Compiladores: Práctica 1

Luis Eduardo Martínez Hernández
314240028
Emmanuel Cruz Hernández
314272588
Daniela Susana Vega Monroy
314582915

19 de Agosto de 2019

1 Desarrollo de la práctica

Durante la implementación de esta práctica revisamos la documentación de Racket para recordar ciertas funciones que nos servirían como apoyo para resolver los problemas. Esta investigación fue desde el uso de cond, comparaciones, creación de estructuras, el concepto de recursividad, listas, el uso de lambdas, hasta funciones más complejas como let* y letrec.

Es importante mencionar que para el ejercicio 2, hicimos uso de una biblioteca de Racket llamada gregor. Esta biblioteca nos permitió hacer uso de las fechas de una forma más cómoda y rápida, ya que las operaciones que nos ofrece nos permitió resolver el problema sin uso de funciones auxiliares. Lo que nos recuerda que es importante conocer la documentación de un lenguaje para saber todas las herramientas que nos ofrece y así, evitar la repetición de código que ya está hecho, además de ser, muy probablemente, más eficiente a comparación de un código creado desde cero por nosotros.

Mientras creabamos el código para esta práctica, fuimos dividiendo cada problema en subproblemas (reflejado cada subproblema en nuestras funciones auxiliares). Tal es el caso del ejercicio 1, en el cual, creamos una función por cada necesidad del problema para revolverlo, es decir, el resultado de una función fue la entrada de otra función, hasta que al final obtuvimos el resultado esperado, la solución del problema.

2 Comentarios

- **Ejercicio 1**

  El problema más grande al que nos enfrentamos en este ejercicio fue que no se nos ocurrió una forma muy directa de obtener el resultado del problema. Por lo que hicimos uso de varias funciones auxiliares, tales como fractalAux, rellena y contiene?. Por lo que era un poco confuso seguir el camino una vez terminada cada una de las funciones auxiliares.

- **Ejercicio 2**

  Se nos complicó un poco el ejercicio, pues tratamos de hacer funciones para todo. Al principio con ayuda de la biblioteca date de racket, para indicar cuándo es bisiesto, cuáles meses tienen 30 días, etc. Sin embargo, una vez que supimos acerca de la biblioteca Gregor todo fue muy rápido.

- **Ejercicio 3**

  Se nos complico el ejercicio 3, en específico el inciso b porque no recordábamos cómo hacer recursión de cola y cómo usar las funciones básicas de Racket (map,filter,fold), pero una vez que leímos la documentación y vimos ejemplos pudimos resolver el ejercicio.

- **Ejercicio 4**

  En este ejercicio, usamos por completo las ventajas que tiene usar las listas y las funciones de Racket. Ya que para saber si un número es palíndromo, dividimos el número en cada uno de sus dígitos con ayuda de la función quotient, ya definida por el lemguaje, guardando cada dígito en una lista, para así comparar está con su reversa (por cierto, también definida por el lenguaje: reverse), y así saber si un número es palíndromo.

- **Ejercicio Extra**

  En lo personal, este ejercicio nos pareció más sencillo que otros de la práctica (el ejercicio 3 por ejemplo), ya que en cursos pasados vimos demasiado sobre la implementación de Fibonacci con recursión y recursión de cola.