CSC344 Problem Set: Memory Management / Perspective on Rust

Task 1 - The Runtime Stack and the Heap

Rust is a statistically and strongly typed programming language.It works with the process of compiling into a binary executable file and can be runned by people who don’t even have rust installed. It differs from programming languages like python, Ruby,javascript which needs interpreters on your local machine to run a code. Rust is very efficient and reliable because it's excellent at handling and identifying bugs in a code. It also has a lot of high level features and gives the programmer control of low level features which makes it likable compared to languages like c or c++. Rust allows people to make command-line tools, web developments,embedded devices,machine-learning,search engines etc.Rust has no runtime type and has two places to store memory which consist of stack and heap.

Runtime stack is the fastest section of RAM (Random Access Memory) in rust and works similarly to data structure and algorithm concepts. The stack follows the “Last in first out”(LIFO) rule which stores data in the order it was received. Stack in rust distributes' as a matter of

course which translates to the stack pushing memories that are relevant to the expected to generate.stacks also automatically deletes unnecessary data in the memory.

The concept of heap in rust is very different from the heap of data structures and algorithms.The heap is used for dynamically sized objects.The heap is slower in the process of allocating memory,however it has unlimited memory size to execute programs.The heap takes advantage of reference which is also known as a pointer to allocate where memory is.In some programming languages, the heap uses a new function to request memory allocation.

Task 2 - Explicit Memory Allocation/Deallocation vs Garbage Collection

Rust has no explicit memory allocation or deallocation in the process of safe programming.Unlike languages like C and C++ that require running through multiple processes to keep programs safe from the interference of unwanted memories since it lacks a garbage collector. Rust’s compiler tracks the timeline of each code generated to incorporate the memory wiping process right when the program stops.

Explicit memory allocation in Rust allows for enhanced control over memory. Languages like C and C++ use malloc to dispense memory on a load, programmers acquire a ton of low-level memory the executives control however in the event that the designer isn't cautious enough then prompts a ton of bugs like twofold free, hanging pointers, and memory spills.

The garbage collector utilizes its compiler and an interpreter to identify the memory area that is presently not being used and opens up the memory. Haskell and Python both use garbage collection to deallocate unused memory.

Task 3 - Rust: Memory Management

1. When compared to other programs like Haskell, Rust does not have a typical garbage collector. It automatically gets rid of memory that prevents the program from running smoothly.
2. C++ and Rust give more control over memory management. However, C++ uses heaps differently. It uses it for storing and deleting memories.
3. Rust's heap memory has the privilege of singular ownership.

4. Variables in Rust cannot be accessed anymore when block programs finish. It’s quite the opposite in the Python programming language. Primitive types in Rust can also be copied since their sizes are rigid.
5. StringTypes are considered non-primitive and can be appended as a result of memory being stored on the heap.
6. In the case of scoop strings, Rust cleans memory from heap and does so by declaring a native object function called "drop", in contrast to C++, which uses the delete or new functions to allocate memory.
7. Rust does not use deep copies because they are expensive

8.Rust utilizes the borrowing reference method rather than sharing ownership with other functions which is similar to C++. It uses the operator to perform this sharing procedure.

9.Rust has mutable references that prevent errors from occurring in programs.

10. Slices in Rust are described as primitive data stored up in the stack and slices also nurture the idea of referencing.

Task 4- Paper Review: Secure PL Adoption and Rust

Rust has many beneficial qualities that make programming easier and faster. According to the surveys taken by participants in the paper review Rust is categorized to have a high quality documentation system. It was also good with giving details or simplification to error messages to allow the reader to comprehend easily what's wrong with their program.

Although Rust has its perks, it also needs room for improvement. Some setbacks that were highlighted by users from the survey includes, adjusting to security guarantees, limited library assistance, slow compiling process,and the worry for instability of the program which can spark the concern for programmers who utilize Rust often due to the possibility of not getting hired.

There's been suggesting ways to improve programmers' experience with using Rust. For example, to avoid security errors, the process of borrowing reference ownerships implemented in the program. Also Even without a garbage collection, Rust has systems in place which perform just like ast garbage collector but much simpler. Overall Rust programming language is an amazing programming tool to execute meaningful programs.