INSTITUTO TECNOLÓGICO DE NUEVO LEÓN

INGENIERÍA EN SISTEMAS COMPUTACIONALES

LENGUAJES Y AUTÓMATAS II

UNIDAD 1 ÁRBOL SEMÁNTICO

Proyecto 1 Árbol de expresiones

ALUMNO: Aaron Ulises Núñez Ávila
N° DE CONTROL: 15480456

MAESTRO: Ing. Juan Pablo Rosas Balazo

Guadalupe, Nuevo León, al 30 de Noviembre del 2018.

Proyecto 1: Árbol de Expresiones

Introducción

Un árbol binario es una estructura de datos en la cual cada nodo puede tener un hijo izquierdo y un hijo derecho. No pueden tener más de dos hijos (de ahí el nombre "binario"). Si algún hijo tiene como referencia a null, es decir que no almacena ningún dato, entonces este es llamado un nodo externo. En el caso contrario el hijo es llamado un nodo interno. Usos comunes de los árboles binarios son los árboles binarios de búsqueda, los montículos binarios y Codificación de Huffman.

Desarrollo

Se realizo un árbol binario de búsqueda de expresiones el cual automáticamente se encarga de reconocer los operadores y los números de manera que genera el árbol binario, el objetivo es que el usuario introduzca una expresión y el programa arroje la expresión en infija y postfija.

Se realizo en el lenguaje de programación java utilizando el ide NetBeans.

Lo primero que hicimos fue creas un “Pseudocodigo” el cual nos ayudo a comprender un poco mas la logística con la que debería contar el código para que pudiera realizar el objetivo final

Pseudocodigo

pedir expresión por teclado

almacenar la expresión en una variable “exp”

declaramos las pilas que se utilizaran En, Pt,Sa

agregamos la expresión a la pila entrada

declaramos la jerarquía de cada uno de los operadores

```
if (op.equals("^")) {
        prf = 5;
    }

    if (op.equals("*") || op.equals("/")) {
        prf = 4;
```

```
        }

        if (op.equals("+") || op.equals("-")) {
            prf = 3;
        }
```

creamos un ciclo para para infijo a postfijo

en base a la jerarquía de cada operador iremos acomodando los datos de la pila entrada en pt
para al final reacomodarlo en la ps

Experimento

Se realizaron 10 expresiones diferentes, en el cual fue tomado el tiempo de ejecucion de cada uno para así observar cual de las operaciones afectaba mas el tiempo en que se esta ejecutando en el programa.
Las pruebas de las expresiones se realizaron con las siguientes computadoras:

	COMPUTADORA 1	***COMPUTADORA 2***
EQUIPO	Laptop HP	ACER Aspire E15
RAM	6GB DDR3	8 GB DDR4
PROCESADOR	AMD A8-7410 APU with AMD Radeon R5 Graphics	AMD A9 7th GEN with Turbo CORE Technology up to 3.5
TIPO DE SISTEMA	Debian 9	WINDOWS 64 bits

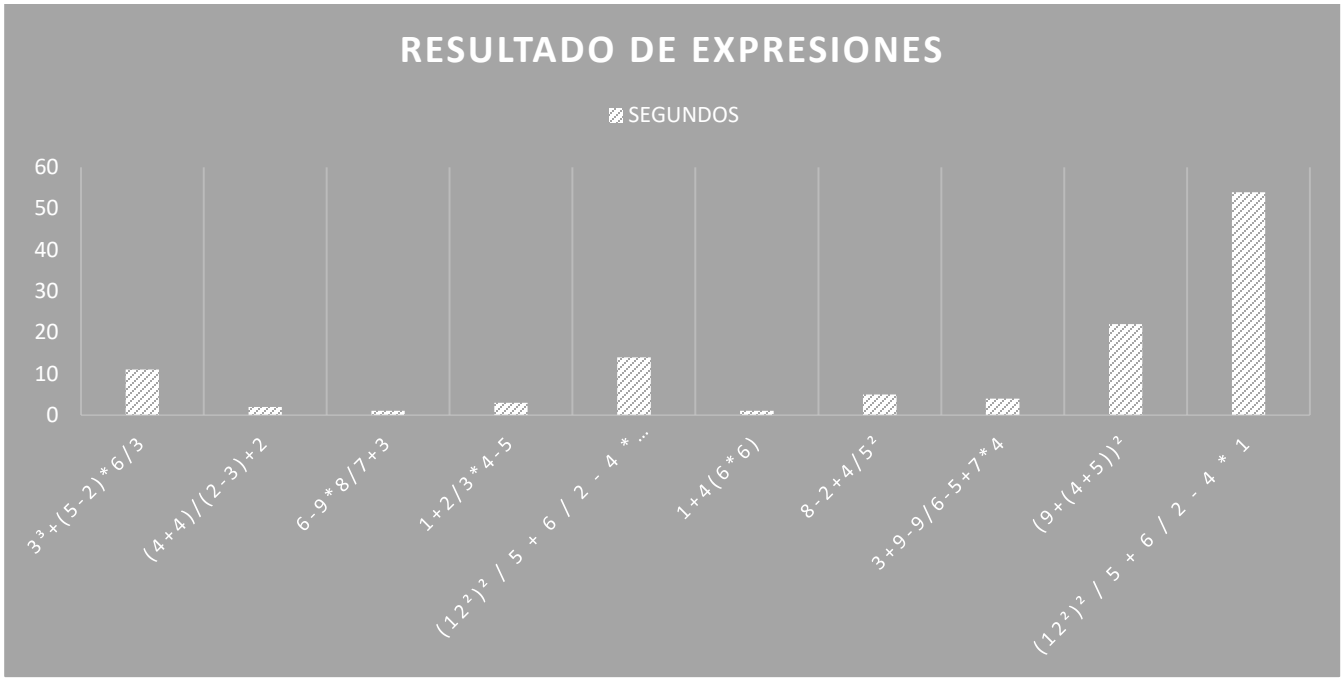
Pruebas

Estas son las siguientes expresiones que realizamos:

Expresión	Tiempo de ejecución(s)
$3^3+(5-2)*6/3$	11 seconds
(4+4)/(2-3)+2	2 seconds
6-9*8/7+3	1 second
1+2/3*4-5	3 seconds
$(12^2)^2$ / 5 + 6 / 2 - 4 * 1$(14^2)^2$	14 seconds
1+4(6*6)	1 second
$8-2+4/5^2$	5 seconds
3+9-9/6-5+7*4	4 second
$(9+(4+5))^2$	22 seconds
$(12^2)^2$ / 5 + 6 / 2 - 4 * 1	56 seconds

Resultados

En esta grafica mostramos el tiempo de ejecución que se tardo cada una de las expresiones.



Conclusiones

En este proyecto se realizo un programa que fuera capaz de realizar una función completa y de mismo modo que se tomara el tiempo que se tardaba en ejecutar.

La función tenía que ser con números y operadores aritméticos, ya que después tenia que mostrar el árbol de expresiones, que es llamado la notación.

Bibliografía

Obtenido de

http://interactivepython.org/runestone/static/pythoned/Trees/RepresentacionDeListaDeListas.html