PROYECTO COIL

VINCULACIÓN INTERNACIONAL COIL-UAEH-UDEA

EQUIPO:
MARISOL CRUZ REYES
MARCO YAHIR BALTAZAR GARCIA
LUIS ALONSO LOZADA CASTELAN
Laura Luna Amaya Otálvaro
DANILLO CHRISTIN CHAMORRO MORA
CANO GARCIA ESMERALDA YOSELIN

Índice

Introducción

El presente documento expone el desarrollo de un proyecto académico orientado a la aplicación práctica de los conceptos fundamentales de la teoría de autómatas, lenguajes formales y construcción de compiladores. A lo largo del trabajo se integran conocimientos sobre análisis léxico, análisis sintáctico, diseño de gramáticas, autómatas finitos deterministas y el uso de herramientas como Lex y Yacc para la generación de analizadores.

Asimismo, se aborda el vínculo entre estas herramientas computacionales y el Objetivo de Desarrollo Sostenible 4, enfocado en garantizar una educación inclusiva y de calidad. Desde esta perspectiva, el proyecto propone una estrategia que permite procesar, interpretar y adaptar textos que contienen regionalismos colombianos.

Finalmente, el documento presenta las etapas del desarrollo, las evidencias de ejecución, los resultados obtenidos y un planteamiento didáctico que muestra cómo estas herramientas pueden contribuir a una comprensión más efectiva de textos culturalmente diversos.

Marco teórico

Objetivo de Desarrollo Sostenible 4: Educación de calidad.

Es uno de los Objetivos de Desarrollo Sostenible de la Agenda 2030 de las Naciones Unidas, cuyo propósito es garantizar una educación inclusiva, equitativa y de calidad y promover oportunidades de aprendizaje a lo largo de toda la vida para todas las personas. Esto implica asegurar que todas las niñas y niños completen la educación primaria y secundaria, tener acceso igualitario a educación preescolar de calidad, promover la formación técnica y superior, y asegurar que todas las personas jóvenes y adultas tengan las competencias necesarias para el empleo y la vida.

Autómatas y Compiladores

El analizador léxico es la primera fase de un compilador. Su principal función consiste en leer los caracteres de entrada y elaborar como salida una secuencia de componentes léxicos que utiliza el analizador sintáctico para hacer su análisis.
El análisis léxico también es llamado exploración o scanner. Su función es lee caracteres uno a uno desde la entrada e ir formando grupos de caracteres con alguna relación entre sí llamados tokens, los que serán la entrada para la siguiente etapa del compilador.

El analizador léxico:

- Es la primera fase del programa traductor.
- Es el único que gestiona el archivo de entrada con el código fuente.
- Es el que lee los caracteres del programa fuente y construye símbolos intermedios, los cuales serán la entrada del analizador sintáctico.

El analizador léxico típicamente detecta los siguientes errores:

- Utilizar caracteres que no pertenecen al alfabeto del lenguaje.
- Cadenas que no coinciden con ninguno de los patrones de los tokens posibles.

Algunas de las funciones del análisis léxico son

- Lee caracteres.
- Produce componentes léxicos (tokens).
- Filtra comentarios.
- Filtra separadores múltiples (espacios, tabuladores y saltos de línea).
- Lleva el contador de línea y columna del texto fuente.
- Genera errores en caso de que la entrada no corresponda a ninguna categoría léxica.

Para realizar dichas funciones, es importante tomar en cuenta los siguientes conceptos:

- **Categoría léxica.** Tipo de símbolo elemental del lenguaje fuente (identificadores, palabras clave, constantes numéricas, operadores, ...).
- **Componente léxico (token).** Elemento perteneciente a una categoría léxica.
- **Lexema.** Cadena de caracteres correspondiente al componente léxico.

Sintaxis y Gramáticas en los Lenguajes de Programación

Todo lenguaje de programación está regido por un conjunto de reglas que determinan la estructura sintáctica de los programas bien formados. La descripción de la sintaxis de estas construcciones puede realizarse mediante Gramáticas Libres de Contexto o utilizando la Notación BNF (*Backus-Naur Form*). Estas opciones ofrecen ventajas para los diseñadores de lenguajes, así como a los desarrolladores de compiladores, ya que constituyen especificaciones sintácticas claras y precisas.

Una gramática permite la generación automática de analizadores sintácticos, contribuyendo a la detección de ambigüedades durante el proceso de construcción. Al proporcionar una estructura formal al lenguaje de programación, facilita la generación de código y la identificación de errores. Además, la descripción de un lenguaje mediante una gramática simplifica su ampliación o modificación futura.

Analizador Sintáctico

El analizador sintáctico constituye una etapa esencial en el proceso de compilación, encargándose de verificar que el texto de entrada cumpla con las reglas definidas por la gramática correspondiente. Si el programa de entrada es válido, el analizador genera un árbol sintáctico que representa su estructura. En teoría, la salida del analizador consiste en una representación formal del árbol sintáctico basada en la secuencia de tokens proporcionada por el analizador léxico.

En la práctica, el analizador sintáctico suele realizar tareas adicionales, como:

- Acceso a la tabla de símbolos para asistir en las funciones del analizador semántico.
- Verificación de tipos, propia del análisis semántico.
- Generación de código intermedio.
- Manejo y generación de mensajes de error.

Manejo de Errores Sintácticos

El diseño de un compilador sería significativamente más sencillo si solo tuviera que procesar programas correctos. Sin embargo, en la práctica, los programadores suelen cometer errores, por lo que un buen compilador debe ser capaz de identificar, localizar y comunicar estos errores de manera efectiva. Incorporar estrategias de manejo de errores desde las etapas iniciales del desarrollo de un compilador no solo simplifica su estructura, sino que también mejora su capacidad para responder a los errores detectados.

Los errores que pueden surgir durante el proceso de programación incluyen:

- **Errores léxicos.** Escribir incorrectamente un identificador, palabra clave u operador.
- **Errores sintácticos.** Derivados de expresiones mal formadas, como paréntesis desbalanceados o operadores mal ubicados.
- **Errores semánticos.** Aplicar un operador a un operando incompatible.
- **Errores lógicos.** Códigos con funciones recursivas infinitas.

El manejo de errores sintácticos es especialmente complejo en la creación de compiladores. Un buen analizador sintáctico debe cumplir con los siguientes objetivos:

1. Reportar los errores de forma clara y precisa.
2. Especificar el tipo de error y su ubicación.
3. Recuperarse del error para continuar con el análisis del programa.
4. Minimizar el impacto en el rendimiento de la compilación.

Por lo tanto, un compilador robusto debe diseñarse considerando los errores potenciales, lo que no solo mejora su funcionalidad, sino que también garantiza una respuesta más eficiente ante los errores detectados.

Autómata Finito Determinista (AFD)

Un *Autómata Finito Determinista (AFD)* se define como una tupla:

$$AFD = (\Sigma,\ Q,\ \delta,\ q_0,\ F)$$

- Σ es el alfabeto de entrada
- Q es el conjunto finito y no vacío de los estados
- δ es la función de transición, $\delta: Q \times \Sigma \rightarrow Q$
- $q_0 \in Q$ es el estado inicial
- $F \subset Q$ es un conjunto de estado finales de aceptación $(F \neq \emptyset)$

Sea $A = (\Sigma,\ Q,\ \delta,\ q_0,\ F)$ un AFD y sea $w = w_1 w_2 \ldots w_n$ una cadena de símbolos donde $w_i \in \Sigma$.

Entonces, A ***acepta*** *w* si existe una secuencia de estados $r_0, r_1, \ldots, r_n \in Q$ con tres condiciones:

1. $r_o = q_0$
2. $\delta(r_i, w_i + 1) = r_i + 1\ para\ i = 0,1, \ldots, n-1$
3. $r_n \in F$

La condición 1. Establece que el AFD comienza en el estado inicial.

La condición 2. Establece que el AFD cambia desde un estado hacia otro estado de acuerdo con la función de transición.

La condición 3. Establece que el AFD acepta la cadena de entrada si termina en un estado de aceptación. Entonces, A ***reconoce el lenguaje*** L si $L = \{w | A\ acepta\ w\}$.

Una **Tabla de transición de estados** es una tabla que muestra a qué estado se moverá un autómata finito, basándose en el estado actual y las entradas. Una tabla de estados es esencialmente una tabla de verdad en la cual algunas de las entradas son el estado actual, y las salidas incluyen el siguiente estado, junto con otras salidas.

Generalmente se asocia con cada autómata un **Diagrama de transición de estados**. Cada nodo del diagrama corresponde a un estado. El estado inicial se indica mediante una flecha que no tiene nodo origen. Los estados finales se representan con un círculo doble. Si existe una transición del estado e_i al estado e_j para un símbolo de entrada *a*, existe entonces un arco rotulado con *a* desde el nodo e_i al nodo *ej*; es decir que $\delta(e_i, a) = e_j$, se representa en el diagrama como se muestra en la Figura 1.

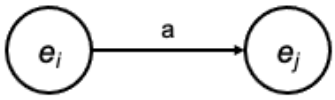


Figura 1. Diagrama de transición.

Sobre la extracción de información de textos, minería de textos…(procesamiento del lenguaje natural (PLN))

Preprocesamiento del texto

- Tokenización
- Eliminación de palabras vacías (stopwords)
- Normalización

Extracción de palabras clave

Nubes de palabras (Word Clouds)

Bibliotecas de PLN en Python: NLTK, spaCy, scikit-learn.

Análisis de sentimientos

Clasificación de texto

Objetivos

Promover la inclusión educativa mediante el diseño de experiencias de aprendizaje diferenciadas que consideren las jergas lingüísticas de los estudiantes, apoyándose en herramientas tecnológicas de reconocimiento y análisis léxico computacional.

Objetivos Específicos

1. Construir el documento fuente siguiendo las especificaciones para la producción del texto.
2. Implementar un analizador léxico y sintáctico utilizando Lex y Yacc, capaz de identificar, clasificar y validar los regionalismos colombianos y las expresiones emocionales presentes en un documento fuente.
3. Desarrollar un sistema de procesamiento lingüístico que traduzca el documento fuente a un lenguaje comprensible para estudiantes iberoamericanos, reconociendo automáticamente las emociones predominantes y generando una nube de palabras representativa.
4. Construir un prototipo funcional que integre las herramientas tecnológicas desarrolladas, mostrando la ejecución, los resultados y la documentación del código, en un entorno visual aplicable a contextos educativos.
5. Esbozar la implementación de los resultados para favorecer estrategias de atención diferenciada en ambientes educativos.

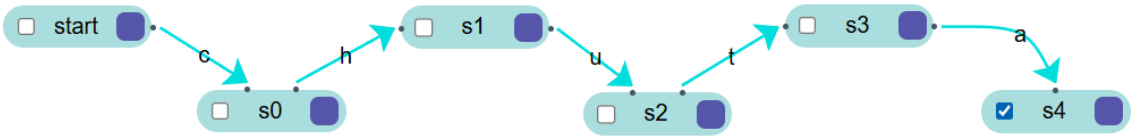
Desarrollo

1. **Análisis Léxico**: **Autómatas empleados** en el para reconocer los tokens requeridos para las resoluciones.

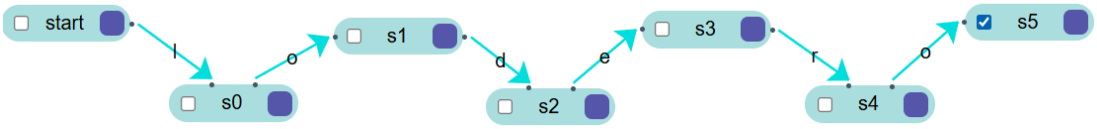
$$AFD = (\Sigma,\ Q,\ \delta,\ q_0,\ F)$$

- Σ es el alfabeto de entrada
- Q es el conjunto finito y no vacío de los estados
- δ es la función de transición, $\delta: Q \times \Sigma \rightarrow Q$
- $q_0 \in Q$ es el estado inicial
- $F \subset Q$ es un conjunto de estado finales de aceptación $(F \neq \emptyset)$

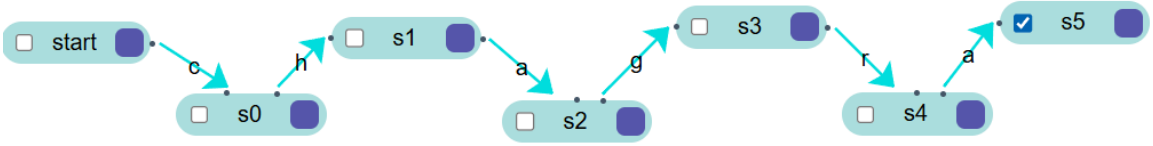
Sea $A = (\Sigma,\ Q,\ \delta,\ q_0,\ F)$ un AFD y sea $w = w_1 w_2 \ldots w_n$ una cadena de símbolos donde $w_i \in \Sigma$.



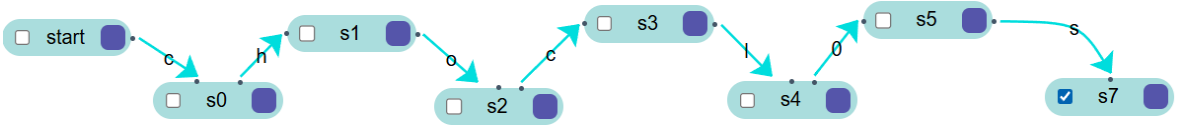
chuta: suerte



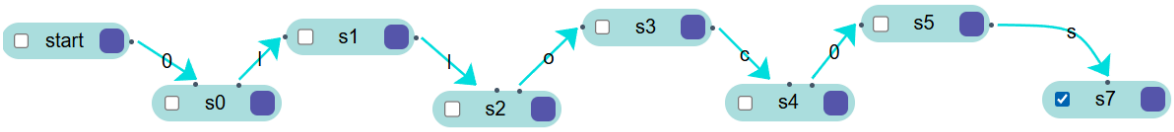
lodero: desordenado



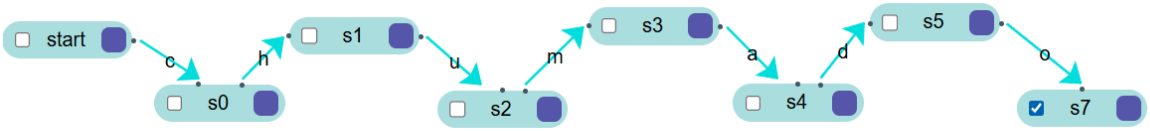
chagra: campesino



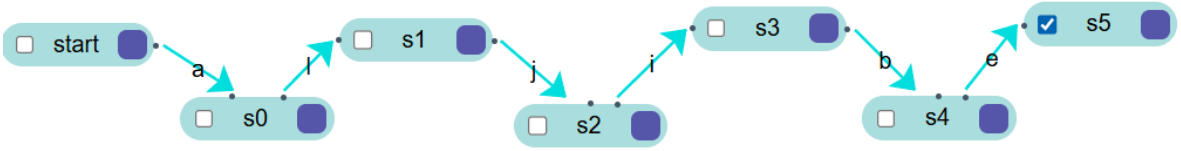
choclos: maiz



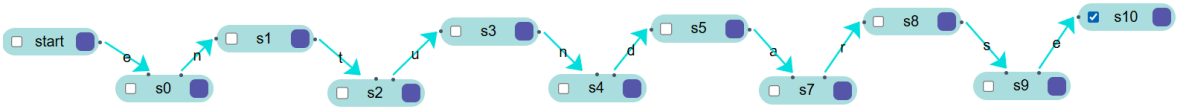
ollocos: papas



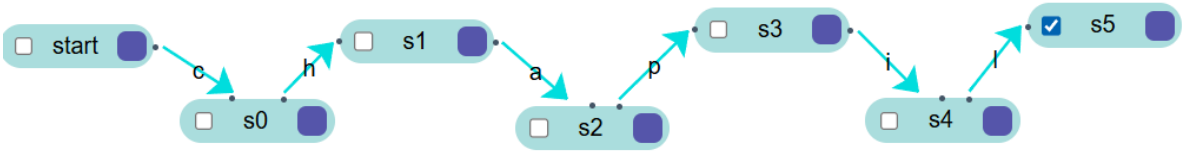
chumado:borracho



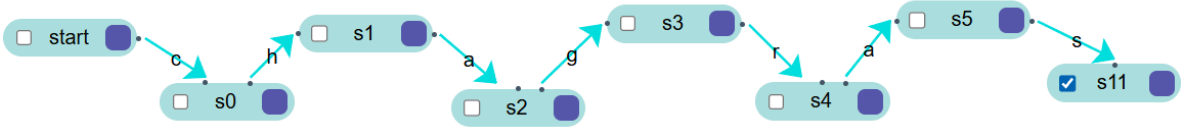
aljibe:depósito de agua



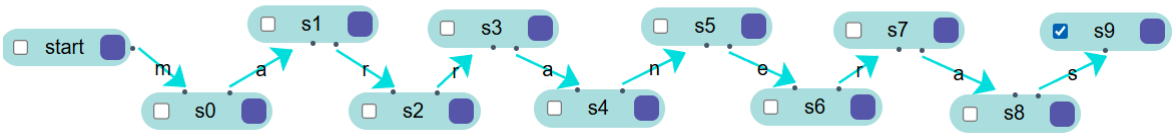
entundarse:confundirse



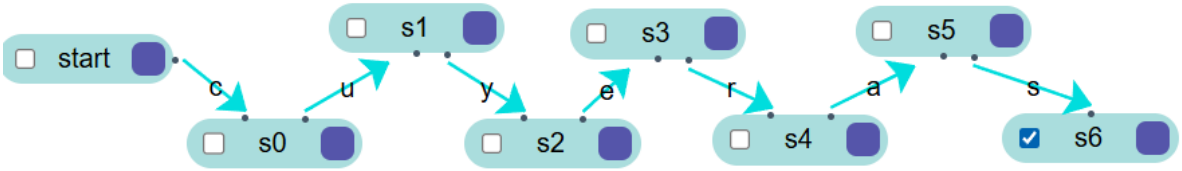
chapíll:sombrero



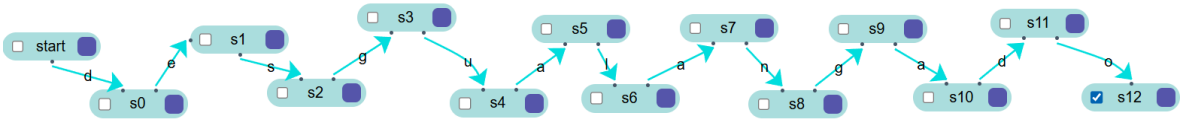
chagras:campesinos



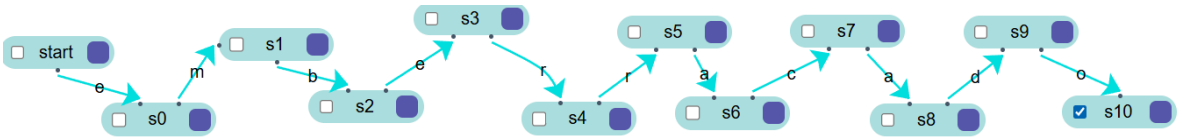
marraneras: porquerizas



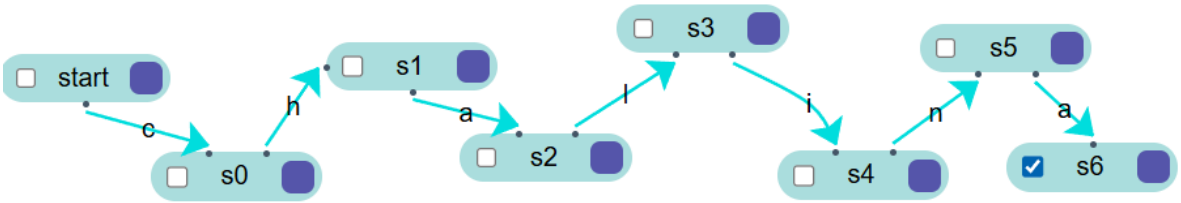
cuyeras:conejeras



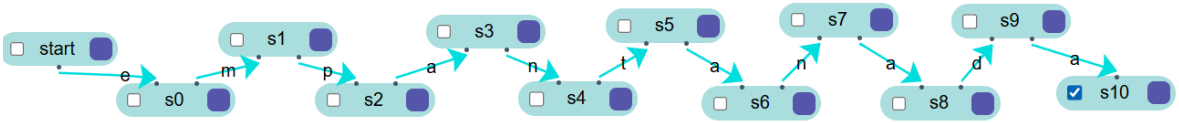
desgualangado: desgarbado



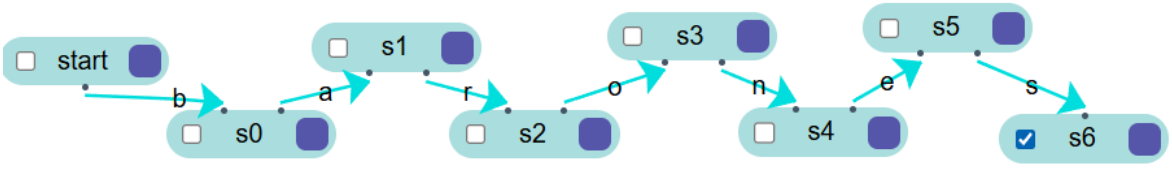
emberracado: enfadado



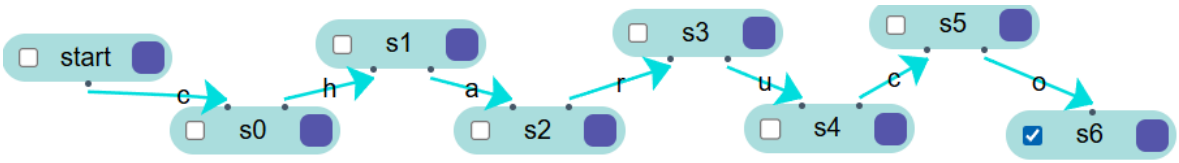
chalina:bufanda



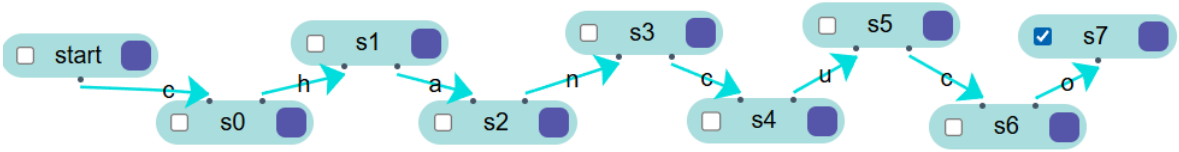
empantanada:estancada



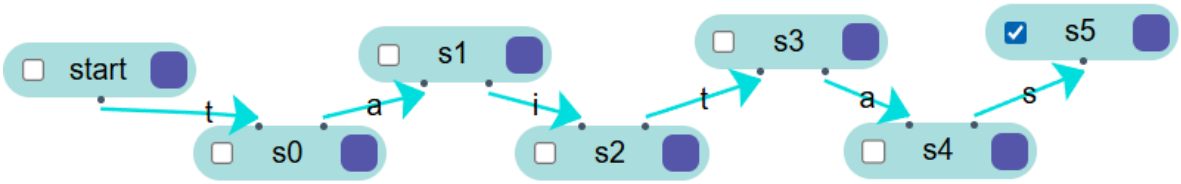
barones:seniores



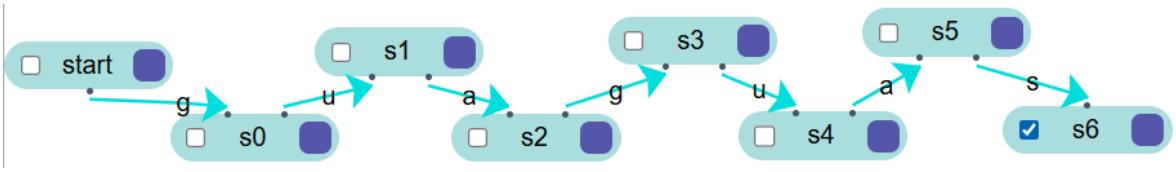
charuco:terreno



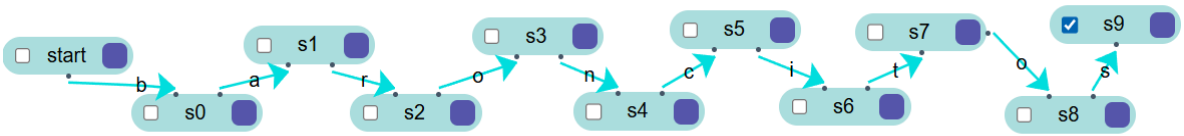
chancuco:dulce



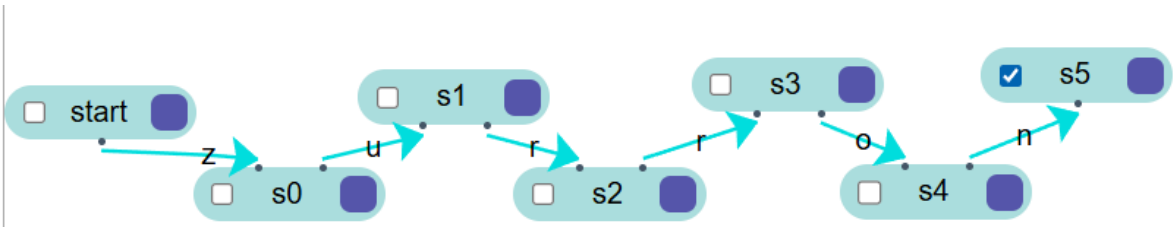
taitas:padres



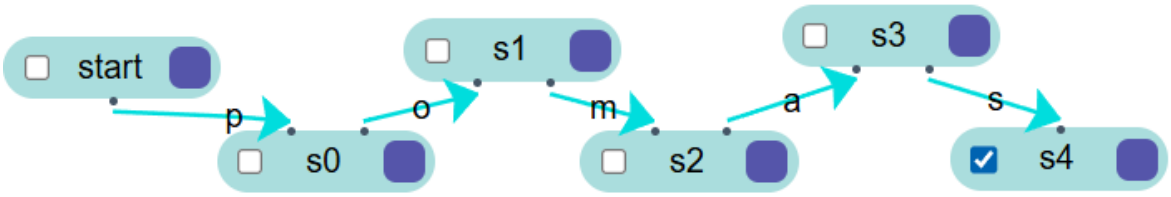
guaguas:niños



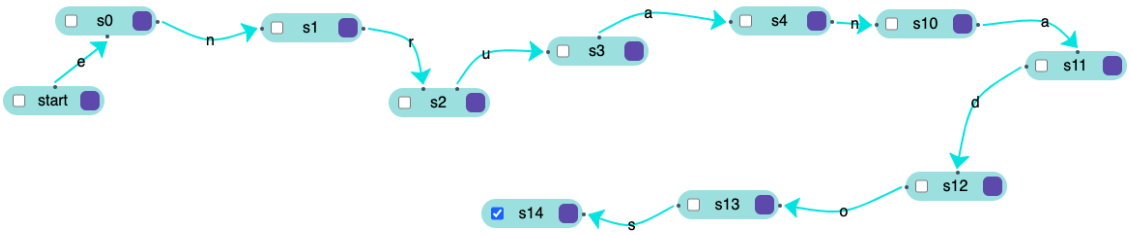
baroncitos:jovenes



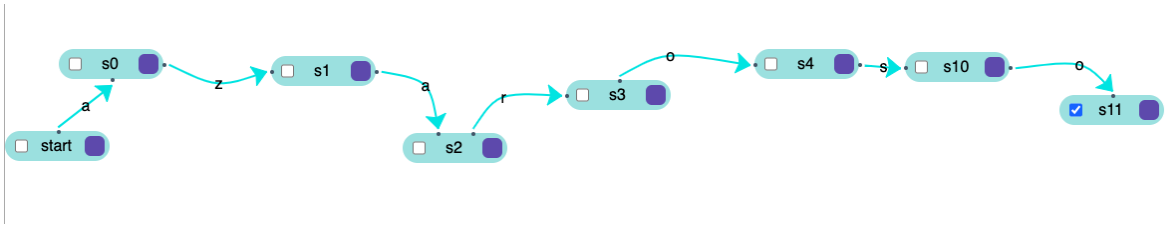
zurrón:mochila



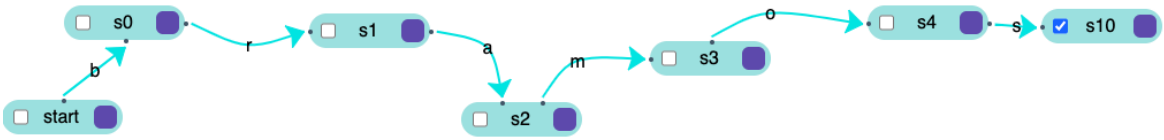
pomas:manzanas



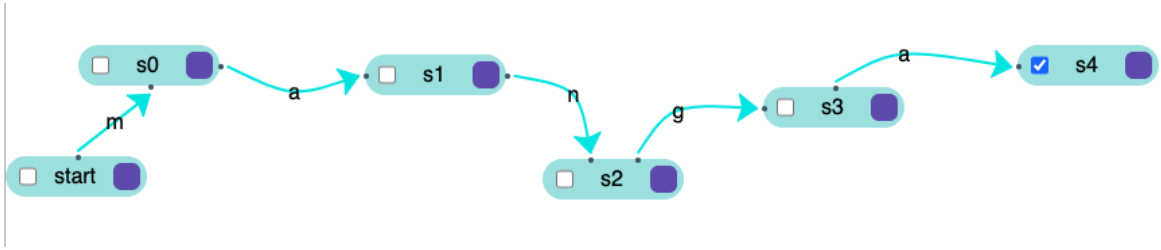
enruanados:abrigados



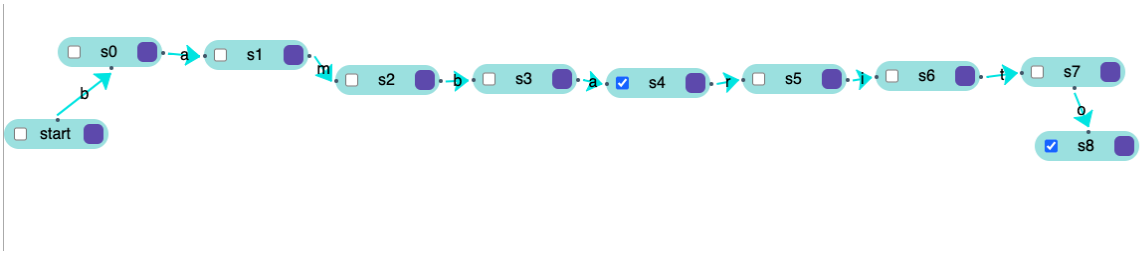
azaroso:arriesgado



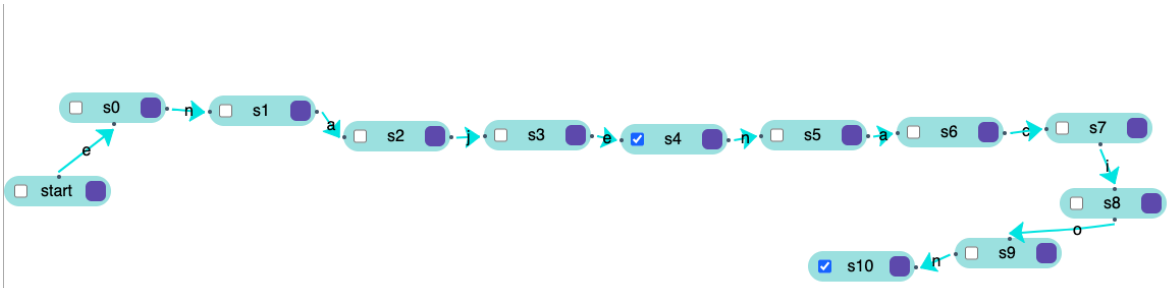
bramos:grito



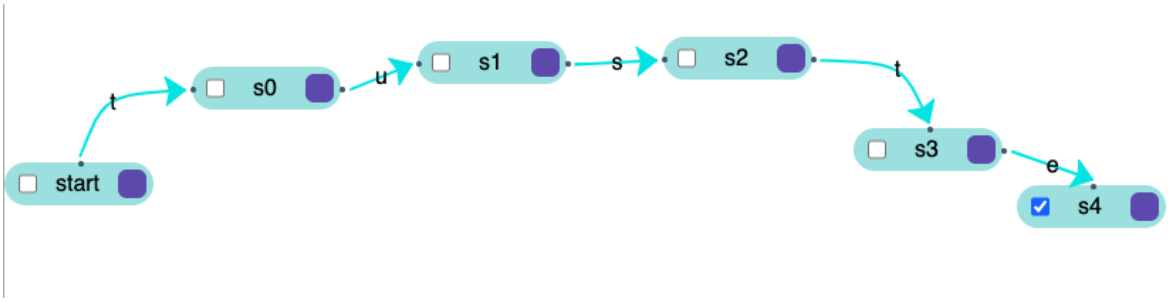
manga:grupo



bambarito:elegante



enajenación:locura



tuste:golpe

2. **Análisis Sintáctico: Gramáticas regulares** utilizadas

$$G = \{\sum_T, \sum_N, S, P\}$$

donde:

- $\sum_T$ es un alfabeto de símbolos terminales
- $\sum_N$ es un alfabeto de símbolos no terminales
- S es el símbolo inicial de la gramática
- P es un conjunto de producciones gramaticales

Además, se cumple:

$$S \in \sum_N$$

$$\sum_T \cap \sum_N = \emptyset$$

$$\sum = \sum_T \cup \sum_N$$

La gramática formal G permite generan un lenguaje $L = \{x \in \sum_T^* | S \ast\rightarrow x\}$

Las **expresiones regulares** permiten describir con exactitud y sencillez cualquier lenguaje regular.

3. **Notación Extended Bakus Naur Form:**

La forma Backus-Naur extendida (EBNF) es un tipo de sintaxis formal utilizada para especificar la estructura de un lenguaje de programación u otro lenguaje formal. Es una extensión de la forma Backus-Naur (BNF), desarrollada originalmente por John Backus y Peter Naur para describir la sintaxis del lenguaje de programación Algol.

EBNF añade varios metasímbolos adicionales a los originales de BNF, lo que permite una especificación más concisa y legible de la sintaxis de un lenguaje. Se utiliza habitualmente en la especificación de lenguajes de programación y, en ocasiones, para describir la sintaxis de otros tipos de lenguajes formales, como los lenguajes de consulta de bases de datos o los lenguajes de marcado.

Elementos básicos de la EBNF

1. **Símbolos terminales**:
   Representan elementos atómicos del lenguaje (tokens). Se escriben entre comillas.
   Ejemplo: "if", "+", "123".
2. **Símbolos no terminales**:
   Representan estructuras sintácticas compuestas. Se escriben sin comillas.
   Ejemplo: expresión, declaración.
3. **Reglas de producción**:
   Definen cómo se construyen los no terminales. Su formato es: no_terminal = expresión ;

Operadores y Constructores en EBNF

Operador	Significado	Ejemplo
=	Definición de una regla.	dígito = "0" | "1" ;
|	Alternativa (uno de varios).	letra = "A" | "B" | "C" ;
{ ... }	Repetición (0 o más veces).	números = { dígito } ;
[...]	Opcional (0 o 1 vez).	signo = ["+" | "-"] ;
(...)	Agrupación.	operación = ("+" | "-") término ;
"..."	Terminal (símbolo literal).	"while"
'...'	Equivalente a "...".	'='
(* ... *)	Comentarios (ignorados en la sintaxis).	(* Esto es un comentario *)

Ejemplos de EBNF

Números enteros con signo opcional:

```
entero = [ "+" | "-" ] , dígito , { dígito } ;
dígito = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;
```

Sentencia if-else

```
sentencia_if = "if" , "(" , condición , ")" , sentencia , [ "else" , sentencia ] ;
condición = expresión ;
sentencia = ... ; (* Otras reglas para sentencias *)
```

4. **Traducción**

Programa de LEX para el análisis léxico del documento fuente (tokenización)

Evidencias de ejecución desde línea de comandos

```
- ┬íEugenia!, ┬íEugenia! ┬íSali, pero ya! -Grito  - Creyo no tener respuesta, por eso golpeo y golpeo la maldita puerta
colonial, hasta casi atravesarla.
A los segundos, salio una mujer canosa, con la trompa enfurecida. Era ella, la duenia de la casa, se encontraba incorpor
andose ya, poniendose la  en el paso de la sequia, empezando a mirar la expresion aterrada y  de Carlos.
- ┬┐Que paso, ve, loco? - pregunto Eugenia - frunciendo el cenio.
- Manuel lo mato, mato a su hermano- respondio Carlos -
Eugenia se desplomo enseguida por la noticia, uno de sus hijos habia muerto.

Ya de noche, fuera de los hogares del caserio, solo permanecian  deseosos por tener un encuentro sucio con el ,  o .
Era una noche de sabado, como cualquier otra.  y , no tan , mas bien  ya, corrian ligerito a los juegos de azar, de una
pequenia cantina. Donde el zurr├│n del Bartolo, el cantinero.
Pasado un par de horas y un buen de , se sentaron en la mesa de naipe para jugar caida, cuatro . Dos de ellos eran Manue
l y Olimpo.
Juego tras juego y trago tras trago, la razon se volvia cada vez mas opaca. De pronto Manuel solto un comentario  sobre
el mucho dinero que tenia. A lo que Olimpo no dudo ni un momento en demostrar su disgusto. Con prepotencia se agarraron
a insultos. La discusion se iba avivando y con ello el danio fisico era inminente. Olimpo al ver que la pelea era obvia,
 : ÔÇ£si vamos a pelear, hagamoslo a , como los , ÔÇØ
A lo que Manuel respondio, con un silencio pesado y una mirada de enajenaci├│n. Para despues desenfundar su revolver, ap
untar fijamente en el  a Olimpo y darle fin a su hermano.
Eugenia recobrando el sentido, que poco de este tenia lo que habia pasado con sus hijos, se dirigio a el lugar mas horri
ble que pudo alguna vez imaginar. Ahi estaban los dos, el uno estupefacto, petrificado, y el otro en una escena perturba
dora de asesinato, asesinado.
No fue capaz de permanecer viendo esa animalidad. Queria pensar, solo pensar, no queria sentir, se derrumbaria en el mom
ento que lo hiciera.
Despues del sepelio Eugenia se encontro en una vuelta: Bien entregaba a Manuel a las autoridades, condenandose a la ause
ncia de sus dos hijos o bien no lo hacia y permitia en su casa de terminaciones coloniales, un asesino familiar.
```

```
C:\Flex Windows\EditPlusPortable>cd lexico

C:\Flex Windows\EditPlusPortable\lexico>flex proyecto.l

C:\Flex Windows\EditPlusPortable\lexico>cc lex.yy.c

C:\Flex Windows\EditPlusPortable\lexico>proyecto.exe
Carlos, dejando caer el  en un , paso corriendo por medio de entre una  de , habas y . Iba , por el letargo de la noche,
 pero no mucho, no tanto como para hundirse en un  o  en algun callejon, pero si lo suficiente como para dejar atras el
 de panio nuevo, que traia.
Su afan de correr posiblemente era uno de los tantos efectos del chap├¡l, pero eso no era lo unico, llevaba consigo una
razon, un acontecimiento, que minutos atras lo hubo de haber marcado para siempre. Paso por , terrenos arados y hasta un
as cuantas  y  de un paisano. Y tanto hacerle, a su paso, llego a la calle principal y unica del caserio. Llego al frent
e de una casa con terminados coloniales y muy bellos a diferencia de el que llego limpio .
interrumpio, con la avidez de un adolescente, el silencio que cobijaba las casas. Estaba de frente a la casa de su herma
na:
- ┬íEugenia!, ┬íEugenia! ┬íSali, pero ya! -Grito  - Creyo no tener respuesta, por eso golpeo y golpeo la maldita puerta
colonial, hasta casi atravesarla.
A los segundos, salio una mujer canosa, con la trompa enfurecida. Era ella, la duenia de la casa, se encontraba incorpor
```

```
===== RESULTADOS =====
chuta           : 2
lodero          : 1
chagra          : 1
choclos         : 1
ollocos         : 1
chumado         : 1
aljibe          : 1
entundarse      : 1
chapil          : 1
chagras         : 1
marraneras      : 1
cuyeras         : 1
desgualangado   : 1
emberracado     : 1
chalina         : 1
empantanada     : 1
barones         : 2
charuco         : 1
chancuco        : 1
Taitas          : 1
guaguas         : 2
baroncitos      : 1
pomas           : 1
enruanados      : 1
azaroso         : 1
bramo           : 1
manga           : 1
bambarito       : 1

Total de regionalismos colombianos encontrados: 31
```

5. **Eliminación de palabras vacías (stopwords)**

Programa de LEX para el análisis léxico del documento fuente

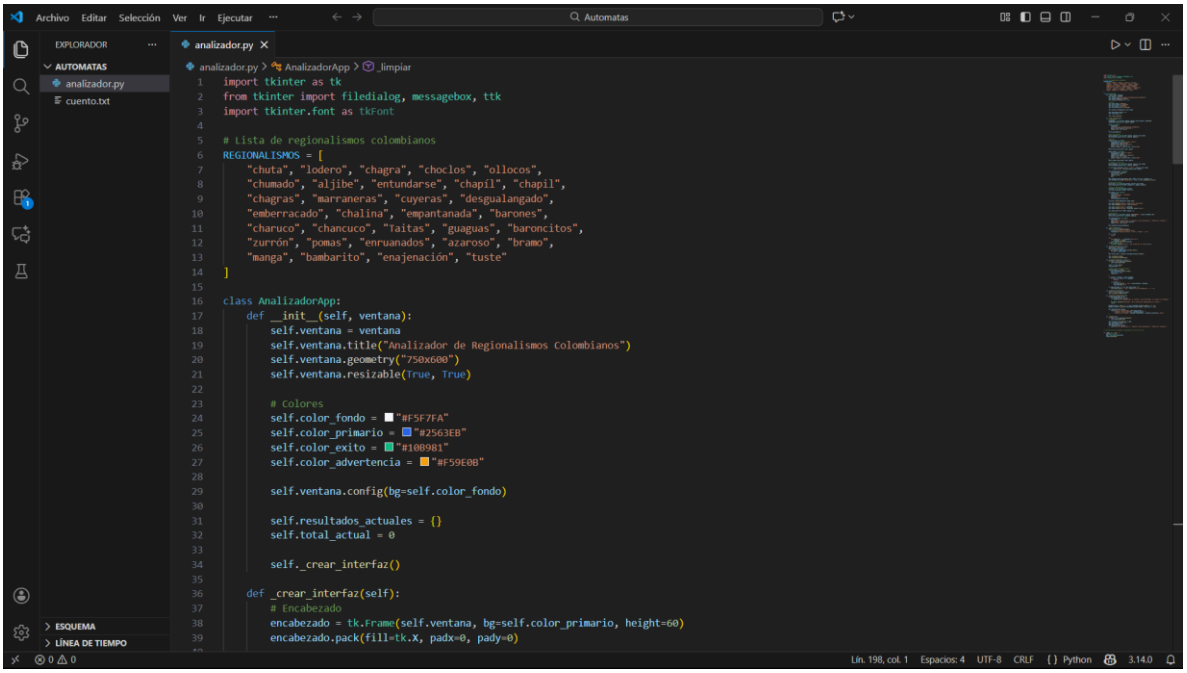
```
C:\Users\canoe>cd "C:\Flex Windows\EditPlusPortable\lexico"

C:\Flex Windows\EditPlusPortable\lexico>lex PalVacia.l

C:\Flex Windows\EditPlusPortable\lexico>cc lex.yy.c

C:\Flex Windows\EditPlusPortable\lexico>PalVacia.exe
Eliminando palabras vacÝas...

carlos dejando caer chuta lodero paso corriendo medio chagra choclos habas ollocos chumado letargo noche mucho tanto hun
dirse aljibe entundarse algun callejon suficiente dejar atras chuta panio nuevo traia afan correr posiblemente uno tanto
s efectos chapÝl unico llevaba consigo razon acontecimiento minutos atras hubo marcado paso chagras terrenos arados cuan
tas marraneras cuyeras paisano tanto hacerle paso llego calle principal unica caserio llego frente casa terminados colon
iales bellos diferencia llego limpio desgualangado interrumpio avidez adolescente silencio cobijaba casas frente casa he
rmana eugenia eugenia sali grito emberracado creyo respuesta golpeo golpeo maldita puerta colonial atravesarla segundos
salio mujer canosa trompa enfurecida duenia casa encontraba incorporandose poniendose chalina paso sequia empezando mira
r expresion aterrada empantanada carlos paso ve loco pregunto eugenia frunciendo cenio manuel mato mato hermano respondi
o carlos eugenia desplomo enseguida noticia uno hijos habia muerto noche fuera hogares caserio permanecian barones deseo
sos encuentro sucio chapil charuco chancuco noche sabado cualquier otra taitas guaguas guaguas mas bien baroncitos corri
an ligerito juegos azar pequenia cantina zurr%n bartolo cantinero pasado par horas buen pomas sentaron mesa naipe jugar
caida cuatro enruanados dos eran manuel olimpo juego juego trago trago razon volvia cada vez mas opaca pronto manuel sol
to comentario azaroso mucho dinero tenia olimpo dudo momento demostrar disgusto prepotencia agarraron insultos discusion
 avivando ello danio fisico inminente olimpo pelea obvia bramo vamos pelear hagamoslo manga barones bambarito manuel res
pondio silencio pesado mirada enajenaci%n despues desenfundar revolver apuntar fijamente tuste olimpo darle fin hermano
eugenia recobrando sentido poco tenia habia pasado hijos dirigio lugar mas horrible pudo vez imaginar ahi dos uno estupe
facto petrificado otro escena perturbadora asesinato asesinado capaz permanecer viendo animalidad queria pensar pensar q
ueria sentir derrumbaria momento hiciera despues sepelio eugenia encontro vuelta bien entregaba manuel autoridades conde
nandose ausencia dos hijos bien permitia casa terminaciones coloniales asesino familiar
```

6. **Extracción de palabras clave**

Programa de LEX para el análisis léxico del documento fuente (normalización y obtención de palabras para Word Cloud)

Programa de YACC para el análisis sintáctico del documento fuente

7. Integración con Python / similar para generar representación visual de la **Word Cloud (**Bibliotecas de PLN en Python / similar)



¿QUÉ ES UNA WORD CLOUD?

Una nube de palabras (Word Cloud) es una representación viEsual de la frecuencia o relevancia de las palabras dentro de un texto.

Las palabras que aparecen con mayor tamaño son aquellas que ocurren con más frecuencia o tienen mayor peso.

INTEGRACIÓN CON PYTHON

Python es el lenguaje más popular para generar nubes de palabras porque combina:

1. **Bibliotecas de PLN (procesamiento lingüístico).**

Estas bibliotecas se usan para procesar y limpiar el texto antes de generar la nube de palabras. Permiten tareas como:

- **Tokenización:** Dividir el texto en palabras individuales.
- **Eliminación de palabras vacías (stop words):** Descartar palabras comunes (p. ej., "el", "la", "y") que no aportan significado para la visualización.
- **Lematización/Stemming:** Reducir las palabras a su raíz o forma base.

Las bibliotecas clave son:

- **NLTK (Natural Language Toolkit):** Una biblioteca muy completa y ampliamente utilizada que proporciona interfaces fáciles de usar para más de 50 corpus y recursos léxicos, junto con un conjunto de bibliotecas de procesamiento de texto.
- **spaCy:** Una biblioteca que ofrece un rendimiento superior y modelos preentrenados más rápidos para tareas como el etiquetado gramatical y el reconocimiento de entidades nombradas. Es una excelente alternativa a NLTK, especialmente para aplicaciones en producción.

2. **Herramientas de visualización gráfica.**

Wordcloud: Esta biblioteca es la herramienta estándar para generar la imagen de la nube de palabras. El tamaño de cada palabra en la visualización indica su frecuencia o importancia dentro del texto analizado.

Otras bibliotecas de visualización que se usan en conjunto con wordcloud incluyen:

- Matplotlib: Se utiliza para mostrar la imagen generada por la biblioteca wordcloud y personalizar aspectos como eliminar los ejes del gráfico.

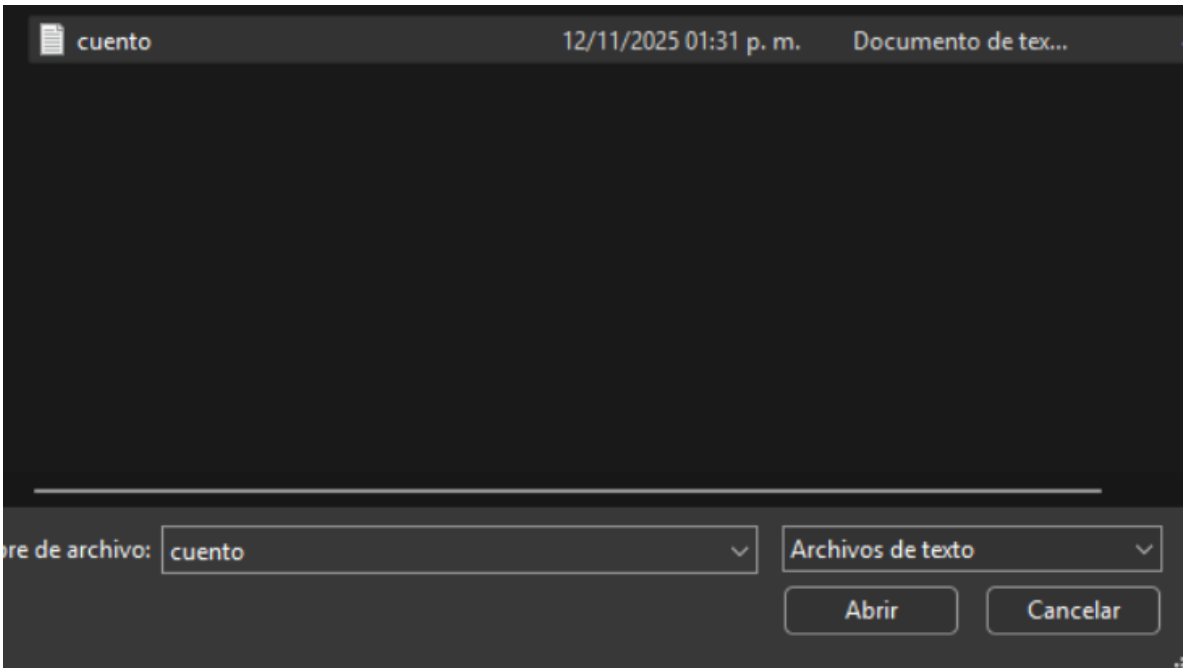
- Pillow (PIL tenedor): A menudo se requiere como dependencia para manejar las operaciones de imagen.

PROCESO DE INTEGRACIÓN

Un flujo de trabajo común implica estos pasos:

1. **Cargar y Preprocesar los Datos:** Usar bibliotecas de PLN (como NLTK o spaCy) para limpiar el texto, eliminar palabras vacías y lematizarlo.
2. **Preparar el Texto:** Unir el texto procesado en una sola cadena de texto grande, que es el formato que espera la biblioteca wordcloud.
3. **Generar la Nube de Palabras:** Utilizar la clase WordCloud de la biblioteca para crear una instancia y generar la nube, personalizando parámetros como el color de fondo, la forma (usando máscaras de imagen), y las dimensiones.
4. **Visualizar:** Usar Matplotlib para mostrar la imagen resultante.

8. Desarrollo de la interfase gráfica


cuento
12/11/2025 01:31 p. m.
Documento de tex...
re de archivo:
cuento
Archivos de texto
Abrir
Cancelar


Analizador de Regionalismos Colombianos
Analizador de Regionalismos Colombianos
Cargar archivo
Limpiar
Filtrar:
Palabra encontrada
Cantidad
chuta 2
chagra 2
barones 2
guaguas 2
lodero 1
choclos 1
ollocos 1
chumado 1
aljibe 1
entundarse 1
chapil 1
chapil 1
chagras 1
marraneras 1
cuyeras 1
desgualangado 1
Total de regionalismos: 36 | Palabras únicas encontradas: 32 | Palabra más frecuente: 'chuta' (2 veces)


Analizador de Regionalismos Colombianos
Analizador de Regionalismos Colombianos
Cargar archivo
Limpiar
Filtrar:
ch
Palabra encontrada
Cantidad
chuta 2
chagra 2
choclos 1
chumado 1
chapil 1
chapil 1
chagras 1
chalina 1
charuco 1
chancuco 1
Total de regionalismos: 36 | Palabras únicas encontradas: 32 | Palabra más frecuente: 'chuta' (2 veces)

Resultados

1. Documento objeto (desde la terminal después de ejecutados en conjunto los programas de Lex y Yacc, con la “traducción” ya sin jerga colombiana)
2. Herramienta tecnológica desarrollada (desde donde se ejecuta automáticamente los programas desarrollados, en un entorno visual) útil para su uso en entornos educativos (que permita descargar un pdf del documento objeto).
3. Pantallas de la herramienta y evidencias de ejecución de la traducción
4. Pantallas de la herramienta y evidencias de ejecución de la extracción de las Stop Words
5. Pantallas de la herramienta y evidencias de ejecución de las palabras clave (cuáles y cuántas, puede ser desde la línea de comandos)
6. Herramienta tecnológica desarrollada (desde donde se ejecuta automáticamente los programas desarrollados, en un entorno visual) útil para su uso en entornos educativos.
7. Pantallas de la herramienta y evidencias del Análisis de sentimientos y Word cloud
8. Esbozo básico del planteamiento didáctico para emplear los resultados obtenidos.
9. Descripción general de la aplicación práctica en estrategias de atención diferenciada.

NOTA: Documentar los resultados mediante capturas de pantalla y descripciones de estas.

Conclusiones

Durante este proyecto, pudimos llevar a cabo todos los conocimientos adquiridos durante el semestre, reforzamos en la práctica los temas vistos en el curso, desde las expresiones regulares, tokens, reglas definidas hasta los AFD y AFND, y especialmente utilizamos el programa de analizadores léxicos “FLEX” para el desarrollo de los programas utilizados; así mismo, adquiriendo nuevas habilidades, experiencias y conocimientos de nuestros compañeros Colombianos y su registro lingüístico del país.

El proyecto realizado junto a la universidad colombiana permitió conocer su cultura y sus regionalismos al igual que fortalecer conocimientos sobre autómatas y analizadores léxicos, aplicando la teoría a la práctica. Esta colaboración fomentó el aprendizaje conjunto, el intercambio de ideas y la comprensión del papel fundamental que tienen estos conceptos en la construcción de lenguajes y compiladores.

En conclusión, este proyecto nos permitió aplicar los conocimientos teóricos del curso sobre autómatas y analizadores léxicos usando FLEX, llevando la teoría a la práctica. Asimismo, la colaboración con la universidad colombiana enriqueció la experiencia, fomentando un aprendizaje conjunto y permitiéndonos conocer su cultura y regionalismos lingüísticos.

Este proyecto nos permitió aplicar conocimientos tanto teóricos y prácticos sobre los temas vistos en el curso de Automatas y Compiladores, utilizando herramientas como flex para los programas desarrollados. Aparte, de la colaboración con la Universidad Colombiana nos permitió adquirir habilidades y conocimientos acerca de los regionalismo colombianos.

Referencias

[En formato APA 7ª. Ed]

Aho, A. V., Ravi, S. A. V., Ullman, J. D. (1998). *Compiladores: Principios, técnicas y herramientas.* Addison Wesley Longman.

Giró, J., Vázquez, J., Meloni, B., Constable, L. (2015). *Lenguajes formales y teoría de autómatas.* Editorial Alfaomega. Argentina.

Gutú, O. (2013). Primer curso en teoría de autómatas y lenguajes formales. Pearson Educación. México.

Hopcroft, J. E. Motwani, R. & Ullman, J. D. (2007). Introducción a la teoría de autómatas, lenguajes y computación. Pearson Educación. México.

Anexo A. Documento fuente

Cristian Danilo Chamorro Mora

Asesino familiar
cuento

Carlos, dejando caer el chuta en un lodero, pasó corriendo por medio de entre una chagra de choclos, habas y ollocos. Iba chumado, por el letargo de la noche, pero no mucho, no tanto como para hundirse en un aljibe o entundarse en algún callejón, pero si lo suficiente como para dejar atrás el chuta de paño nuevo, que traía.

Su afán de correr posiblemente era uno de los tantos efectos del chapíl, pero eso no era lo único, llevaba consigo una razón, un acontecimiento, que minutos atrás lo hubo de haber marcado para siempre. Paso por chagras, terrenos arados y hasta unas cuantas marraneras y cuyeras de un paisano. Y tanto hacerle, a su paso, llegó a la calle principal y única del caserío. Llegó al frente de una casa con terminados coloniales y muy bellos a diferencia de él que llegó limpio desgualangado.

interrumpió, con la avidez de un adolescente, el silencio que cobijaba las casas. Estaba de frente a la casa de su hermana:

- ¡Eugenia!, ¡Eugenia! ¡Salí, pero ya! -Grito emberracado - Creyó no tener respuesta, por eso golpeo y golpeo la maldita puerta colonial, hasta casi atravesarla.

A los segundos, salió una mujer canosa, con la trompa enfurecida. Era ella, la dueña de la casa, se encontraba incorporándose ya, poniéndose la chalina en el paso de la sequía, empezando a mirar la expresión aterrada y empantanada de Carlos.

- ¿Qué pasó, ve, loco? - preguntó Eugenia - frunciendo el ceño.

- Manuel lo mato, mato a su hermano- respondió Carlos -

Eugenia se desplomó enseguida por la noticia, uno de sus hijos había muerto.

Ya de noche, fuera de los hogares del caserío, solo permanecían barones deseosos por tener un encuentro sucio con el chapil, charuco o chancuco.

Era una noche de sábado, como cualquier otra. Taitas y guaguas, no tan guaguas, más bien baroncitos ya, corrían ligerito a los juegos de azar, de una pequeña cantina. Donde el zurrón del Bartolo, el cantinero.

Pasado un par de horas y un buen de pomas, se sentaron en la mesa de naipe para jugar caída, cuatro enruanados. Dos de ellos eran Manuel y Olimpo.

Juego tras juego y trago tras trago, la razón se volvía cada vez más opaca. De pronto Manuel soltó un comentario azaroso sobre el mucho dinero que tenía. A lo que Olimpo no dudó ni un momento en demostrar su disgusto. Con prepotencia se agarraron a insultos. La discusión se iba avivando y con ello el daño físico era inminente. Olimpo al ver que la pelea era obvia, bramo: “sí vamos a pelear, hagámoslo a manga, como los barones, bambarito”

A lo que Manuel respondió, con un silencio pesado y una mirada de enajenación. Para después desenfundar su revólver, apuntar fijamente en el tuste a Olimpo y darle fin a su hermano.

Eugenia recobrando el sentido, que poco de este tenía lo que había pasado con sus hijos, se dirigió a el lugar más horrible que pudo alguna vez imaginar. Ahí estaban los dos, el uno estupefacto, petrificado, y el otro en una escena perturbadora de asesinato, asesinado.

No fue capaz de permanecer viendo esa animalidad. Quería pensar, solo pensar, no quería sentir, se derrumbaría en el momento que lo hiciera.

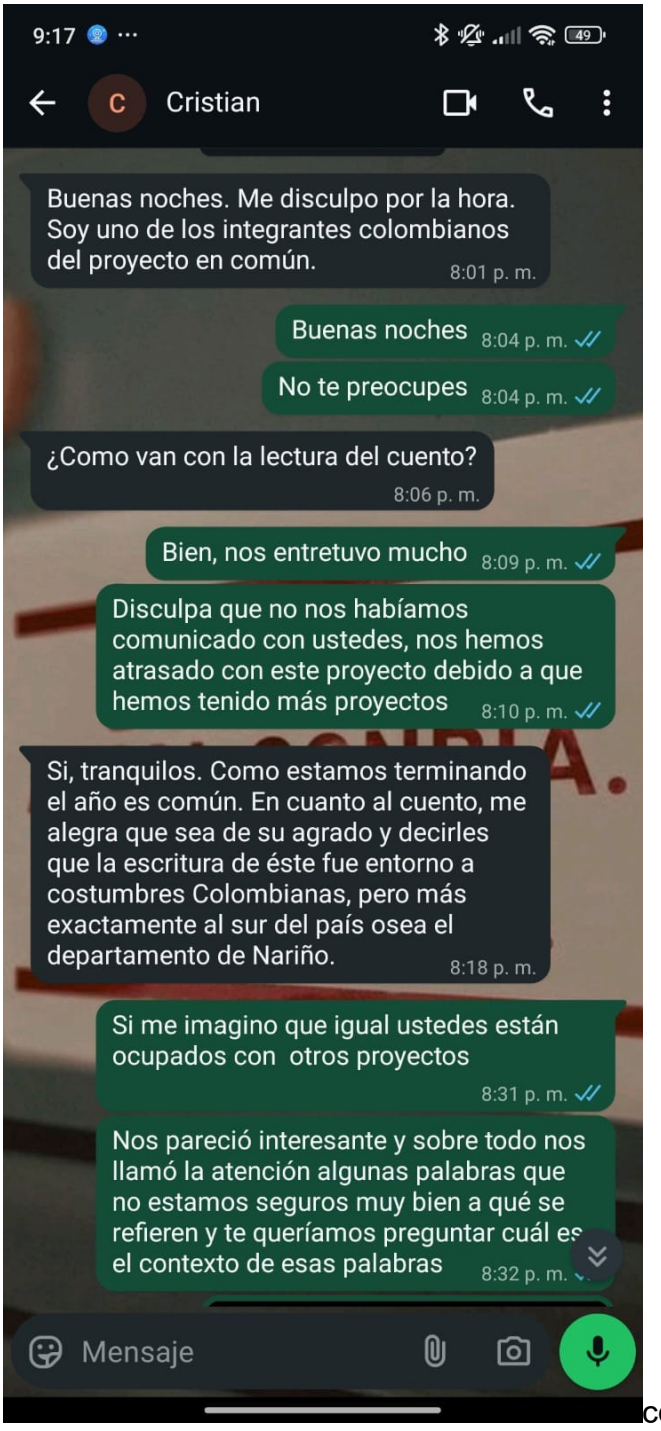
Después del sepelio Eugenia se encontró en una vuelta: Bien entregaba a Manuel a las autoridades, condenándose a la ausencia de sus dos hijos o bien no lo hacía y permitía en su casa de terminaciones coloniales, un asesino familiar.

Anexo B. Traducción de Palabras

chuta	zapato
lodero	charco
chagra	milpa
choclos	maiz
ollocos	papa
chumado	borracho
letargo	cansancio
aljibe	pozo
entundarse	perderse
chapil	mezcal
marraneras	porquerizas (establo o granja porcina)
cuyeras	conejeras
desgualangado	desaliñado
emberracado	enojado
chalina	rebozo
empantanada	llena de lodo
barones	hombres
chapil/charuco/ chancuco	mezcal/aguardiente
taitas	señores
guaguas	niños
baroncitos	hombrecitos /jovencitos

zurrón	gruñón
pomas	tragos
enruanados	hombres con sarape
azaroso	imprudente
bramo	grito
manga	mano limpia
bambarito	valiente
enajenación	enloquecido/fuera de sí
tuste	frente

Anexo C. Evidencia de contacto con Colombiano



cqs

LUIS ALONSO LOZADA CASTELAN <lo398914@uaeh.edu.mx> mié, 12 nov, 12:27 (hace 1 día)

para luna.amaya, cristian.chamorro@udea.edu.co

Hola, nos comunicamos con ustedes sobre el proyecto que tenemos en conjunto, les ofrecemos una disculpa ya que hemos estado atareados con proyectos y trabajos, hemos visto su cuento y hemos trabajado en ello, si gustan mandarnos mensaje les dejo nuestros números para mensaje de whatsapp, les agradezco, tengan un bonito dia.

+52 5549211751 - Sol

+52 7751099546 - Luis

+52 7721056437 - Marco