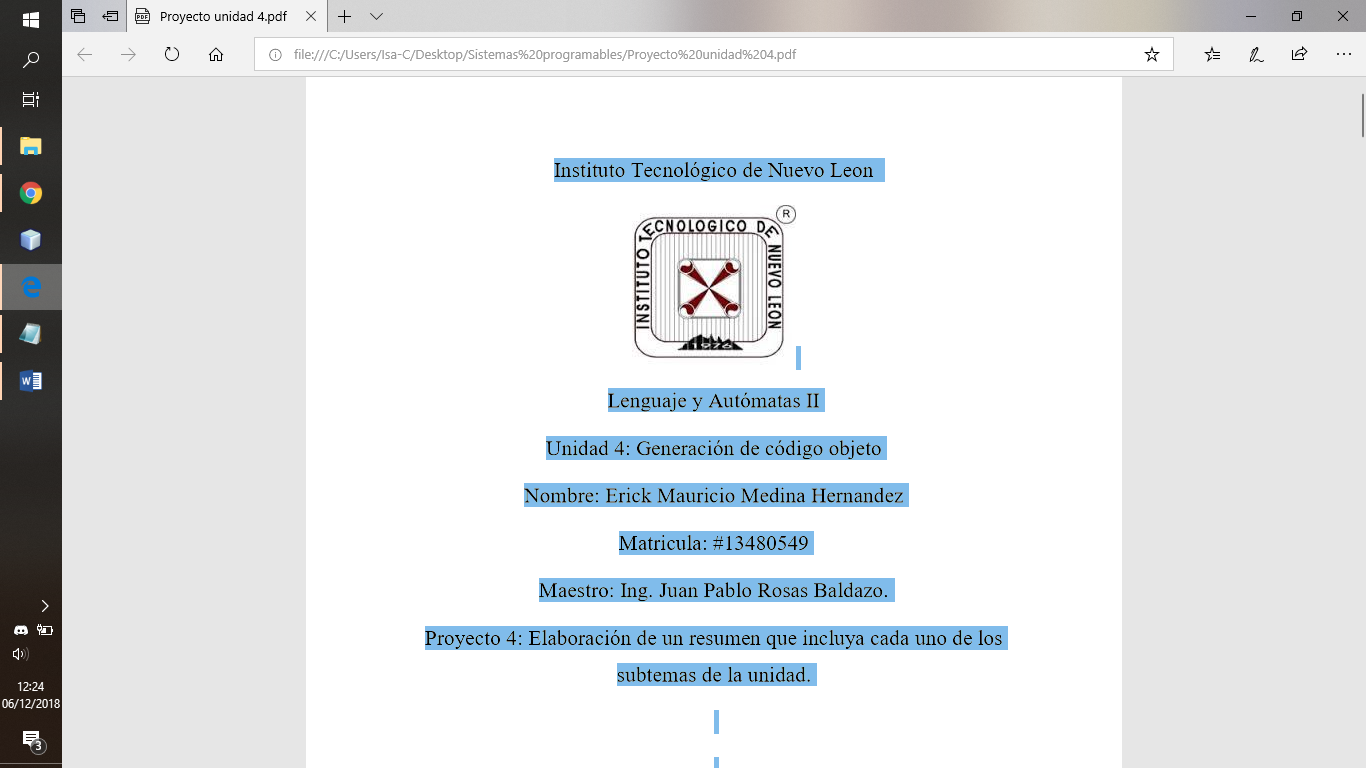
Instituto Tecnológico de Nuevo león



Lenguaje y Autómatas II

Unidad 1

Maestro: Rosas Baldazo Juan Pablo

Nombre: Oscar Isaac Chaires García

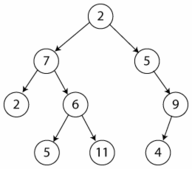
Matricula: 15480843

Actividad: Árbol de expresión

Cd. Guadalupe; Nuevo León; a 6 de diciembre de 2018

**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. Si algún hijo tiene como referencia a “null”, es decir que no almacena ningún dato, entonces este es llamado un nodo externo.



**Desarrollo**

Se realizó un programa en phyton, referente a un árbol binario que se encargara de imprimir el orden lógico de una expresión numérica. Se usó como referencia de arranque el borrador echo en clase y un enlace que indica cómo crear un árbol de expresiones en java.

**Métodos base a utilizar para el árbol recibido de pagina**[**http://pensandocomoprogramador.blogspot.com/2012/07/c8-arboles-de-expresion-python.html**](http://pensandocomoprogramador.blogspot.com/2012/07/c8-arboles-de-expresion-python.html) 1: def tomaToken(listaToken, esperado):

2: if listaToken[0] == esperado:

3: listaToken[0:1] = [] # quita el token

4: return 1

5: else:

6: return 0

7: def obtieneProducto(listaToken) :

8:

9: a = obtieneNumero(listaToken)

10: if tomaToken(listaToken, '\*') :

11: b = obtieneProducto(listaToken)

12: return Arbol('\*', a, b)

13: else :

14: return a

15:

16: def obtieneSuma(listaToken) :

17: a = obtieneProducto(listaToken)

18: if tomaToken(listaToken, '+') :

19: b = obtieneSuma(listaToken)

20: return Arbol('+', a, b)

21: else :

22: return a

23:

24: def obtieneNumero(listaToken):

25:

26: if tomaToken(listaToken, '(') :

27: x = obtieneSuma(listaToken) # obtiene subexpresion

28: tomaToken(listaToken, ')') # se come el cierre de parentesis

29: return x

30: else :

31: x = listaToken[0]

32:

33: if type(x) != type(0) : return None

34: listaToken[0:1] = [] # quita el token

35: return Arbol(x, None, None) # devuelve una hoja sin el numero

**Borrador echo en clase**

Operador=list("^\*/+-")

Operacion= (input("Ingresa la operacion sin espacios ni parentesis. /n Operadores compatilbles:

^\*/+-"))

x=0

y=0

def ArbolBinario(r):

return [r, [], []]

def insertarIzquierdo(raiz,nuevaRama):

t = raiz.pop(1)

if len(t) > 1:

raiz.insert(1,[nuevaRama,t,[]])

else:

raiz.insert(1,[nuevaRama, [], []])

return raiz

def insertarDerecho(raiz,nuevaRama):

t = raiz.pop(2)

if len(t) > 1:

raiz.insert(2,[nuevaRama,[],t])

else:

raiz.insert(2,[nuevaRama,[],[]])

return raiz

def obtenerValorRaiz(raiz):

return raiz[0]

def asignarValorRaiz(raiz,nuevoValor):

raiz[0] = nuevoValor

def obtenerHijoIzquierdo(raiz):

return raiz[1]

def obtenerHijoDerecho(raiz):

return raiz[2]

String Pila=[]

while x=0:

Op= (Operacion.find(Operador[y]))

Opizq=(Operacion.find(Operador[y])-1)

Opder=(Operacion.find(Operador[y])+1)

if Operacion.find(Operador[y])>=0:

else:

y +=1

print (Opizq, Op, Opder)

**Experimento**

Durante la prueba del programa se intentó correr en un simulador de una página educativa llamado NCLab, pero debido a pequeñas fallas técnicas de este simulador no se logró llevar a cabo el problema.

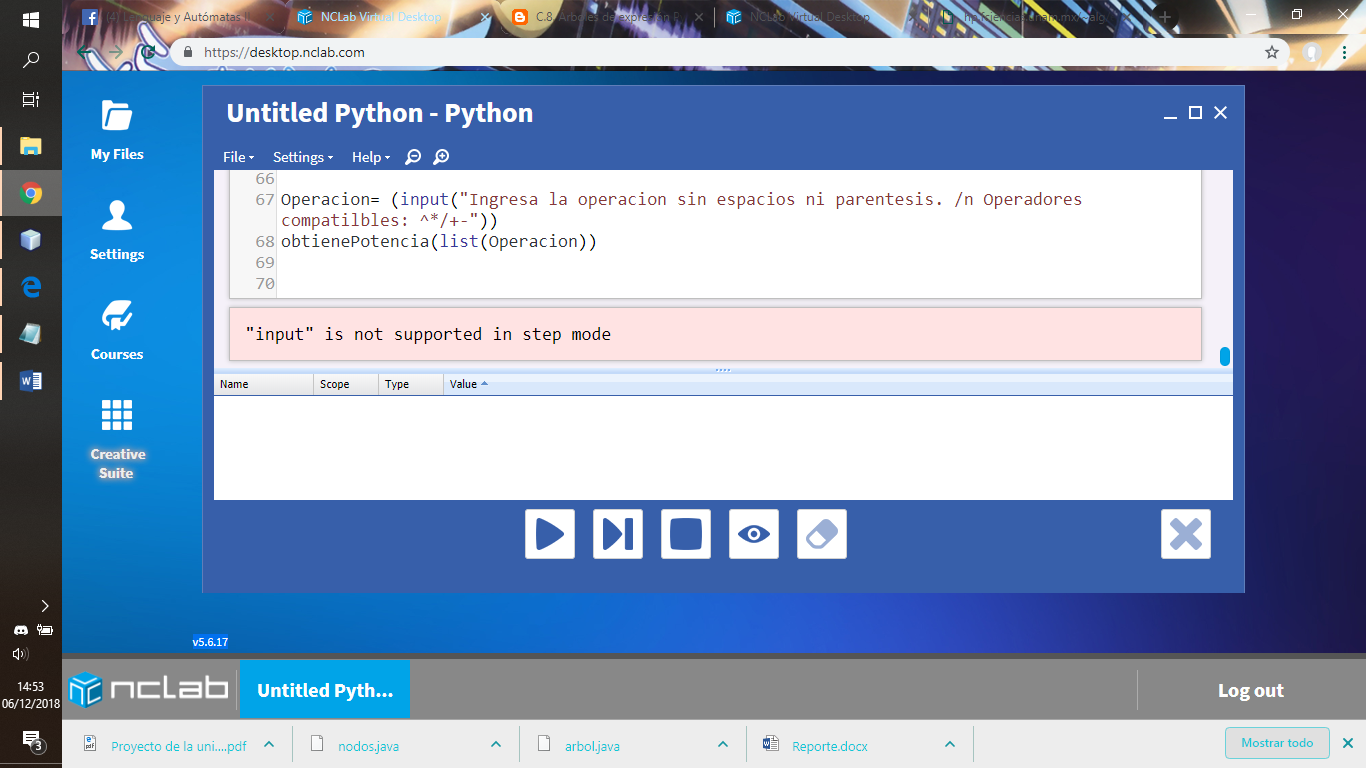
Equipo: Laptop Acer

Memoria RAM: 2 GB DDR2

Procesador: AMD DUAL-CORE

Sistema: Windows 10, 64 bits

**Resultado**



**Experimento 2**

Como se recibieron problemas al momento de realizar el trabajo, se tomó la decisión de realizar el trabajo en otro lenguaje de programación con el que tenga mejor familiarización. Se utilizo el programa NetBeans para hacer el código en lenguaje java.

Equipo: Laptop Acer

Memoria RAM: 2 GB DDR2

Procesador: AMD DUAL-CORE

Sistema: Windows 10, 64 bits

**Resultado 2**

|  |  |
| --- | --- |
| **Expresión** | **Resultado (seg)** |
| **3^3+(5-2)\*6/3** | **14** |
| **(4+4)/(2-3)+2** | **3** |
| **6-9\*8/7+3** | **2** |
| **1+2/3\*4-5** | **5** |
| **(12^2)^2/5+6/2-4\*1(14^2)^2** | **16** |
| **1+4(6\*6)** | **2** |
| **8-2+4/5^2** | **7** |
| **3+9-9/6-5+7\*4** | **6** |
| **(9+5(4+5))2** | **25** |
| **(12^2)^2/5+6/2-4\*1** | **60** |

**Conclusión**

A pesar de los inconvenientes, el árbol de expresiones logro ser terminado. El Python es un lenguaje de programación muy amigable pero debido a la plataforma en la que trabajaba me llego a ser tedioso y como consecuencia preferí realizar el trabajo en Java.

**Referencias**

C.8. Árboles de expresión Python. (2012, 27 agosto). Recuperado 21 septiembre, 2018, de <http://pensandocomoprogramador.blogspot.com/2012/07/c8-arboles-de-expresion-python.html>

Lenguajes y Aut. II [Presencial]. (2018, 21 septiembre). Recuperado 6 diciembre, 2018, de http://pensandocomoprogramador.blogspot.com/2012/07/c8-arboles-de-expresion-python.html