Entrega no.4: Estructura de Datos

Una estructura de datos puede ser definida como una forma de organizar y gestionar los datos de memoria del programa de manera eficiente; estas estructuras contienen reglas propias para almacenar, acceder, agregar y eliminar los datos ingresados previo o durante la ejecución del programa. El tipo de estructura a utilizar dependerá de la situación o problemática que se este trabajando, pero se puede tomar de referente la optimización del recurso del tiempo y memoria del programa verificada mediante la notación Big O, aunque el tipo de dato con el que se esta trabajando puede ser un factor de influencia en esta elección. Algunos de los ejemplos mas conocidos son: arreglo, pilas, colas, listas enlazadas o árboles. Cada programa cuenta con metodologías propias para llamar la misma estructura de datos, como el caso de Java y Rust, dándonos comparaciones de este tipo:

Java	Rust
Int String	i32, 5
Arraylist	Vec(T)
Queque	VecQue

Para la creación del programa Concesionaria.rs, se utiliza la estructura box y cons para definir los listados dentro de otro listado y evitar el problema de potencial de crecimiento infinito de dichos listados para el funcionamiento correcto del programa.

La estructura box puede ser definida como un puntero inteligente para dirigir el almacenamiento para el heap. Dentro de la lógica de Rust se necesita definir el espacio que ocupara cada dato almacenado dentro del stack durante el tiempo de compilación, al no cumplirse este requerimiento el programa presentara el error E0072 (el dato recursivo tiene un tamaño infinito) reusándose a funcionar. Las estructuras con crecimiento indefinido como los vectores tienen un sistema de traslado de datos ya implementado para enviar los datos al heap y para definir un rango de espacio dentro del stack, sin embargo, para estructuras “indirectas” para Rust, como el caso de los cons, este traslado implementado no se presenta. Para resolver la falta de implementación de traslado se utiliza la estructura box, dicha estructura crea una referencia de los datos de crecimiento infinito que son traslados al heap, dicha referencia que ya tiene un valor definido para el stack (de 4 a 8 bytes) al momento de copilar, arreglando el problema E0072 de recursividad. (Tapia, 2022)

La estructura cons es una estructura inspirada de lenguajes funcionales, como es el caso de Lisp. Esta estructura es la conexión lineal de nodos, dichos nodos están conformados por valores y referencia al siguiente elemento de la lista (cabeza, cola). Con esta estructura combinada con un enum se puede presentar un listado de crecimiento de tamaño infinito. Este crecimiento de tamaño infinito se da ya que se hace referencia a un valor de tipo lista que puede hacer referencia a otra lista

y así hasta llegar a un loop infinito de estructura de almacenamiento. Con la ayuda de la estructura box hace que desaparezca este crecimiento infinito para el stack pero no para el heap, cumpliendo con los requisitos del stack al momento de copilar el programa. (Hermo, 2023)

Las principales ventajas que se dan con estas estructuras en Rust comparadas con otros idomas son; no tener la necesidad de un recolector de basura y rendimiento predecible al momento de crear estructuras de crecimiento dinámico. Rust no necesita de un recolector de basura por su sistema de pertenencia y prestamos entre las variables del programa, esto elimina el costo de corrida del recolector de basura y reduce las pausas del recolector de basura; a diferencia de programas como Java, Ruby, Python o javascript. Respecto al modelo de crecimiento predecible hace referencia a tiempos controlados y definidos en la sección de stack del programa gracias a la implementación de box, como es el caso en C++, Haskell o Swift. (Escamila, 2021)

Bibliografia:

- Escamilla-Ochoa, V. R. (2021). Biblioteca de evaluación y entrenamiento de redes neuronales artificiales implementada en Rust y OpenCL.
- Hermo González, J. (2023). Librería de Estructuras de Datos Compactas en Rust.
- Tapia Toapanta, L. E. (2022). *Implementación de un módulo didáctico de energía solar fotovoltaica para el LTI-ESFOT* (Bachelor's thesis, Quito: EPN, 2022).