

Institute for Advancing Intelligence (IAI), TCG-CREST  
Mid-Semestral Examination  
Ph.D Program Session: 2022–2023  
Subject: Introduction to Computer Programming and Data Structures

Date: 18. 04. 2023

Full Marks: 40

Time : 4 Hours

Instructions:

- Try not to answer more than **Three** questions. The maximum you can score is 40.
- Some of the questions require files. They can be downloaded from the digital version of the question paper kept in the *course webpage*.
- For submission, keep the names of the solution files as `MS0x_firstName.c` and send them to `laltu.sardar[at]outlook[dot]com` with subject as “*midsem\_submission\_firstName*” with necessary supporting files.
- Please keep your *roll number* and *name* in the header of each solution file.
- Assume inputs are correct to avoid unnecessary error handling. Language to be used: C.

1. Problem id #MS01:

- Given a word (of maximum 20 characters long) and the path of a file, write a function that outputs 1 if the file contains the word, outputs 0 if it does not contain the word, and outputs -1 if there is any failure to check.
- Suppose you are given a folder/directory containing a set of  $n$  ( $< 500$ ) documents that contains the statements of purpose (SOP). Let `xyz` is the three digit format of  $seq < 500$ . Let the SOP of the candidate having sequence  $seq$  has name of the form `tcgiai23xyz.txt` where. Write a program that prints the list of SOPs (filenames only) which contain the given word. *Your program should not be case sensitive.*
- *Input:* path to the directory as user input from the terminal. Download sample files
- *Output:* Display list of fileNames, each in a new line
- Possible hint: `fopen` returns `NULL`, in case it fails.

2. Problem id #MS02: A game of number.

Suppose Sruti and Nikhil want to play a game of numbers. Given a natural number  $N$  of up to 15 digits, Sruti and Nikhil get  $N$  and  $N + 9$  respectively, in each round, both payers do either the followings (based on user input from the terminal).

- Multiply the digits of the number. If the product becomes a single digit number, then the player declares **end**, else passes the product to the other player.
- Add the digits of the number, If the sum becomes single digit, then the player declares **end**, else, pass the sum to the other player.

In the last played round, if both of them declare **end**, the game is a draw, else the declarer becomes the winner.

Write a C program that finds the winner of the game. Show the status of the game after each round.

Input:  $N$ , up to 15 digit number.

Output: Status of the game for each round until the game is finished.. [15]

3. Problem id #MS03:

A complex square matrix is said to be *Hermitian* if  $A^* = A$  and *skew-Hermitian* if  $A^* = -A$  where  $A^* = \bar{A}^t$ , the complex conjugate transpose of  $A$ . Thus,  $A$  (of order  $n$ ) is *Hermitian* if  $\bar{a}_{ij} = a_{ij}$  and *skew-Hermitian* if  $\bar{a}_{ij} = -a_{ij}$  for  $i = 1, 2, \dots, n; j = 1, 2, \dots, n$ . For a complex number  $a = x + iy$ ,  $\bar{a} = x - iy$ .

It is a fact that any complex square matrix  $A$  can be expressed as a sum of a Hermitian and a skew-Hermitian matrix as:

$$A = S + K, \text{ where, } S = \frac{1}{2}(A + A^*) \text{ and } K = \frac{1}{2}(A - A^*)$$

Given  $A$ , output  $S$  and  $K$ .

- Input: A square matrix kept in the file. Download **sample input** from here. A square matrix of order  $n$  contains  $n$  rows. Each row contains  $2n$  float type number where  $2i$ th and  $(2i + 1)$ th entries are the real and imaginary part of  $i$ th complex number of that row ( $i = 0, 1, \dots, n - 1$ ).
- Output: Two files **symmetric\_part.txt** and **skew\_symmetric\_part.txt** for  $S$  and  $K$  respectively.
- The format of the input/output matrices is as usual where the first line contains number of rows, and second line onward contains the rows of the matrix, each entry separated by a space.

[15]

4. Problem ID # MS04: Consider a special list structure `struct special_list {int n; int k; int *A };` where  $A$  is an integer array of length **always** multiple of  $k$  ( $\geq 2$ ) and  $n$  is the total number of integers stored in that array, from index 0 to index  $n - 1$ .

Write two operations *insert* and *delete* as follows.

- *insert*( $L, x$ ): inserts an integer  $x$  in the List at index  $n$  and update  $n = n + 1$ . If  $L$  is already full, at first extend the array keeping the length multiple of  $k$ .
- *delete*( $L, x$ ): deletes all appearances of an integer  $x$  in the array  $L$ . After deletion, it rearranges the elements in the array. It **reallocates** the memory so that its length remains multiple of  $k$  not keeping more than  $k - 1$  entries empty.

- Input: The number of inputs  $n$  followed by the inputs as

$n$

$op_1 \ val_1$

$op_2 \ val_2$

$\vdots$

$op_n \ val_n$

where  $op_i \in \{+, -\}$  and  $val_i$ s are positive integers.

Here is a **sample files**.

- Output: input values followed by the elements in the list. E.g.,

$op_1 \ val_1 \Rightarrow L[0], L[1], \dots$

$op_2 \ val_2 \Rightarrow L[0], L[1], \dots$

$\vdots$

$op_n \ val_n \Rightarrow L[0], L[1], \dots$

[15]