INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

TEORÍA COMPUTACIONAL

2CM4

PROFESOR: LUZ MARÍA SÁNCHEZ GARCÍA

PRÁCTICA 6 AUTÓMATA DE PILA

INTEGRANTES:

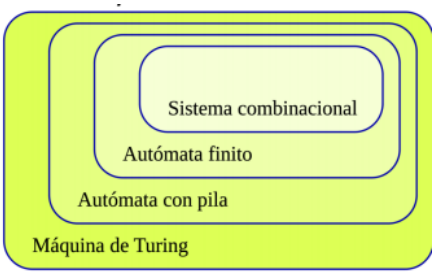
VÁZQUEZ MORENO MARCOS OSWALDO	**2016601777**
QUINTANA RUÍZ AJITZI RICARDO	**2017631261**

FECHA DE ENTREGA: 24 DE MAYO DE 2018

INTRODUCCIÓN

En la siguiente práctica se pretende codificar un programa que simule la ejecución de un autómata de pila el cual es un modelo matemático de un sistema que recibe una cadena constituida por símbolos de un alfabeto y determina si esa cadena pertenece al lenguaje que el autómata reconoce.

A su vez, el lenguaje que reconoce un autómata con pila pertenece al grupo de lenguajes libres de contexto en la clasificación de la jerarquía de Chomsky.



PLANTEAMIENTO DEL PROBLEMA

Implementar el algoritmo de codificación de un programa el cual permita ingresar una cadena de caracteres con un lenguaje definido por:

$\{a^n b^n$ donde n>0$\}$

En donde acepte los siguientes casos:

1.- ab

2.-aabb

3.-aaabbb

Y no acepte los siguientes:

1.-ba

2.-aba

3.-baa

4.-abb

5.-abba

8.-baba

9.-aabbaa

10.-aabaa

11.-baaab

12.-abaaba

6.-aaab

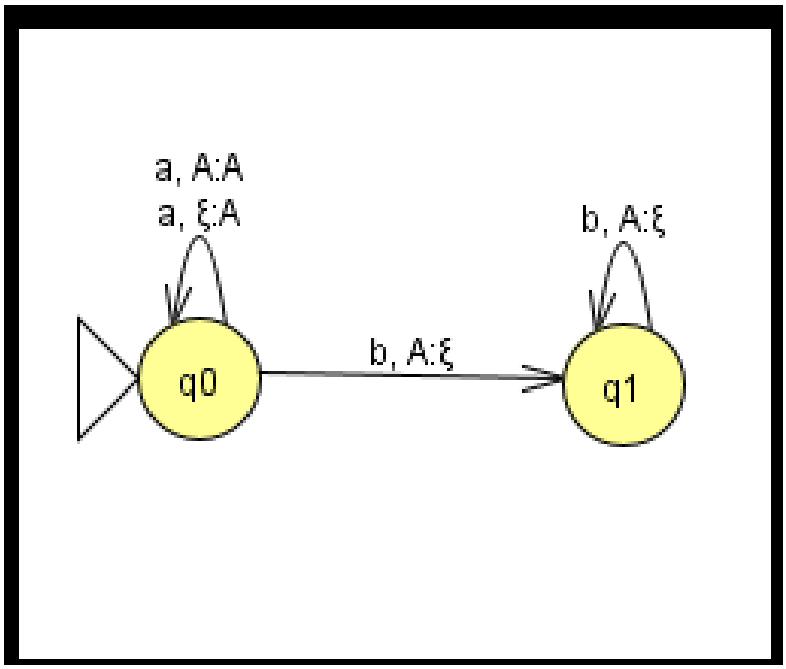
7.-abab

Teniendo de esta manera validación de cadenas con un método distinto, obteniendo en cada inserción una pila nueva, donde ella misma sabrá de acuerdo con nuestro autómata programado si podemos o no aceptar la cadena, si la pila debe de estar vacía, si podemos o no insertar el símbolo de pila correspondiente.

DISEÑO DE LA SOLUCIÓN

Una vez planteado el problema lo llevamos al lenguaje de programación Java en donde es más fácil crear pilas, en este caso es una pila muy sencilla y sin necesidad de hacer un gran esfuerzo o muchas validaciones, es por eso por lo que se decidió tomar este problema en Java, creando una clase “AutomataPila” que extenderá a un JFrame el cual es nuestra interfaz gráfica en donde podremos ingresar la cadena dentro de un *TextField* creando nuestra validación mediante un botón con el método *ActionListener* el cual al ser presionado entra a diferentes ciclos para lograr la validación tomando en cuenta las condiciones y el lenguaje principalmente, dando así como resultado en un JLabel el mensaje de si la cadena fue aceptada o no, por otra parte, no solo tenemos la validación de la cadena en el entorno gráfico sino que mostramos el contenido de la pila en consola, mostrando inserciones o bien si la cadena queda vacía o tiene algún contenido.

Siendo este nuestro autómata diseñado en JFlap:



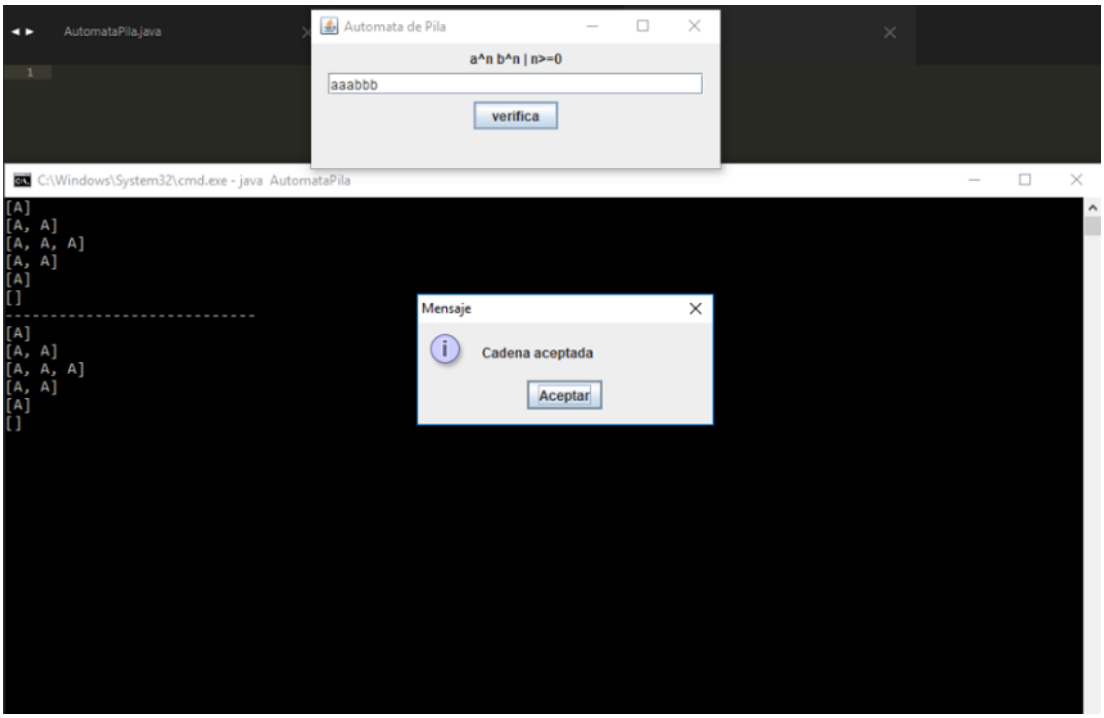
IMPLEMENTACIÓN DE LA SOLUCIÓN

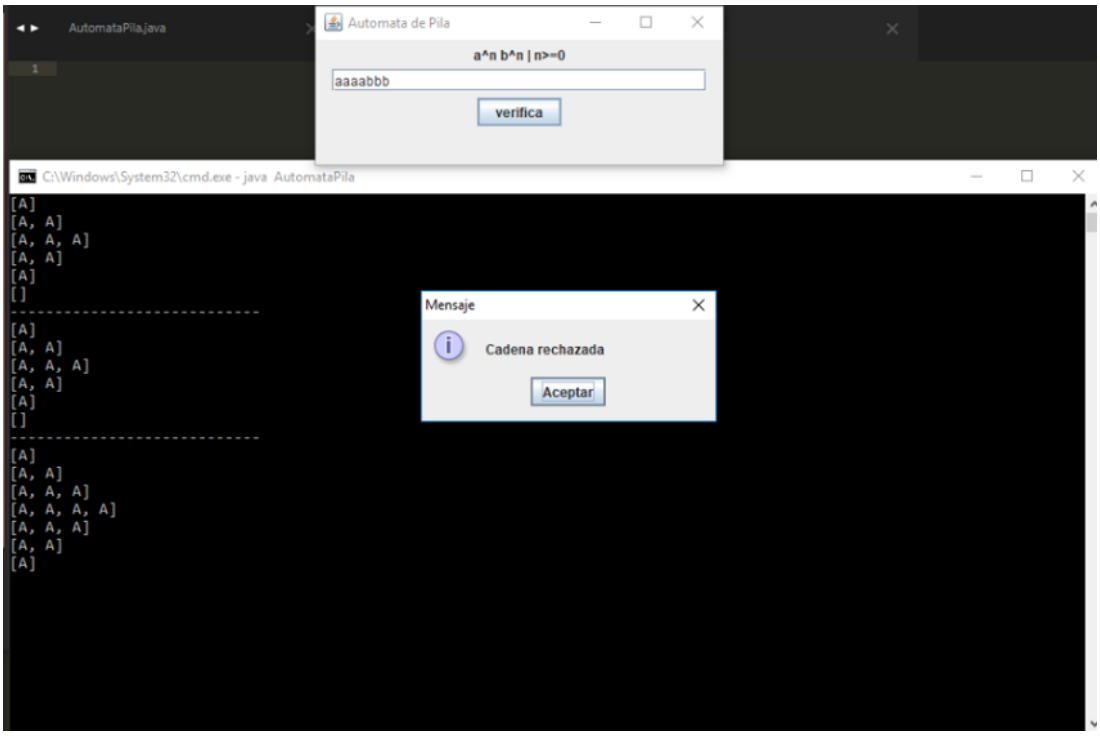
```java
import java.util.Stack;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.EmptyStackException;

public class AutomataPila extends JFrame implements ActionListener{

    Stack<Character> pila = new Stack<Character>();
    JTextField cadena;
    JLabel resultado,encabezado;
    JButton verifica;

    public AutomataPila(){
        cadena=new JTextField(30);
        encabezado=new JLabel("a^n b^n | n>=0");
        verifica=new JButton("verifica");
        verifica.addActionListener(this);

        setTitle("Automata de Pila");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel=new JPanel();
        panel.setLayout(new FlowLayout());
        panel.add(encabezado);
        panel.add(cadena);
        panel.add(verifica);

        Container cp=getContentPane();
        cp.add(panel,BorderLayout.CENTER);

        setSize(380,150); setVisible(true);
    }
```

```java
    public void actionPerformed(ActionEvent e){
        String c=cadena.getText();
        vaciar(pila);
        boolean flag=false;
        try{
            if(validar(c)){
                for (int n=0;n<c.length();n++ ) {
                    char a = c.charAt (n);
                    if (a=='a'&& pila.empty()&&flag==false) {
                        pila.push('A');
                        System.out.println(pila);
                        flag=true;
                    }
                    else if(a=='a'&& pila.peek()=='A'){
                        pila.push('A');
                        System.out.println(pila);
                    }
                    else if (a=='b'&&pila.peek()=='A') {
                        pila.pop();
                        System.out.println(pila);
                    }
                    else if (a=='a'&&pila.empty()&&flag) {
                        JOptionPane.showMessageDialog(null,"Cadena rechazada");
                    }
                    else if (a=='b'&&pila.empty()) {
                        JOptionPane.showMessageDialog(null,"Cadena rechazada");
                    }
                    else
                        JOptionPane.showMessageDialog(null,"Cadena rechazada");

                }
                if (pila.empty())
                    JOptionPane.showMessageDialog(null,"Cadena aceptada");
                else
```

```
67                      JOptionPane.showMessageDialog(null,"Cadena aceptada");
68                  else
69                      JOptionPane.showMessageDialog(null,"Cadena rechazada");
70              }
71              else
72                  JOptionPane.showMessageDialog(null,"El lenguaje no pertenece al Automata");
73          }
74          catch(EmptyStackException ex){
75              JOptionPane.showMessageDialog(null,"Cadena rechazada");
76          }
77          System.out.println("---------------------------");
78      }
79
80      public boolean validar(String s){
81          for (int n=0;n<s.length() ;n++ ) {
82               char c = s.charAt (n);
83               if (c!='a'&&c!='b')
84                  return false;
85
86          }
87          return true;
88      }
89
90      public void vaciar(Stack pila){
91          while(pila.empty()==false)
92              pila.pop();
93      }
94
95      public static void main(String[] args) {
96          AutomataPila a= new AutomataPila();
97      }
98  }
```

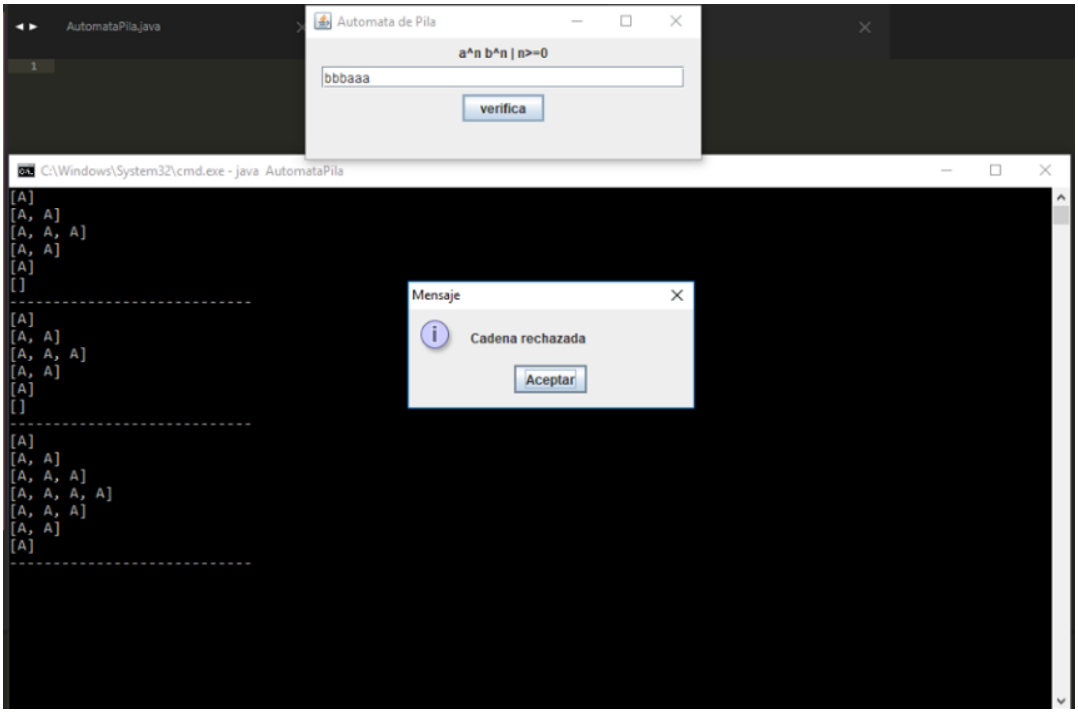
FUNCIONAMIENTO




AutomataPila.java
Automata de Pila
a^n b^n | n>=0
aaaabbb
verifica
C:\Windows\System32\cmd.exe - java AutomataPila
[A]
[A, A]
[A, A, A]
[A, A]
[A]
[]

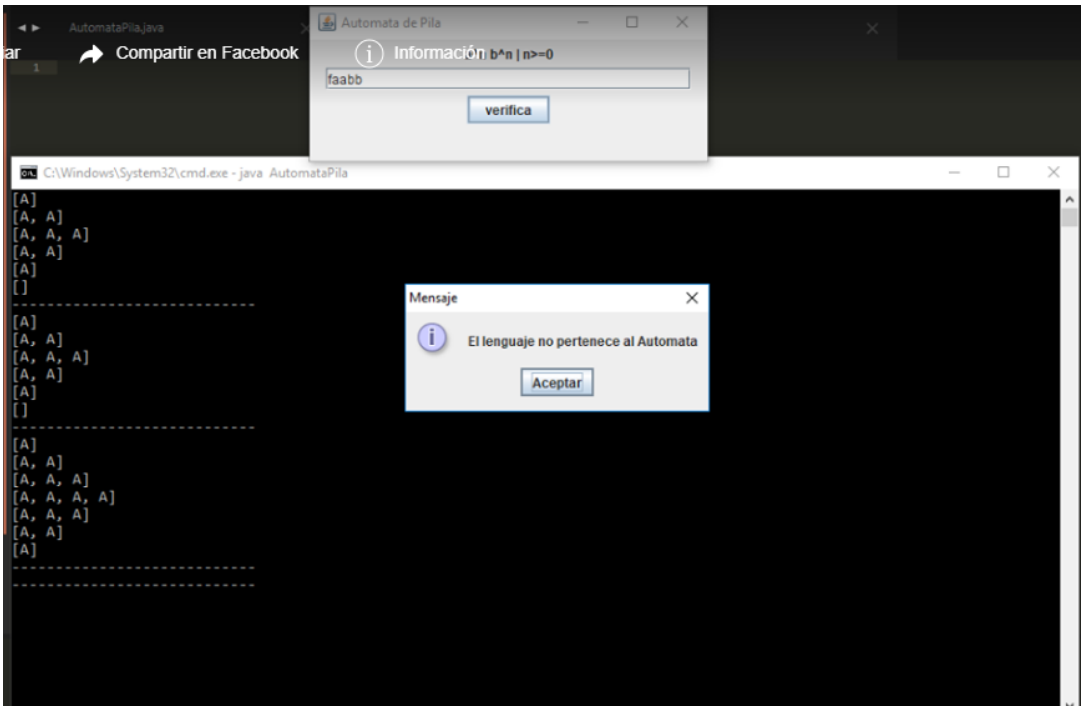
[A]
[A, A]
[A, A, A]
[A, A]
[A]
[]

[A]
[A, A]
[A, A, A]
[A, A, A, A]
[A, A, A]
[A, A]
[A]
Mensaje
Cadena rechazada
Aceptar


AutomataPila.java
Automata de Pila
a^n b^n | n>=0
bbbaaa
verifica
C:\Windows\System32\cmd.exe - java AutomataPila
[A]
[A, A]
[A, A, A]
[A, A]
[A]
[]

[A]
[A, A]
[A, A, A]
[A, A]
[A]
[]

[A]
[A, A]
[A, A, A]
[A, A, A, A]
[A, A, A]
[A, A]
[A]
Mensaje
Cadena rechazada
Aceptar



CONCLUSIONES

En conclusión, podemos decir que fue una práctica sencilla, con un grado de dificultad de acuerdo a lo que el modelo matemático requiere, sin duda debemos mencionar que el hecho de realizar la práctica en lenguaje Java facilito todo ya que si hubiera sido en lenguaje C hubiéramos tenido que crear la pila por nuestra propia mano, de esta manera concluyo que es de gran utilidad controlar no solo un lenguaje de programación sino dominar varios para este tipo de casos en donde nos damos cuenta que lo más viable es salir de lo cotidiano.

Por otro lado, fue un gran punto de motivación para nosotros que lo aprendido en programación orientada a objetos nos sirviera para llevar una práctica a un entorno gráfico, se vea mejor y sea más entendible, de la misma manera tener más organización en cuanto a información.