byte (16 bit)

2) A RGB-colour is given by a hexadecimal number. What code represents a dark grey colour?

a) #FFFFFF
b) #404040
c) #506496
d) #300000

3) How much space do you need to store 20 minutes of (uncompressed) stereo sound in CD-quality?

a) Approximately 200 MB
b) Approximately 500 MB
c) Approximately 20 MB
d) Approximately 1 GB

4) Which statement is correct if we assume the IEEE floating-point representation of numbers?

a) The representation consists of 3 parts: Sign bit, mantissa and exponent.
b) The accuracy is affected by the number of bits in the mantissa.
c) The size of the range of numbers that can be represented is affected by the number of bits in the exponent.
d) All the claims (a-c) are correct.

5) A phone number (8 digits) must be stored. Which representation will take the least space?

a) As an integer.
b) As a string of ASCII-characters.
c) As a double.
d) All alternatives (a-c) will take the same amount of space.

6) Each step in the binary search algorithm

a) will reduce the search space to half
b) finds the search key
c) moves one element
d) switches 2 elements

7) Which of the following is a requirement for a list where we want to use the insertion sort algorithm?

a) The list must have an odd number of elements
b) The elements must be sorted
c) There must be ways to remove and add elements to the list
d) None of the mentions requirements are necessary

8) What is the fastest secure way to search for a single value in an unsorted list of numbers?

a) Scan linearly through all the elements in the list until the value is found
b) Sort the numbers and do a binary search
c) Choose a random element from the list until the number is found
d) There is no fastest way

9) Worst case in a linear search algorithm happens when

a) the element is somewhere in the middle of the list
b) the element is not in the list
c) the element is the last element in the list
d) the element is the last element in the list, or is not in the list at all

10) What memory technology is the fastest?

a) DDR-RAM
b) SSD
c) Cache
d) They all have the same speed

11) How does a monitor work?

a) It shows three different colours in each pixel
b) It mixes the colours Red, Yellow and Blue to make all possible colours
c) It regulates the brightness depending on the frequency of the signals from the computer
d) All the alternatives are correct

12) What is correct related to primary and secondary memory?

a) The primary memory is permanent (non-volatile)
b) The secondary memory is often named RAM
c) The primary memory is much larger than the secondary memory
d) None of the alternatives are correct

13) What is correct related to the historic performance improvement of computers

a) The computer becomes faster the closer the transistors in the integrated circuits are
b) The miniaturization makes it possible to increase the clock frequency to above 1 GHz
c) Moore's law says that the number of transistors in a given area is doubled approximately every 2 years
d) All the alternatives are correct

14) Which 5 types of circuits do we find in the processor (CPU)?

a) Instruction-fetch (IF), Inst.-decode (ID), Data-fetch (DF), execute (EX), Result-return (RR)
b) Control Unit, Arithmetic-Logic Unit (ALU), Registers, Input- and Output-circuits
c) None of the alternatives are correct
d) BIOS, ROM, Primary memory (RAM), Secondary memory, Cache

15) What is a protocol?

a) Rules for what a payload in an IP packet can contain
b) A description of how fast a message can be transferred in a packet-switched net like the Internet
c) Rules that decides how the communication is executed and what functions can be used
d) An overview of who is participating in the communication on the Internet

16) Which task does the TCP protocol, that is used on the Internet, have ?

a) Assigning IP address, network mask and default gateway
b) Offer logical connections and multiplexing og these
c) Error correcting codes
d) Parity, CRC or Hash functions

17) If you often find that a service doesn't work when it is needed, it is described as:

a) Bad performance on your connection to the Internet
b) Low availability of the actual service
c) Low confidence of the actual service
d) Unstable or false DNS function

18) What aspects best describes the technical properties of an access technology?

a) Capacity, Marked penetration and price model
b) Protocols, Installation and Terminal equipment
c) Flexibility, Price and Use pattern
d) Quality, Capacity and Efficiency

19) How can you discover if a message has been altered on the way between the sender and the receiver?

a) By using an analogue signature
b) By using IPv6 instead of IPv4
c) By using functions that can be used by the receiver to verify the integrity og the message
d) By using functions to preserve the confidentiality of the message

20) Why is CRC often used to detect errors in digital signals?

a) Because CRC is well suited to discover burst errors
b) Because CRC is better than parity and simple checksum, and as good as hash functions
c) Because CRC is very simple and effective
d) CRC has very good support of standardised protocols.

Problem 2: Basic programming (20%)

In a game of chess the winner gets 1 point, the loser gets 0 points, and a draw (undecided) gives each player ½ points. A chess match is played in a predefined number of games: n. Trondheim Chess Association (TCA) is arranging a match between the grand masters Carl Magnøssen (player no. 1) and Sjakkma Ghandi (player no. 2). TCA needs your help to program a piece of software to administrate the match. Instead of the name of the players the numbers 1 and 2 will be used.

Problem 2a) (6%)

```
procedure chess_match()
      Set total_score1 □ 0 # Total points for player 1
      Set total_score2 □ 0 # Total points for player 2

      Ask the user how many games the match consists of
      Set num_games □ number of games

      If the user gives a number <1 the following text should be printed "Sorry, then there will be no match!"
      Else, as long as there are games left to play:
          Print "Parti" and the number of the game
          Ask the user for the number of points for player 1 in this game
          Set score1 □ number of points for player 1 in this game
          Ask the user for the number of points for player 1 in this game
          Set score2 □ number of points for player 2 in this game
          Set total_score1 □ total_score1 + score1
          Set total_score2 □ total_score2 + score2

      Print "The match is over!"
      Print "Player 1 got" followed by the total points of player 1 and "points."
      Print "Player 2 got" followed by the total points of player 2 and "points."
```

Write the function `chess_match()` described by the following pseudo code:

Problem 2b (3%)

The player that achieves more than half the possible number of points (n/2+0.5 or more points if the match has up to n games) wins the game - the rest of the games do not have to be played. If all *n* games have been played and the two players have the same amount of points, the match ends in a draw, and you will have to play extra games to decide the winner. If the match is up to 12 games, it can end 6-6 with an extra game, or by one of the players achieving 6.5 or 7 points (after 7-12 games).

Write the function

```
end_of_match(num_games, game, total_score1, total_score2)
```

that checks if the match is over, and reports who won. The function will have to check if the total score for one player is high enough for the player to have won the match. The function takes 4 arguments, two integers (`num_games` and `game`) and two floating point numbers (`total_score1` and `total_score2`), and returns 0 if the match is not ended, the number of the player who won (1 or 2) if the match is over, and 3 if it is a draw.

Problem 2c (5%):

Instead of asking the user for the number of points for player 2 in a game, we can use what we know of the points of player 1, and that the points of player 2 are dependent on this.

Write the function

`chess_scorer()`.

The function shall ask the user for the result for one player in a game (1, 0.5 or 0) and return this together with the result for the opponent in the game (also 0, 0.5 or 1). If the user types an improper result, the function should print "Impossible result" and ask again.

Problem 2d (6%):

The program in problem 2a only reports the total number of points for a player, but does not store the results for each game. Instead we want the program to store all the results for one player in a list so that we can get the total points from this list.

Write the function

`player_score(results)`.

The function must accept as an argument a list with results from all played games for one player, and return the players total number of points so far in the match (as a floating point number).
The list in the argument `results` has the same length as the number of games that should be played in the match. The elements in the list can have 4 different values: the three possible results in a chess game (0, 0.5, 1) and the value '`None`' indicating that the game has not yet been played. (Remember that the data type for '`None`' is `string`, and not for instance `double` like the other values in the list).

Problem 3 – Program comprehension (15 %)

Problem 3 a) (5 %)

What values will `a` and `b` have after the assignment: `[a,b] = secret1(11,3)` has been performed?

```
function [r,s] = secret1(a, b):
    r = 0;
    while a >= b
        a = a - b;
        r = r + 1;
    end % while
    s = a;
end % function
```

Explain with one sentence what the function does.

Problem 3 b) (5 %)

What will the value of `answer` be after the assignments shown have been performed?
Explain with one sentence what the function does.

```
>> m = [1:4; 5:8; 9:12; 13:16]
```

```
function m = secret2(m):
    [r,c] = size(m);
    if (r==c)
        for i = 1:r-1
            for j = i+1:c
                temp = m(i,j);
                m(i,j) = m(j,i);
                m(j,i) = temp;
            end % for
        end % for
    else
        m=-1
    end % if
end % function
```

```
>> answer = secret2(m)
```

Problem 3 c) (5 %)

What will the value of `answer` be after the following assignment?

```
>> answer = secret3('148')
```

```
function f = secret3(code):
     L = length(code);
     if L >= 0
         switch code(L)
             case '0'
                 decode = '0000';
             case '1'
                 decode = '0001';
             case '2'
                 decode = '0001';
             case '3'
                 decode = '0011';
             case '4'
                 decode = '0100';
             case '5'
                 decode = '0101';
             case '6'
                 decode = '0110';
             case '7'
                 decode = '0111';
             case '8'
                 decode = '1000';
             case '9'
                 decode = '1001';
             case 'A'
                 decode = '1010';
             case 'B'
                 decode = '1011';
             case 'C'
                 decode = '1100';
             case 'D'
                 decode = '1101';
             case 'E'
                 decode = '1110';
             case 'F'
                 decode = '1111';
             otherwise
                 decode = 'XXXX';
         end % switch
         if L == 1
             f = decode;
         else
             f = [secret3(code(1:L-1)) decode];
         end % if
```

Explain with one sentence what the function does.

Problem 4: More programming (40%)

UKA needs a system to help with the ticket sales. You have volunteered to help. (If you have problems solving one problem, you can still use functions from earlier parts of the problem set as if they were correctly implemented). Commenting the code can be helpful.

Problem 4a (5%)

Write a function, `payment`, that accepts a ticket price and the number of tickets and returns how much the customer should pay. If you have bought more than 3 tickets you will be given a 10% discount on all the tickets.

Problem 4b (5%)

Assume that there exists a text file, `prices.txt`, that contains concert name and the ticket price for the concert. Every concert is stored on one line in the file, with the name of the concert first, and the price of the concert after the name separated with a semi-colon (;).

Write the function, `get_price`, that accepts a concert name and returns the price for this concert. (You must take into account that a concert may not exist!). If the concert does not exist the price 0 will be returned.

Example of file contents:

```
The Rectorats;100
Gloshaugkameratene;150
The aller beste;250
```

Problem 4c (5%)

Write a function, `ticket`, that gets the buyers name, concert name and the number of tickets as arguments. Use the function in 4a to generate the price to be used in the ticket text that this function shall generate. The ticket shall contain the buyer's name, what concert, the number of tickets and the total price. The ticket price for a concert will be fetched from the file `prices.txt` mentioned in problem 4b.

```
**************************************
******
Uka 2015
**************************************
******
Navn:                        Nils Nilsen
Konsert:                   The Rectorats
Antall billetter:                     8
Totalpris:                      720 kr
```

Example of a printed ticket:

Problem 4d (10%)

Write a function, `write_to_file`, that gets its ticket information from the function in problem 4b: (name, concert name and the number of tickets) and saves this to a file (`concerts.txt`). The file shall contain 1 line for each ticket transaction. One line shall contain concert name, the number of tickets, total price and customer name. Each element on a line is separated by a semi-colon (;). The file name will be a parameter to the function. The file shall be updated and not be erased each time it is opened.

Example of file content:

```
The Rectorats;8;720;Nils Nilsen
Gloshaugkameratene;4;540;Per Persen
The Rectorats;2;200;Nina Karlsson
The aller beste;4;900;Even Evenrud
```

Problem 4e (15%)

Write a menu-driven program that that lets you get the following information from the file `concerts.txt`:

- How many tickets there are sold for a given concert
- How large sum each concert has generated
- The total income for the whole arrangement.

Appendiks: Nyttige funksjoner

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.

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.

LENGTH 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.

MOD Modulus after division.

MOD(x,y) is x - n.*y where n = floor(x./y) if y ~= 0.

RAND Uniformly distributed pseudorandom numbers.

R = RAND(N) returns an N-by-N matrix containing pseudorandom values drawn from the standard uniform distribution on the open interval(0,1).

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

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.*y where n = fix(x./y) if y ~= 0.

SIZE 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.

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 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.

Form for Multiple-Choice question

Candidate no: _____ Program: _____

Course code: _____ Date: _____

No. of pages: _____ Page: _____

Problem	***A***	***B***	***C***	***D***
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				

Form for Multiple-Choice question

Candidate no: _____ Program: _____

Course code: _____ Date: _____

No. of pages: _____ Page: _____

Problem	***A***	***B***	***C***	***D***
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				