



INSTITUTO POLITÉCNICO NACIONAL.

ESCUELA SUPERIOR DE CÓMPUTO.

TEORÍA COMPUTACIONAL

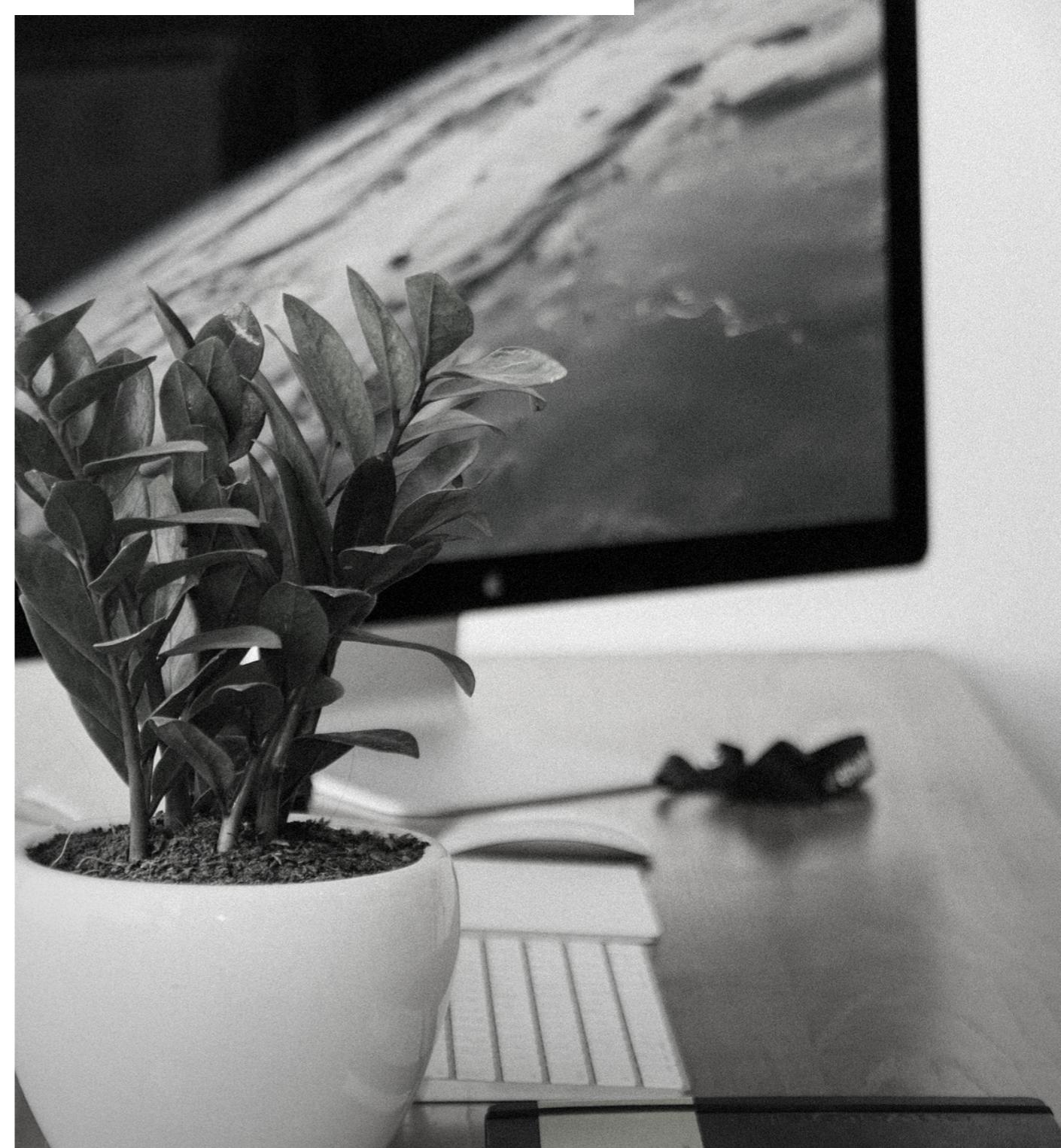
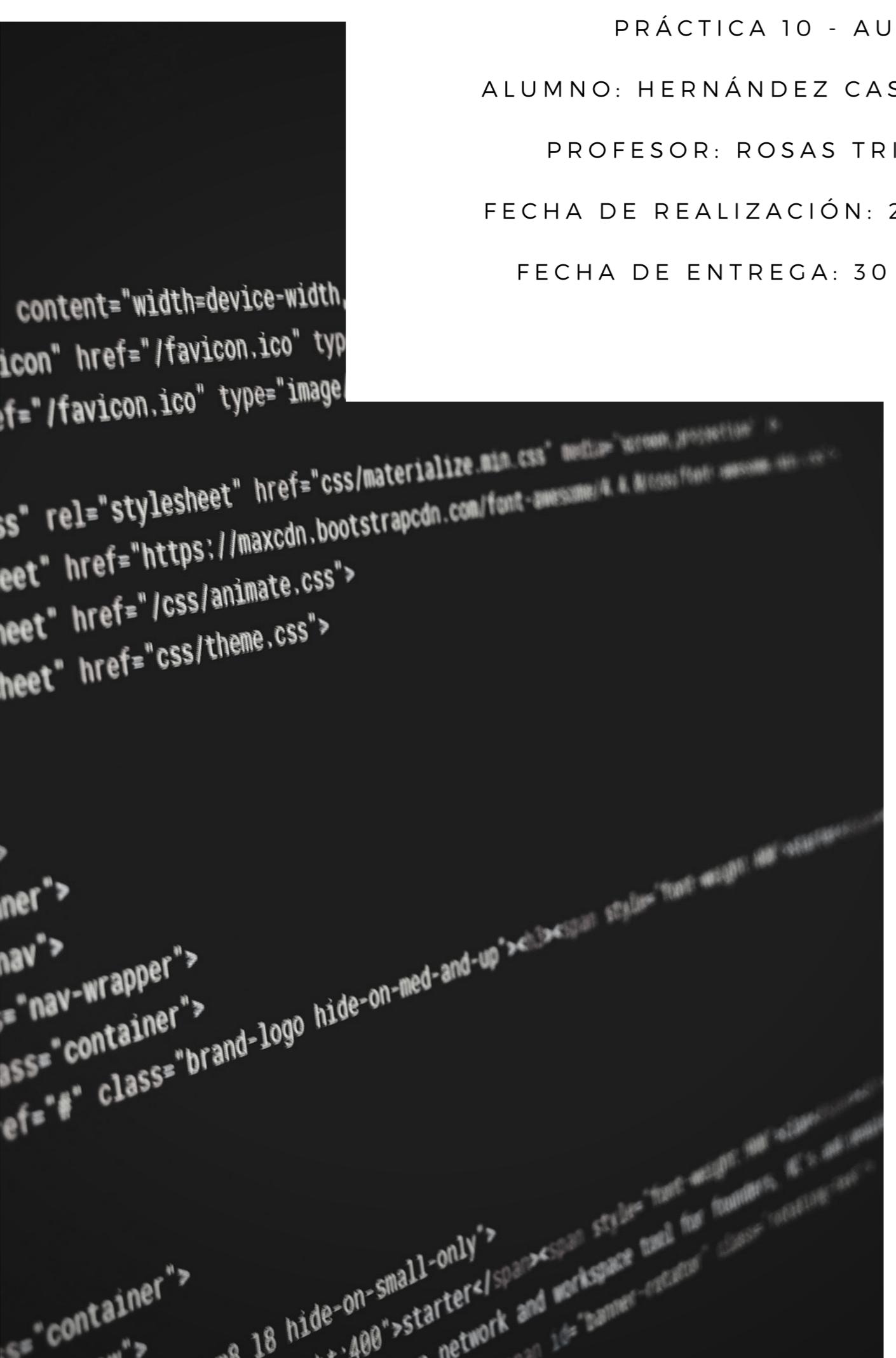
PRÁCTICA 10 - AUTÓMATA DE PILA

ALUMNO: HERNÁNDEZ CASTELLANOS CÉSAR URIEL.

PROFESOR: ROSAS TRIGUEROS JORGE LUIS

FECHA DE REALIZACIÓN: 25 DE OCTUBRE DEL 2017

FECHA DE ENTREGA: 30 DE OCTUBRE DEL 2017



## Marco teórico.

Los autómatas de pila, en forma similar a como se usan los autómatas finitos, también se pueden utilizar para aceptar cadenas de un lenguaje definido sobre un alfabeto A. Los autómatas de pila pueden aceptar lenguajes que no pueden aceptar los autómatas finitos. Un autómata de pila cuenta con una cinta de entrada y un mecanismo de control que puede encontrarse en uno de entre un número finito de estados. Uno de estos estados se designa como estado inicial, y además algunos estados se llaman de aceptación o finales. A diferencia de los autómatas finitos, los autómatas de pila cuentan con una memoria auxiliar llamada pila. Los símbolos (llamados símbolos de pila) pueden ser insertados o extraídos de la pila, de acuerdo con el manejo last-in-first-out (LIFO). Las transiciones entre los estados que ejecutan los autómatas de pila dependen de los símbolos de entrada y de los símbolos de la pila. El autómata acepta una cadena x si la secuencia de transiciones, comenzando en estado inicial y con pila vacía, conduce a un estado final, después de leer toda la cadena x.

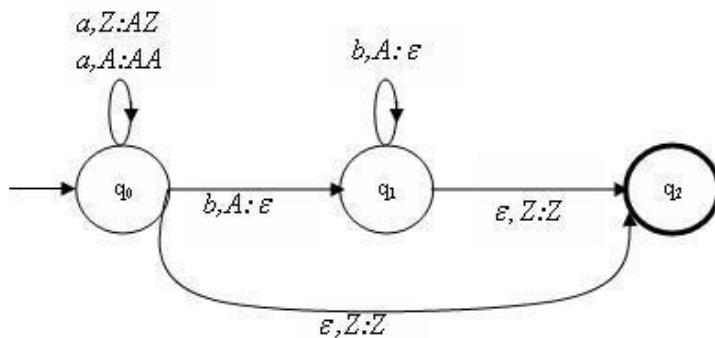


Imagen 1.0 Autómata de pila.

### Autómata de pila reconocedor determinístico

$$APD = \langle E, A, P, \delta, e_0, Z_0, F \rangle$$

E: Conjunto finito de *estados*,

A: *Alfabeto* o conjunto finito de símbolos de la cinta de entrada,

P: *Alfabeto* o conjunto finito de símbolos de la Pila.  $P \cap A = \emptyset$

$\delta$ : *función de transición de estados*

$e_0$ : *Estado inicial*  $e_0 \in E$ .

$Z_0$ : Símbolo distinguido  $Z_0 \in P$

F: Conjunto de *estados finales o estados de aceptación*.  $F \subseteq E$ .

### Ejemplo 1

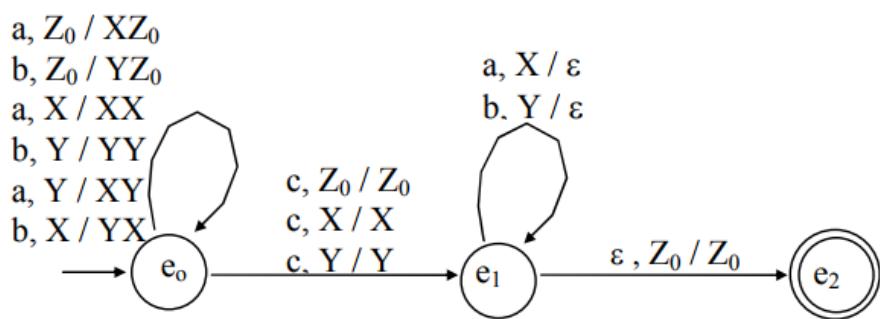
$$A = \{a, b, c\}$$

$$L_1 = \{\omega c \omega^R \mid \omega \in \{a, b\}^*\}$$

APD<sub>1</sub> es un autómata de pila que reconoce L<sub>1</sub>.

$$\text{APD}_1 = \langle \{e_0, e_1, e_2\}, \{a, b, c\}, \{X, Y, Z_0\}, \delta, e_0, Z_0, \{e_2\} \rangle$$

$\delta$ :



Ejemplos de cadenas aceptadas ó no aceptadas por AP<sub>1</sub>

$$\delta^*(e_0, abcba) = e_2 \quad abcba \in L_1$$

$$\delta^*(e_0, c) = e_2 \quad c \in L_1$$

$$\delta^*(e_0, abcab) = e_1 \quad abcab \notin L_1$$

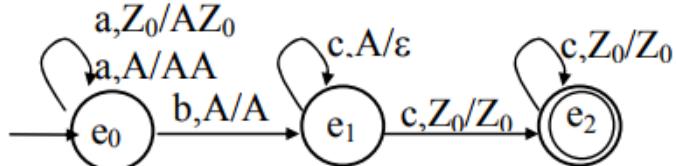
$$\delta^*(e_0, a) = e_0 \quad a \notin L_1$$

### Ejemplo 2

$$L_2 = \{a^i b c^k \mid i, k \geq 1 \text{ y } i < k\}$$

$$\text{APD}_2 = \langle \{e_0, e_1, e_2\}, \{a, b, c\}, \{A, Z_0\}, \delta_2, e_0, Z_0, \{e_2\} \rangle$$

$\delta_2$ :

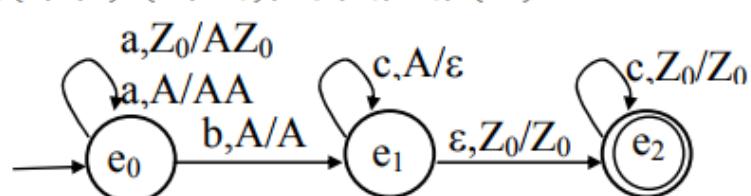


### Ejemplo 3

$$L_3 = \{a^i b c^k \mid i, k \geq 1 \text{ y } i \leq k\}$$

$$\text{APD}_3 = \langle \{e_0, e_1, e_2\}, \{a, b, c\}, \{A, Z_0\}, \delta_3, e_0, Z_0, \{e_2\} \rangle$$

$\delta_3$ :



## **Material y equipo.**

El material utilizado en la práctica es el siguiente:

### **Herramientas de software:**

- \* Oracle VM VirtualBox
- \* Ubuntu 17.04
- \* Python 3.6.2
- \* eric6 Web Browser (QtWebKit)
- \* JFlap

### **Herramientas de hardware:**

- \* Computadora personal

## **Desarrollo de la práctica.**

La práctica número ocho de teoría computacional consistió en implementar un autómata de pila en el lenguaje python que reconociera el siguiente lenguaje:

$$L = \{a^n b^n \mid n \geq 0\}$$

Para poder llevar a cabo esta práctica se hizo uso de la herramienta JFLAP el cual nos auxilió para llevar a cabo, mediante la experimentación y el error, fue como se fue construyendo el automáta hasta llegar a la versión optima del autómata, para posteriormente implementarlo en el lenguaje de programación python.

Las dificultades que se pudiero observar en la práctica, fue la falta de práctica con esta clase de autómatas, por ende una menor habilidad para

desarrollar autómatas de pila con mayor facilidad, el código empleado se presenta a continuación:

```
# -*- coding: utf-8 -*-
#!/usr/bin/python

import os

Q={'q1','q2','q3'}
s='q1'
f='q3'
Sigma = ['a','b']
DELTA={('q1','a','z'): ('q1',['A','Z']),
       ('q1','b','A'): ('q2,['e']),
       ('q2','b','A'): ('q2,['e']),
       ('q2','e','z'): ('q3,['z']),
       ('q1','a','A'): ('q1,['A','A'])}
GAMMA={'e','z','A'}
pila=['z']
pila.append('A')
gamma=pila.pop()

def probadorDeAplicacion():
    valores = raw_input("Escriba los valores:")
    validar(valores)
    automata_de_pila(valores)

def validar(val):
    flag = 0
    for x in val:
        if x != Sigma[0] and x != Sigma[1]:
            flag = 1
    if flag == 1:
        print "Esta palabra no es valida tiene que ser de la forma: "
        print "{a^n b^n | n >= 0}"
        exit()
    else :
        posicion = val.find("b")
        recorre_unos = val[posicion:]
        recorre_ceros = val[:posicion]
        for y in recorre_unos:
            if y == "a":
                print "Esta palabra no es valida tiene que ser de la forma: "
                print "{axB0n bxB0n | n >= 0}"
                exit()
        if len(recorre_unos) != len(recorre_ceros):
            print "Esta palabra no es valida tiene que ser de la forma: "
            print "{axB0n bxB0n | n >= 0}"
            exit()
```

```

def automata_de_pila(val):
    pila = ["#"]
    alfabeto_pila = "z"
    # print "Palabra del alfabeto: ",val
    # print "Estado inicial de la pila: ",pila
    print
    print
    for i in val:
        if i == "a":
            pila.append(alfabeto_pila)
            print "pila: ",pila
            print
        if i == "b":
            pila.pop()
            print "pila: ",pila
            print
    print "La cadena es aceptada"

probadorDeAplicacion()

```

Resultados obtenidos al ejecutar la aplicación.

```

uriel@Uriel-PC:~/Escritorio$ python adp.py
Escriba los valores:aaabbb

pila: ['#', 'z']

pila: ['#', 'z', 'z']

pila: ['#', 'z', 'z', 'z']

pila: ['#', 'z', 'z']

pila: ['#', 'z']

pila: ['#']

La cadena es aceptada
uriel@Uriel-PC:~/Escritorio$ python adp.py
Escriba los valores:aaaababababbabababaababababa
Esta palabra no es valida tiene que ser de la forma:
{an bn | n >= 0}
uriel@Uriel-PC:~/Escritorio$ 

```

## **Conclusiones y recomendaciones.**

El trabajo descrito en esta práctica me aportó el aprendizaje de los automátas de pila, saber el como funcionan, el como conformar un automáta que reconozca el lenguaje de nuestro interés.

Los automátas de pila son similares a los ADSs pero con la diferencia en que estos pueden contar con una memoria en la cual se le almacena información.

En cuanto a las dificultades que se presentó en el desarrollo de la práctica, fue la falta de experiencia en estos automátas, en mi opinión sería bueno considerar algún tipo de listas de ejercicios de la materia, con el propósito de ejercitarse la teoría aprendida en clase.

## Referencias

- Delta.cs.cinvestav.mx. (2017). *Autómatas de pila*. [online] Available at: <http://delta.cs.cinvestav.mx/~gmorales/ta/node14.html> [Accessed 31 Oct. 2017].
- Es.wikipedia.org. (2017). *Autómata con pila*. [online] Available at: [https://es.wikipedia.org/wiki/Aut%C3%B3mata\\_con\\_pila](https://es.wikipedia.org/wiki/Aut%C3%B3mata_con_pila) [Accessed 31 Oct. 2017].
- Kelley, D. (2001). *Teoría de autómatas y lenguajes formales*. Madrid: Prentice Hall.
- Uhu.es. (2017). *Citar un sitio web - Cite This For Me*. [online] Available at: <http://www.uhu.es/francisco.moreno/talf/docs/tema7.pdf> [Accessed 31 Oct. 2017].