Computer Science & Engineering 171
Assignment #1: Imperative Programming

Due: April 23rd at 6:00 pm

The goal of this assignment is to introduce you to writing code in two imperative languages designed for different purposes. Use the Rocky 9 machines in the Engineering Computing Center for this assignment. The Pascal compiler is `fpc` and the C compiler is `gcc`.

The practicum will be done in class in teams of two or three. Your score will be based on your participation. The homework is to be done individually and submitted using Camino. You score will be based on correctness and coding style. The practicum and the homework are each worth 5 points.

1 Practicum

1.1 Quicksort

On Camino you will find an implementation in Pascal of ***quicksort***, an efficient sorting algorithm. Translate the program to C, keeping the same procedure and function names and parameters, for all but the top-level program, which will become the function `main` in C.

Goal: To learn about methods of passing parameters and recursion.

Hints: None.

1.2 Binary Search Trees

A ***binary search tree*** is either empty, or it consists of a node with two binary search trees as subtrees. Each node holds an integer. The elements in a binary search tree are arranged so that smaller elements appear in the left subtree of a node and larger elements appear in the right subtree. On Camino, you fill find an implementation of a binary search tree in C. Translate the program to Pascal, keeping the same function names and parameters, for all but the function `main`, which will become the top-level program in Pascal.

Goal: To learn about types and data representation.

Hints: You will need to use a record or structure to represent a tree node, and tree nodes need to be dynamically allocated. It is easiest to have both functions be recursive.

1.3 Coding Guidelines

You should not mimic Pascal code in C, nor C code in Pascal, but rather use the best mechanisms provided by each:

- C does not have a boolean type so `int` is typically used instead; Pascal has the built-in type `boolean`.
- Arrays in C always begin at index zero, and although arrays in Pascal can be begin at any index, they traditionally begin at index one.
- C traditionally uses a `#define` for a constant, whereas Pascal uses `const`.

- Since C does not support call-by-reference, a common idiom used to modify a parameter is to pass the parameter to the function by value and return the updated value; Pascal natively supports call-by-reference through `var` parameters, which should only be used for this purpose or for reasons of efficiency. Note that C passes arrays as pointers, but Pascal allows arrays to be passed by value though they typically should be passed by reference for efficiency.
- Pascal allows functions and procedures to be defined within other functions or procedures, thereby limiting their visibility; C does not allow such definitions, but does allow a function to be visible only within the file by using the keyword `static`.
- You should use `typedef` to give a name to an array type of a specified size, allowing the name to be used in multiple places.

2 Homework

Build upon the provided solutions to the practicum (and not your own solutions) when solving the homework.

2.1 Quicksort

Modify the quicksort function to accept a comparison function as an additional parameter and use the comparison function when comparing two values in the array. The comparison function requires two arguments and returns an integer greater than, equal to, or less than zero depending if the first argument is greater than, equal to, or less than the second argument. Provide your own comparison function to sort the values in the array in ascending order. You will find it easiest to use a type declaration for the comparison function:

- Use a `typedef` in C: `typedef int (*comparator)(int x, int y);`

Call this program `sort.c` and submit it using Camino.

2.2 Binary Search Trees

Extend the binary search tree programs by implementing an operation to deallocate all nodes in a tree. Define a function in C and a procedure in Pascal. Furthermore, extend your programs with an additional operation to compute and return the maximum value in the tree:

- Define a function in C: `int maximum(tree root);`
- Define a function in Pascal: `function maximum (root : tree) : integer;`

After the user terminates searching for values by entering `-1`, print the maximum value and then deallocate the tree. Call these programs `tree.c` and `tree.p` and submit them using Camino.