Tarea 1b.

Martin Felipe Espinal Cruces *

13 de febrero de 2019

1. Objetivo

"Identificar las ventajas y despentajas en la resolución de problemas en paradigmas distintos"

2. Planteamiento del problema

- ¿Qué carácteristicas de las vistas en clase tienen los lenguajes (y el paradigma) usados para resolver los ejercicios propuestos?
  - Respecto al lenguaje de paradigma funcional se encuentran:
    - Al utilizar un lenguaje como haskell es claro que es un lenguaje de alto nivel(más cercano a nuestro lenguaje).
    - Es un lenguaje interpretado.
    - Utiliza claramente la recursión para la resolucion de los problemas,
    - Su evaluzción es perzosa ya que evalua las funciones sólo cuando son llamadas.
    - La utilización de la recursión como herramienta fundamental para la realización de las distintas funciones.
  - Respecto al lenguaje de paradigma iteratívo se encuentran;
    - Al utilizar el leguaje c++ utilizo un lenguaje compilado, que verifica los errores antes de ser ejecutado.
    - Se encuentra en la categoría de lenguajes iterativos.
    - Las estructuras de control en especial el ciclo while para realizar diversas acciones para cumplir con las especificaciones de los ejercicios.

* cofy43@ciencias.unam.mx

        - Al ser un lenguaje de bajo nivel, debido a que las intrucciones que se encuentran en el código funete son claramente mas parecidas al lenguaje que maneja la máquina.

- Justificación:
    - La razón por la que elegí el lenguaje de c++ fue para poder intentar interactuar un poco con los apuntadores que es una caracterítica principal del lenguje. Se puede observar esta implementación en el método que realiza la descomposición en factores prinmos, dado que en el lenguaje no está permitido la devolución de un arreglo, utilizo un apuntador a un arreglo de tipo entero para poder devolverlo.
- ¿Qué ventajas y desventajas tiene implementar los ejercicios anteriores con respecto al paradigma y al lenguaje de programación elegidos?
    - ventajas
        - Paradigma funcional
            - La sintaxis fue sumamente amigable. Fue en general para todas las funciones.
            - Pocas líneas de código. Fue en general para todas las funciones.
            - Recusión, muy útil para la realización del método que factorizaba un número debido a que bastaba con una lista por comprensión para poder realizarla.
        - Paradigma Iterativo
            - Utilización razonable de memoria para cada método
            - Alta cohesion entre las funciones mcd y mcm.
            - El uso del estado para obtener el mínimo común múltiplo y el mínimo común divisor a traves de sólo una operación aritmética.
    - Desventajas
        - Paradigma funcional
            - El uso exagerado en el stack para la llamada recuriva en las funciones de mínimo común multiplo y máximo común divisor.
            - El uso de método auxiliar en la sucesión de Collatz.
        - Paradigma iterativo
            - Lamentable uso de memoria para almacenar un arreglo suficientemente grande para poder almacenar los factores primos de un número en un arreglo
            - Uso plentamente necesario del ciclo while para obener la sucesión de Collatz y la división en factores primos.

- En que escenario conviene útilizar los paradigmas que elegiste para cada caso? Considera las ventajas y características que expusiste en las preguntas anteriores.
  - Obtener el mínimo común múltiplo: iterativo.
  - Obtener el máximo común divisor: iterativo.
  - Dado un número natural N, devolver verdadero si la sucesión de Collatz de ese número termina en el dígito 1: Funcional.
  - Devolver una lista o arreglo con la descomposición de factores primos de un natural N: Funcional

3. Desarrollo

"Todo comienza con la elección de los paradigmas de programción, los cuales fueron Funcional e iterativo. Posteriormente en la elección de los lenguajes, Haskell y c++. Finalmente La implementación de las funciones y métodos para cada lenguajes sobre cada ejercicio."

4. Solución

- Problema 1:
  - Paradigma funcional
    - Para resolver esta función necesite de 2 funciones auxiliares "minimoComun"que dadas dos listas de enteros devolvía el número menor que se encontraba en amabas listas; y "multilos"que dado un número, genera una lista con todos los números que lo dividen. Al utilizar estas funciones en la llamada recursiva, obtenía el mínimo común múltiplo.
  - Paradigma iterativo
    - En este paradigma útilizo el método máximo común divisor, con una opreción aritmética que consideraba el número mayor y lo dividía entre el máximo común divisor de ambos números y el resultado lo múltiplicaba por el número menor.
- Problema 2:
  - Paradigma funcional
    - Para realizar esta función necesite de dos funciones auxiliares y una lista pro comprensión, la primera era "divisoresPropios.[el] cual utilizaba la función "divisores"para regresar una lista de los divisores incluyendo el número con el que se invocaba. La primera función auxiliar es çomun"que dado un par de listas devuelve una lista de elementos en común de ambas listas, la segunda función es "maximun"nativa del lenguaje que devuelve

el elemento mayor de una lista. Finalmente la concatenación de las funciones resultaba en el máximo común divisor.

- Paradigma iterativo
  - Similar que en el método para obtener el mínimo común múltiplo utilizo dos variables auxiliares para obtener el elemento mayor y menor del imput, teniendo esto creo una variable para almacenar el resultado, y el proceso lo realizo mediante un ciclo while donde la condición es que el elemento mínimo sea distinto a cero, cumpliendo esa condición actualizo el valor de la variable del resultado al valor mínimo, actualizo el valor del número menor en el residuo de la división del número mayor sobre el número menor, teniendo eso actualizo el valor del número mayor al del resultado (el que en un principio fue el número mayor) cuando la condición deja de cumplirce regreso el mínimo común múltiplo.

- Problema 3:
  - Paradigma funcional
    - Para esta función requiero de una función auxiliar llamada "sucesíon"que sirve para devolver una lista con las condiciones de la conjetura de Collatz. Finalmente en la función Çollatz"verifico si el último elemento de la lista generada es uno, en cuyo caso devuelvo el valor booleano verdadero, en caso contrario falso.
  - Paradigma iterativo
    - En este caso utilizo un ciclo while para comenzar a reducir el número según las condiciones planteadas, y finalmente devuelvo el valor de la sentencia "Si el número resultante del ciclo es uno".

- Problema 4:
  - Paradigma funcional
    - Para la esta función fue necesaría una función auxiliar llamada "primos"que devuelve un booleano, resultante de la verificación de saber si un número es primo. Finalmente realizo una lista por comprención que recibe una lista desde 2 hasta n (dado que el número 1 dívide a todos los número y no es primo) y la condicion resultante de la función "primos".
  - Paradigma iterativo
    - Dado la escencia de l lenguaje son los iteradores, use esta característica para la realización de este punto, dado un arreglo de tamaño constante utiliza 2 variables principales, el primero çantidad"que representa el tamaño real donde se encuentran

alamcenados los factores primos, para evitar recorrer inceserariamente el resto del arreglo, y la segunda ï”que representará el número primo que factoriza el número del imput, finalmente dentro de un cliclo hago la verificación de si el número ï”divide al imput, en tal caso se alamcena en el arreglo, cuando la condición de que el imput se vuelva 1 devuelve el arreglo.