INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

Teoría Computacional

Práctica 6

Autómata de Pila

Profesora: Luz María Sánchez García

Grupo: 2CM1

Alumnos:

Mendoza Parra Sergio.

Paz Sánchez Brandon

MEXICO, D.F. a 20 de Mayo del 2017

**Introducción:**

En este programa se verá el funcionamiento de un autómata de pila el cual tendremos que validar una cadena de lenguaje L = {a^n b^n | n >= 2}, de este modo el usuario tendrá que dar una cadena ya sea válida o no y el programa dependiendo del autómata que tenga decidirá si es una cadena valida o no y por otro lado también se tendrá que imprimir los elementos que hay en la pila por ejemplo, si la pila tiene elementos, se tendrá que imprimir esos elementos en la pila o de lo contrario si fue una cadena válida solo se tiene que imprimir “Zo” en el tope de la pila mostrando que solo hay ese elemento y por lo tanto la cadena fue válida.

También se explicará cómo fue la solución del problema al igual que la explicación teórica de cómo fue pasando letra por letra a cada estado del autómata y finalmente ver el resultado en terminal.

**Planteamiento del Problema:**

Implementar un programa que identifique si una palabra es aceptada en un autómata e imprimir las derivaciones de la cadena al igual que los elementos de la pila; el lenguaje de este autómata solo serán cadenas que contengas las letras “a” y “b”, de este modo el lenguaje será:

Y el autómata a desarrollar será el siguiente:

Ab | AB

Ba | BA

Zoa | ZoA

Zob | ZoB

Aa | λ

Bb | λ

Ab | AB

Ba | BA

Zo | λ

Aa | λ

Bb | λ

**Diseño y funcionamiento de la solución:**

Para dar solución al autómata de pila lo que tenemos que hacer primero es ver qué tipo de cadenas son aceptadas para el autómata para este problema, de este modo primero analizamos y vemos que las únicas palabras que pueden ser aceptadas por el autómata son palabras tipo palíndroma, esto quiere decir que solo acepta el mismo número de a’s como de b’s de este modo tenemos la siguiente tabla la cual ingresaremos algunos ejemplos de palabras aceptadas tanto las no aceptadas:

|  |  |
| --- | --- |
| Cadenas | |
| Aceptadas | No Aceptadas |
| baab | aabbbb |
| bababaababab | baabbabb |
| abaaba | aba |
| aa | bbaa |
| abba | bababa |
|  |  |

Analizaremos primero un caso de las palabras aceptadas, por lo tanto tenemos lo siguiente:

|  |  |  |  |
| --- | --- | --- | --- |
| b | a | a | b |

Palabra:

Estados de la Pila

Para: b Para: ba Para: baa Para: baab

Se ingresa “B” Se ingresa “A” Se quita “A” Se quita “B”

|  |
| --- |
|  |
|  |
| B |
| Zo |

|  |
| --- |
|  |
| A |
| B |
| Zo |

|  |
| --- |
|  |
|  |
| B |
| Zo |

|  |
| --- |
|  |
|  |
|  |
| Zo |

Derivaciones: Cadena aceptada

S => 1 Zob | ZoB = > 1 Ba | BA = > 2 Aa | e = > 2 Bb | e = > 3 Zo | e

Esta cadena es aceptada ya que como lo había mencionado antes si en la pila solo queda “Zo” automáticamente es aceptada porque Zo nos define si una cadena es aceptada o no.

Ahora analizaremos una cadena que no sea aceptada para eso haremos el mismo procedimiento que hicimos con la cadena aceptada, solo que en este caso hay varias opciones por las cuales una cadena no puede ser aceptada o no como por ejemplo:

1. La cadena que se ingresó se quede en una transición siempre.
2. La cadena quiera sacar elementos que ya no hay en la pila.

De este modo solo explicaremos 2 para que haya mejor comprensión a lo que estamos diciendo, así que analizaremos la cadena “bbaa” y la cadena “aba”:

|  |  |  |  |
| --- | --- | --- | --- |
| b | b | a | a |

Palabra:

Estados de la Pila

Para: b Para: bb Para: bba

Se ingresa “B” Se quita “B” Se quita “A” y sale un error

|  |
| --- |
|  |
|  |
| B |
| Zo |

|  |
| --- |
|  |
|  |
|  |
| Zo |

|  |
| --- |
|  |
|  |
|  |
| Zo |

Derivaciones:

S => 1 Zob | ZoB = > 2 Ba | BA = > 2

Cadena no aceptada

¿Porque no es aceptada? R.- Como se vio la cadena no pasa a ninguna transición después de la primera “a”, porque no se puede ingresar la letra A cuanto tenemos “Zo” en la pila, de este modo el programa solo manda un mensaje de que la cadena no fue aceptada y aparte manda la transición en la que se quedó la cadena.

|  |  |  |
| --- | --- | --- |
| a | b | a |

Palabra:

Estados de la Pila

Para: a Para: ab Para: aba

Se ingresa “A” Se quita “B” Se quita “A” y sale un error

|  |
| --- |
|  |
|  |
| A |
| Zo |

|  |
| --- |
|  |
| B |
| A |
| Zo |

|  |
| --- |
| A |
| B |
| A |
| Zo |

Derivaciones:

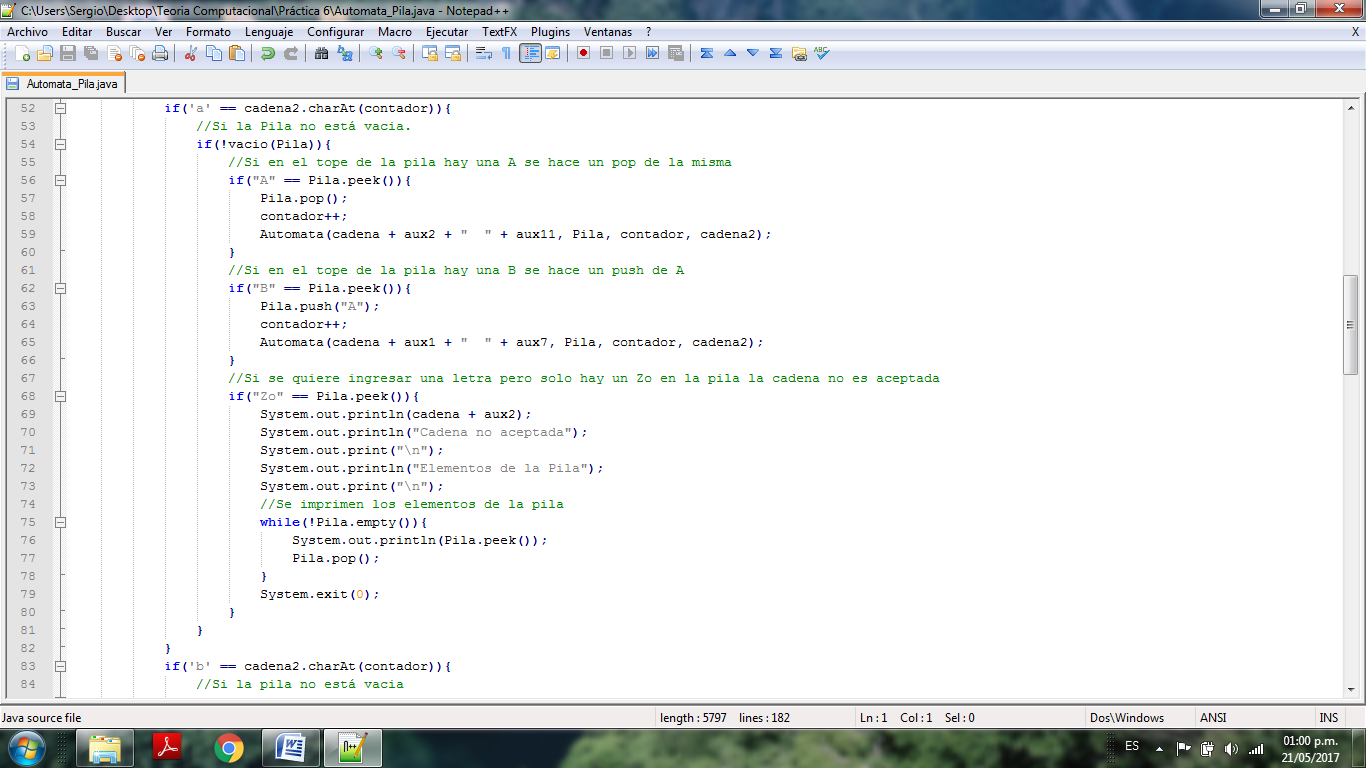
S => 1 Zob | ZoA = > 1 Ab | AB = > 1 Ba | BA

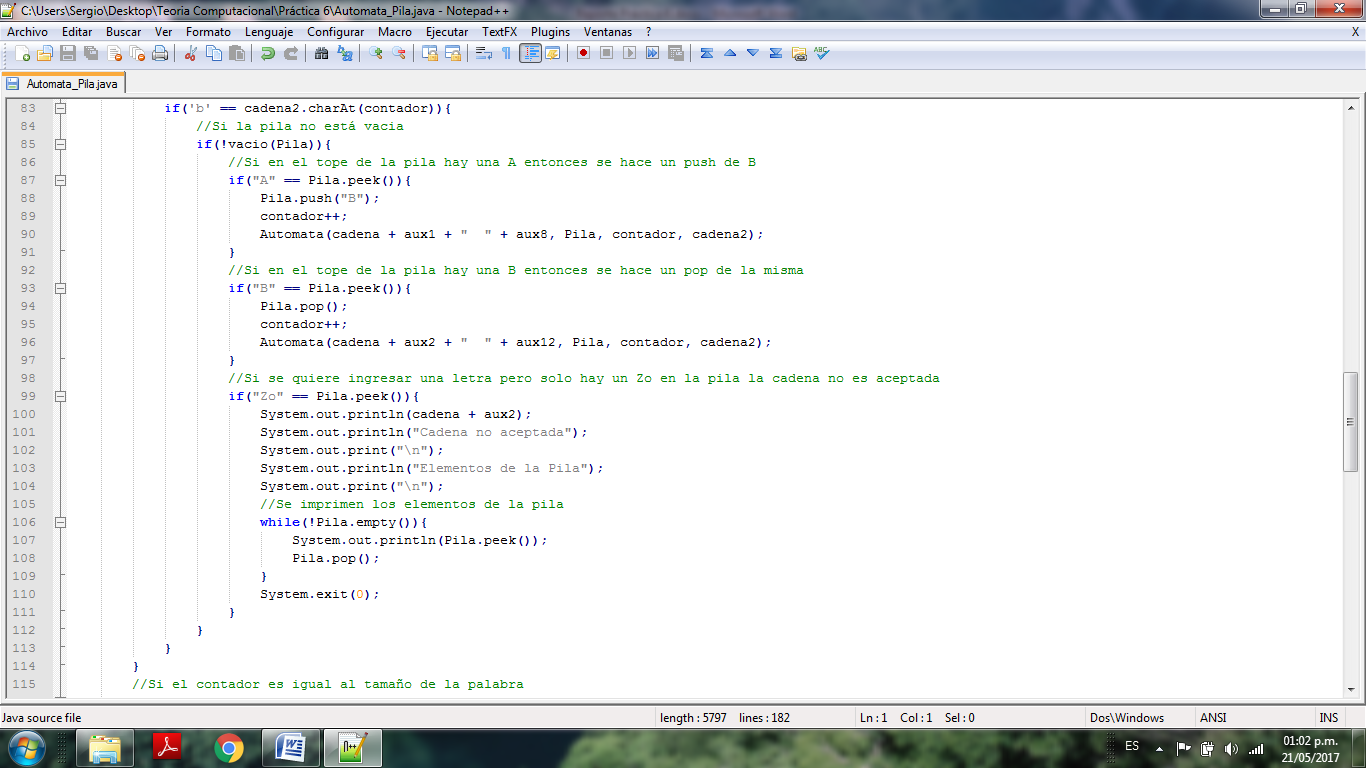
Cadena no aceptada

En este caso la cadena no es aceptada ya que la pila que con elementos los cuales obviamente hacen que la cadena no sea aceptada, en la explicación del programa se verá más a detalle este tipo de situaciones.

**Implementación de la solución en código:**

Como se explicó anteriormente, al igual que se desarrollaron algunos ejemplos, en la parte del código se hace exactamente lo mismo que se explico, solo que para poder realizar las derivaciones al igual que los elementos de la pila se utilizó recursividad para que dado las transiciones en las que iba pasando la cadena se mandaban a llamar a la mismo función y de este modo la cadena que se le mandaba ya tenía incluido la información anterior, también se utilizó la pila la cual al momento de que ya no había ninguna letra se imprimía lo que estaba en la pila y de ese modo se veía si una cadena era aceptada o no dependiendo de los valores que estén en la pila.

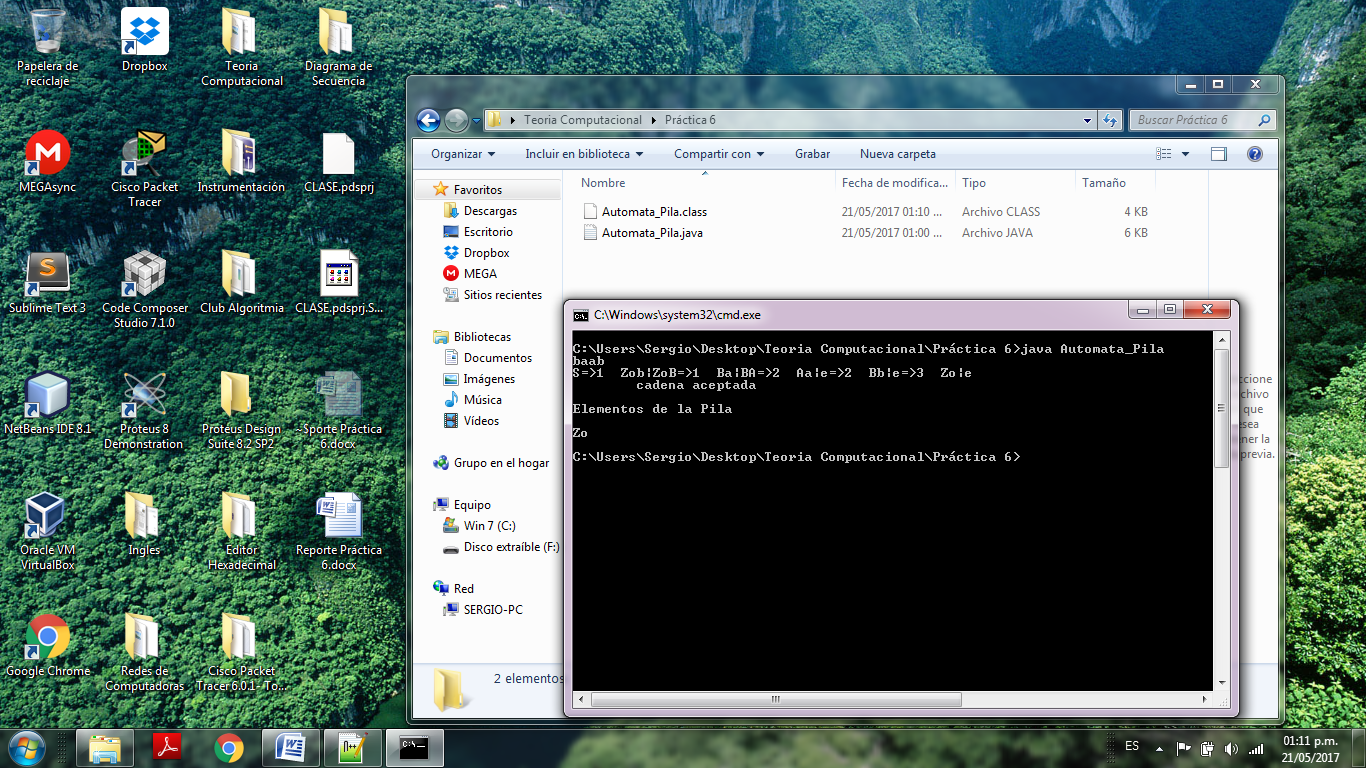




Esta es la parte del código más importante ya en toda esa parte estamos validando tanto los elementos de la pila tanto que todavía se puedan ingresar elementos a la pila de este modo solo como se dijo anteriormente si una cadena quiere ingresar más elementos a la pila pero en la pila solo tenemos “Zo” entonces no podemos y solo ponemos que la cadena no es aceptada como se puede ver en el código y también dependiendo de los elementos que tenga la pila se hará un push o un pop ya sea el caso.

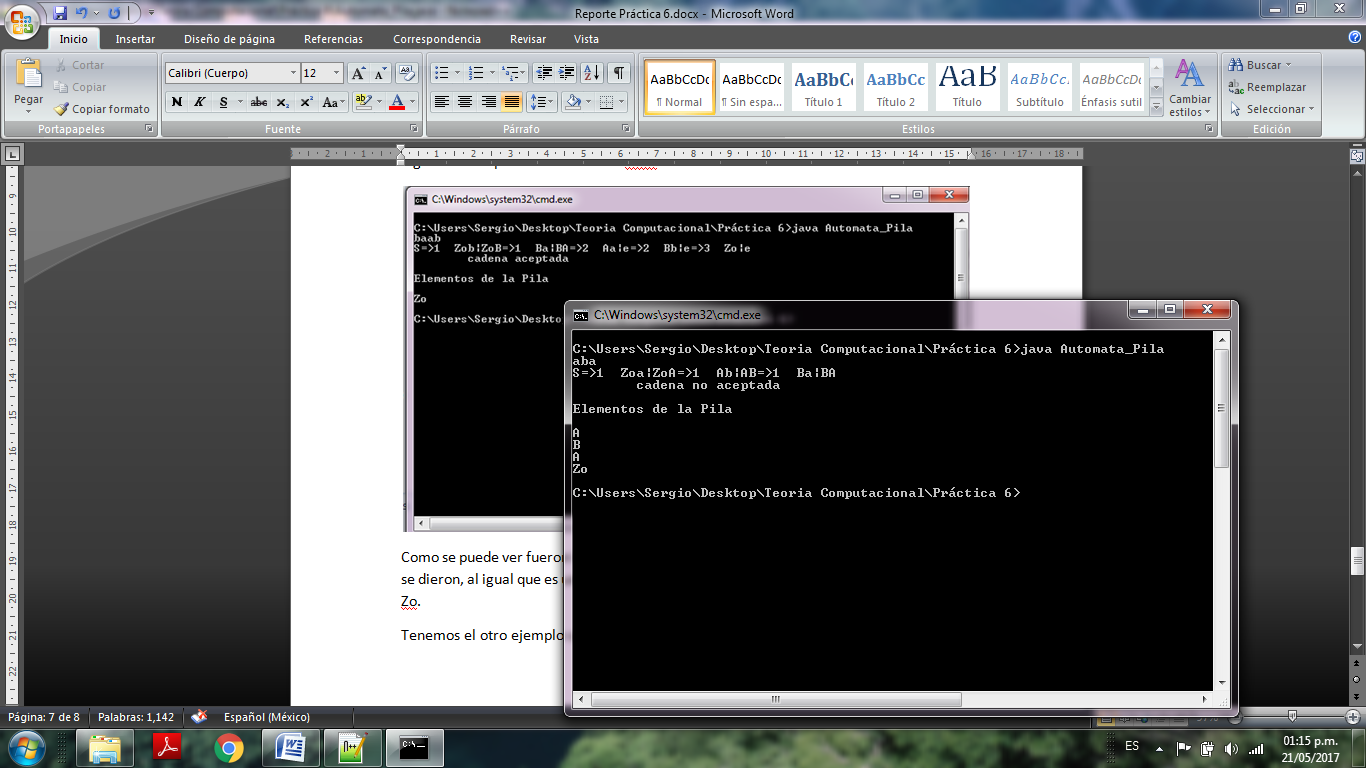
**Funcionamiento en terminal**

Ahora veremos el funcionamiento en terminal el cual nos dirá si una cadena es aceptada o no, vamos a poner los mismos ejemplos que describimos anteriormente los cuales habíamos determinado las derivaciones y los elementos que se tenían en la pila, ingresaremos primero la cadena “baab”



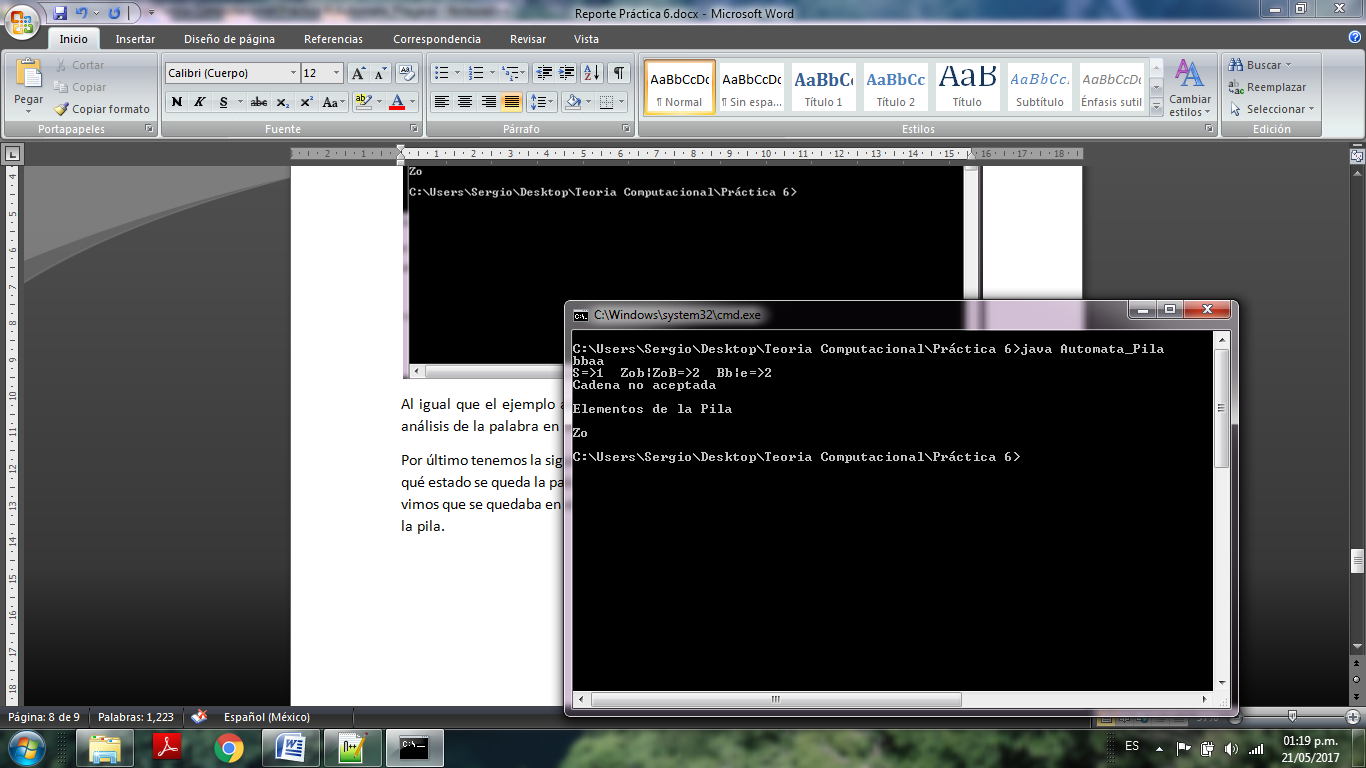
Como se puede ver fueron las mismas transiciones que se obtuvieron en los ejemplos que se dieron, al igual que es una cadena aceptada y el único elemento que tiene en la pila es Zo.

Tenemos el otro ejemplo que es la palabra “aba”, vamos a ver sus resultados.



Al igual que el ejemplo anterior fueron los mismos resultados que se obtuvieron en el análisis de la palabra en el autómata al igual que en la pila.

Por último tenemos la siguiente palabra que es “bbaa”, vamos a ver si el programa dice en qué estado se queda la palabra porque al momento de analizar la palabra en el autómata vimos que se quedaba en el esta 2 porque quería ingresar una letra y solo se tenía”Zo” en la pila.



Se comprobó que si se queda en el estado 2 por lo que el programa solo dice el estado en donde se quedó.

**Errores detectados**

1. Puede que unos de los errores sea que se ingrese una cadena muy larga y entonces posiblemente el programa no pueda con tanta información.
2. El programa no manda mensaje que solo se tiene que ingresar las letras a y b.
3. Si ingreso una palabra el programa no regresa a la parte principal.
4. Tengo que correr el programa siempre porque no hay un do while que se ejecute tantas veces quiera.

**Posibles mejoras**

Como se mencionó las mejoras que se podrían hacer es mandar un mensaje cada que se ingresen letras que no son o hacer el do while que ayude a ingresar tantas palabras quiera el usuario.

**Conclusión**

Esta práctica se me hizo interesante ya que pudimos corroborar muchos de los ejercicios que hicimos en clase con el programa y también una de las partes que me costó un poco de trabajo fue la primera parte de hacer las validaciones porque tuve que pensar tanto en los mejores como en los peores casos y fue algo tedioso.