Csc344 Problem Set: Memory Management / Perspectives on Rust

Preliminary Remarks

One way to think about approaches to learning is to imagine yourself in a direct learning, guided learning, or self learning scenario. All are good, and you should do your best to be comfortable with, to benefit from, all three approaches. So far, this course has been skewed towards the direct and guided modes of learning. This problem set, on the other hand, while providing some guidance, is decidedly oriented towards the self learning mode. Thus, you, along with whatever resources you can find, are expected to make some sense of a selection of ideas, and to come out of the experience knowing a little bit more about the topics than when you entered into it.

The ideas that you are being asked to explore in this problem set involve (1) basics of memory management, and (2) bits of the Rust programming language. This is not intended to be an in-depth investigation of these topics. Rather, it is intended merely to acquaint you with some basic ideas pertaining to the run time management of memory, and to afford you an opportunity to take a superficial look at the Rust programming language. It turns out that these two topics, memory management and Rust programming, possess some interesting points of intersection.

Big picture, please think of this problem set as an opportunity to consolidate some knowledge about memory management in the context of programming language design and the practice of programming, and an invitation to learn to program in Rust some day.

Task 1 - The Runtime Stack and the Heap

Just as there is a fairly standard model of human memory, one with three component systems (sensory, short term, long term), each possessing its own characteristics in terms of capacity and duration and storage/retrieval considerations, there is a fairly standard model of run time memory associated with computer programs. This fairly standard model involves the runtime stack and the heap. This tasks invites you to acquaint yourself, or further acquaint yourself, with these component systems of a program's memory, and to write a little something about them.

There are any number of materials that you can find either online or offline pertaining to the run time stack and the heap. I will suggest just one very modest resource (a broken English resource), that casually describes the concepts in passing, as it presents a bit of c/c++ context and a bit of exposure to Rust's memory management system. I will leave it to you to find a small number of other resources on the topic that resonate with you. My suggestion:

`https://medium.com/@rabin_gaire/memory-management-rust-cf65c8465570`

When you feel prepared, please write a three paragraph essay with the title "The Runtime Stack and the Heap" that refines the following three paragraph stubs:

<<Paragraph 1: Introductory paragraph that indicates what the following two paragraphs are going to talk about, and why the discussion is of some significance.>>

<<Paragraph 2: Conceptual description of the runtime stack.>>

<<Paragraph 3: Conceptual description of the heap.>>

Task 2 - Explicit Memory Allocation/Deallocation vs Garbage Collection

If explicit memory allocation/deallocation ideas and garbage collection are new to you, please find some basic resources that will serve to acquaint you with the two approaches to dealing with memory management. (The item suggested previously should certainly help to orient you to these ideas.) Once you feel ready, please write a three paragraph essay titled "Explicit memory allocation/deallocation vs Garbage Collection" that refines the following three paragraph stubs:

<<Paragraph 1: Introductory paragraph that indicates what the following two paragraphs are going to talk about, and why the discussion is of some significance.>>

<<Paragraph 2: Conceptual description of explicit allocation/deallocation of memory, along with the mention of at least one well-known programming language that requires the programmer to engage in the explicitly allocation and deallocation of memory.>>

<<Paragraph 3: Conceptual description of garbage collection, including an abstract description of how a system collects garbage, along with the mention of at least one well-known programming language that performs garbage collection.>>

Task 3 - Rust: Memory Management

Please read through the following blog entry, `https://mmhaskell.com/rust/memory`, taken from the "Monday Morning Haskell" blog, with an eye towards finding what you believe are the most salient bits of knowledge presented in the piece. Then, write down 10 "salient sentence sequences" that you mine from the blog entry, where a **salient sentence sequence** is 1-4 contiguous sentences that you find particularly informative or important or interesting, with respect to the topic of this problem set – namely, memory management.

Task 4 - Paper Review: Secure PL Adoption and Rust

Without feeling any obligation at all to read the entire thing, please spend a little meaningful time with the following paper:

`https://obj.umiacs.umd.edu/securitypapers/Rust_as_a_Case_Study.pdf`

Then, write a **three paragraph review** of the paper that you think might resonate with a senior computer science major who is about to graduate, and who is about to commence a search for an entry level software developer position.

Task 5 - Document creation

Please craft a document to present your work on the preceding tasks, **a document that is consistent with the accompanying template**. Save your document **in PDF form**, and post it to your web work site.

Due Date

Friday, May 12, 2022