Lab 08 : Structures, Binary Files and Personal Libraries

I. Overview and Objectives

So far our data structures have been limited to numbers, characters, words, sentences and homogeneous lists of these objects that we called arrays. However there is more to the world we live in than words and lists of numbers.

In programming we must be able to model realities that are more complex, not only numbers and words but also cars, atomic elements, animals, people, trees and planets. Defining new data types that contain multiple components of different basic types is the solution to our current data structure problem.

The learning objective of this lab is to practice working with data structures in C and how to extend our programs by using binary files and personal function libraries.

Reading and related topics: Course slides lessons 10 and 11. Book chapters 10, 11 and 12.

II. Lab Tasks and Submission Guideline

Write complete C programs to solve the following two problems. Save the code you wrote to solve them, together with the result of it in a report. Make sure you include enough comments in your code.

For each problem, copy/paste the source code (in text, not image) and copy/ paste the execution results of each of the outputs (high-resolution screenshots) into your lab report. Save your report in .pdf format and submit it on D2L. You should submit your lab at the end of your lab session or soon after. In all cases it must be submitted before the deadline indicated in the D2L dropbox or it will not be accepted for marking.

Problem 1: Numeric addresses for computers using the IPv4 format are composed of four parts, separated by periods in the form of `aa.bb.cc.dd` where aa, bb, cc, and dd are integers larger or equal to zero. Locally, computers are usually known by a nickname as well.

Write a C program to process a list of IPv4 addresses identifying all pairs of computers from the same locality. Create a new structure type named **address_t** with components for the four integers of the IP address and a fifth component in which to store an associated nickname of up to 15 characters.

Your program should read a list of addresses and nicknames from a file (maximum 300 addresses) terminated by a sentinel IP address of all zeros and a sentinel nickname (sample file here). It should then print a list of messages identifying each pair of computers from the same locality, that is, each pair of computers with matching values in the first two components of the address. In the messages, the computers must be identified by their nicknames.

Ex: *Servers mirkwood and gandalf are on the same local network.*

For this purpose, you must write a function named **localnet** that takes two IP structured variables and returns true (1) if they are on the same local network and false (0) if not.

Follow the messages by printing the full list of addresses and nicknames.

Problem 2: A data file named (sample file here) contains 100 real numbers. Read all 100 numbers into a two-dimensional array of 10 rows and 10 columns.

Write a personal library of functions named **mylibrary.h** containing the following functions:

sumdiag: sums all the numbers in the main diagonal of the array ([0][0] to [9][9]).

sumall: sums all the numbers in the array.

avright: calculates the average of the last (rightmost) column of the array.

corners: sums the four corners of the array.

largeanti: returns the largest number found in the antidiagonal ([0][9] to [9][0]) of the array.

In the main program, read the file into the array, call the five functions and write the five numbers that are the result of the five functions into a binary file named **results.bin**.

Finally, read the five numbers from the binary file and display the results in sentences with one decimal precision.

Ex: *The sum of the four corners is 0.0.*

Have fun!