CS44002 Learning Journal 4

The Learning Journal is a tool for self-reflection on the learning process. In addition to completing directed tasks, you should use the Learning Journal to document your activities, record problems you may have encountered and to draft answers for Discussion Forums and Assignments. The Learning Journal should be updated regularly (on a weekly basis), as the learning journals will be assessed by your instructor as part of your Final Grade.

Your learning journal entry must be a reflective statement that considers the following questions:

* Describe what you did. This does not mean that you copy and paste from what you have posted or the assignments you have prepared. You need to describe what you did and how you did it.
* Describe your reactions to what you did
* Describe any feedback you received or any specific interactions you had. Discuss how they were helpful
* Describe your feelings and attitudes
* Describe what you learned

Another set of questions to consider in your learning journal statement include:

* What surprised me or caused me to wonder?
* What happened that felt particularly challenging? Why was it challenging to me?
* What skills and knowledge do I recognize that I am gaining?
* What am I realizing about myself as a learner?
* In what ways am I able to apply the ideas and concepts gained to my own experience?

Finally, describe one important thing that you are thinking about in relation to the activity.

Answer all questions that apply, however, you are expected to complete the Learning Journal assignment in 400-600 words.

During this week of learning, I have gained exposure to recursion and the semantics of flow control statements and subprograms.

I also like to explore the question of minimum spanning trees. The minimum spanning tree is a subset of the edges of a connected, edge-weighted graph that connects all the vertices without any cycles. All the weights of the edges of the tree must be distinct. This is one of the common problems for economic calculation like building the road with the least repetition.

This can be found through the greedy algorithm. This can be solved with the Kruskal algorithm.

The steps are as below:

Step 1 sort all edges with ascending order of the edge weights.

Step 2 pick the smallest edge.

Step 3 check if the new edge creates a cycle or loop in the spanning tree

Step 4 if this did not form a cycle, then integrate the edge into the minimum tree.

Step 5 Repeat step 2 until it includes V-1 edges.

I learned the basis of the switch/case implementation.

The break needs to be explicitly declared else will fall to the next case. I also got a basic understanding of the empirical formula of if-statements.

I got to know the assembly formula of while loops. The side effects of the loop should be avoided for some of the below reasons.

Side effects make the loop difficult to fully verify and test. Side effects adversely affect the readability and maintainability of the program.

The subprogram consists of a declaration which defines the interface of the subprogram. It contains the name and parameters.

Local declaration which is accessible only within the body of the subprogram.

And a sequence of executable statements.

The formal parameter is a declaration that appears in the declaration of the subprogram.

The computation in the body of the program is written in terms of formal parameters.

An actual parameter is a value that the calling program sends to the subprogram.

Besides that, I learned the unsafe cause of using call-by-reference.

A block is an entity of declaration and executable statements. Blocks in general and procedures in particular can be nested within one another.

Each declaration has associated with it three properties.

They are scope, visibility, and lifetime. The advantage of block structure is that it provides an easy and efficient method of decomposing a procedure.

I learned the nature of recursion implementation.

The compiler must emit pure code. The code must not modify itself.

During the running time, it must be possible to allocate an arbitrary number of memory cells for the parameters and local variables.

Then I learned the stack structures which process a property that is last-in, first-out.

The stack contains the special data on top of the stack pointer.

Overall, I felt more confident in understanding some recursion-based algorithms and their implementations. To understand all the details of the empirical formula of each paradigm.