Pontificia Universidad
JAVERIANA
Bogotá

Documento de Diseño Proyecto

Manuel Alejandro Rios Romero

Daniel Alejandro Millán Peña

Andrés José Rodríguez Ortega

Estructuras de Datos

Andrea Rueda

2021-01

Pontificia Universidad Javeriana

Bogotá D.C

Colombia

Contents

TADS

TAD Diccionario:

Conjunto mínimo de datos:

-nombreUlt, cadena de caracteres, representa el nombre del diccionario.

-palabras, lista de Palabra, representa las palabras guardadas en orden.

-palabrasInv, lista de Palabra, representa las palabras guardadas en orden inverso.

-totalPalabras, numero entero, representa cuantas palabras hay en el diccionario.

-arbolPal, árbol general de Palabra, representa las palabras guardadas en un árbol general.

-arbolPalInv, árbol general de Palabra, representa las palabras guardadas en orden inverso en un árbol general.

Operaciones:

Diccionario (), constructor del TAD que empieza con todos los datos vacíos.

Diccionario (nombre, pala, palInv, total), constructor del TAD que usa los parámetros enviados para asignar a los datos.

obtenerNombre(), retorna el nombre.

obtenerPalabras(), retorna lista con Palabras.

obtenerPalabrasInv(), retorna lista con Palabras inversas.

obtenerTotal(), retorna cantidad total de palabras.

obtenerArbolPal(), retorna el árbol general de Palabra.

obtenerArbolInv(), retorna el árbol general de Palabra inverso.

fijarArbolPal(arbol), fija el árbol general usando el parámetro.

fijarArbolInv(árbol), fija el árbol general inverso usando el parámetro.

fijarNombre(nombre), fija el nombre usando el parámetro.

fijarPalabras(pala), fija las Palabras usando el parámetro.

fijarPalabrasInv (palInv), fija las Palabras inversas usando el parámetro.

fijarTotal(total), fija la cantidad de palabras usando el parámetro.

TAD Palabra:

Conjunto mínimo de datos:

-pal, cadena de caracteres, los caracteres que forman la palabra.

Operaciones:

Palabra(), Constructor del TAD que empieza con la cadena de caracteres vacia.

Palabra(pala), Constructor del TAD que usa el parámetro enviado para asignar el dato.

puntaje(), calcula los puntos de la palabra y los imprime en pantalla.

obtenerPalabra(), retorna la palabra.

fijarPalabra (pala), fija la palabra.

TAD ArbolGeneral

Conjunto mínimo de datos:

-raiz, apuntador a nodo de tipo abstracto, indica el nodo que corresponde a la raíz del árbol

Operaciones:

ArbolGeneral(), constructor de TAD que empieza con todos los datos vacíos.

ArbolGeneral(val), constructor del TAD que empieza con raíz de valor del parámetro.

~ArbolGeneral(), destructor del árbol.

esVacio(), booleano, retorna un verdadero si la raíz del árbol es nula.

obtenerRaiz(), retorna el nodo de la raíz del árbol.

fijarRaiz(nraiz), asigna la raíz del árbol a partir del nodo recibido.

insertarNodo(padre, n), booleano que confirma con verdadero si se inserto el nodo “n” como hijo del nodo “padre”.

eliminarNodo(n), boleano que confirma con verdadero si se elimino el nodo “n”.

buscar(n), boleano que confirma con verdadero si se encuentra el nodo “n”.

altura(), retorna un entero equivalente a la altura del árbol.

tamano(), retorna un entero equivalente a la cantidad de nodos en el árbol.

preOrden(), imprime el árbol en pre-orden.

posOrden(), imprime el árbol en pos-orden.

nivelOrden(), imprime el árbol en nivel-orden.

TAD Nodo General

Conjunto mínimo de datos:

-dato, de tipo abstacto, es el contenido del nodo.

-desc, lista de nodos, contiene los apuntadores a los nodos hijos.

Operaciones:

NodoGeneral(), crea un nodo vacio.

~NodoGeneral(), destruye el nodo y sus contenidos.

obtenerDato(), retorna el dato contenido por el nodo.

obtenerDesc(), retorna la lista que contiene los hijos del nodo.

fijarDato(val), ingresa el dato val en el nodo.

limpiarLista(), limpia la lista de hijos del nodo.

adicionarDesc(val), adiciona a la lista de hijos el parámetro val.

eliminarDesc(val), boleano que retorna verdadero si se elimina el nodo con valor igual a val de los descendientes.

esHoja(), boleano que retorna verdadero si un nodo es hoja.

insertarNodo(padre, n), boleano que retorna verdadero si se inserta un nodo “n” a un nodo “padre”.

eliminarNodo(n), boleano que retorna verdadero si se elimina el nodo “n”.

buscar(n), boleano que retorna verdadero si se encuentra el nodo “n”.

altura(), retorna un entero igual a la altura del nodo.

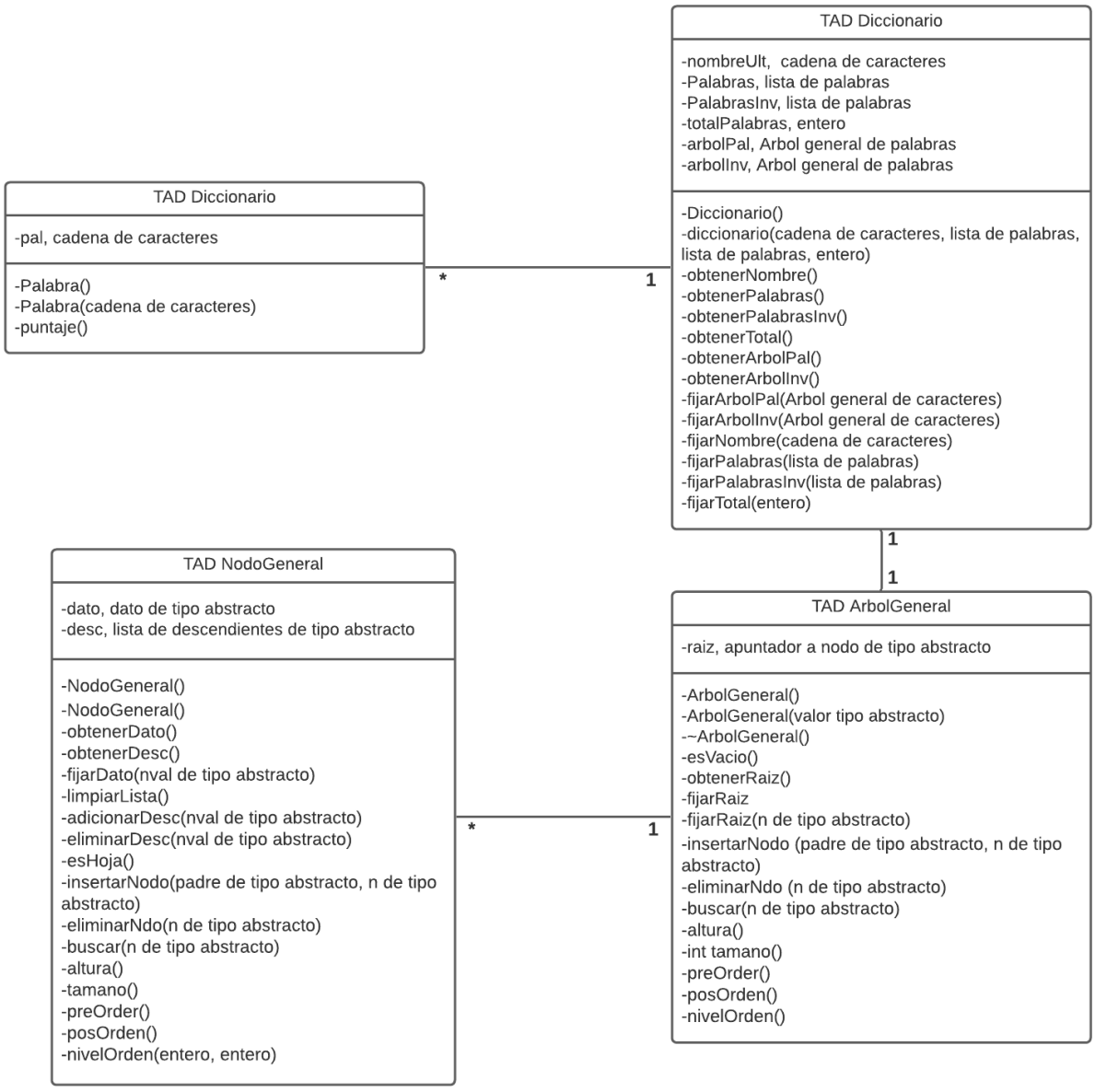
tamano(), retorna un entero igual a la cantidad de nodos dentro del árbol.

preOrder(), imprime el árbol en pre-orden.

posOrden(), imprime el árbol en pos-orden.

nivelOrden(nivel, lvActual), imprime el árbol en nivel-orden, con los parámetro verifica el nivel en el que se encuentra.

Diagrama de relación entre TADs



Funciones Principales:

Main:

Entradas:

- Ninguna.

Salidas Posibles:

- El comando no es válido.
- Se llama la función correspondiente al comando y opera a partir del parámetro si es necesario.

Proceso:

- Se solicita que el usuario ingrese el comando.
- El comando ingresado por el usuario se divide en distintas cadenas de caracteres.
- A partir de la primera cadena de caracteres, se hace llamado a la función correspondiente. Si la función requiere de la segunda cadena de caracteres, esta tambien es utilizada como un parámetro adicional.

Diseño:

Anexo 1

Inicializar Diccionario:

Entradas:

- Cadena de caracteres con el nombre del archivo.

Salidas Posibles:

- Se imprime en pantalla error al cargar archivo.
- Se informa que el diccionario ya está cargado.
- Se carga el diccionario al programa.

Proceso:

- Se crea un stream con el que se intenta abrir el archivo con el nombre recibido. Si esto no es posible, se imprime el error y termina la función.
- Si ya se ha leído el diccionario, se informa que este diccionario ya está cargado y termina la función.
- Al abrir el archivo se revisa linea por linea las palabras, si la palabra es válida esta se guarda en una lista de la variable Palabra.

Diseño:

Anexo 2

Inicializar Diccionario Inverso:

Entradas:

- Cadena de caracteres con el nombre del archivo.

Salidas Posibles:

- Se imprime en pantalla error al cargar archivo.
- Se informa que el diccionario ya está cargado.
- Se carga el diccionario al programa.

Proceso:

- Se crea un stream con el que se intenta abrir el archivo con el nombre recibido. Si esto no es posible, se imprime el error y termina la función.
- Si ya se ha leído el diccionario, se informa que este diccionario ya esta cargado y termina la función.
- Al abrir el archivo se revisa linea por linea las palabras, si la palabra es válida esta se invierte el orden usando una función strrev. Se guarda en una lista de variable Palabra.

Diseño:

Anexo 3

Puntaje Palabra:

Entradas:

- Palabra a revisar puntaje.

Salidas Posibles:

- Si la palabra contiene símbolos inválidos, se informa del error y termina la función.
- Si la palabra no está registrada en el diccionario, se informa del error y termina la función.
- Si la palabra es válida, se imprime en pantalla los puntos que vale la palabra.

Proceso:

- Se verifica que la palabra sea válida.
- Se verifica que la palabra este en el diccionario cargado.
- Si la palabra falla una de las verificaciones, se informa a cerca del error.
- Si la palabra cumple con todas las verificaciones, la palabra sigue a calcular su valor dependiendo de los caracteres que presente.

Diseño:

Anexo 4

Inicializar Arbol:

Entradas:

- Cadena de caracteres con el nombre del archivo.

Salidas Posibles:

- Se imprime en pantalla error al cargar el archivo.
- Se informa que el diccionario elegido ya está cargado.
- Se carga el diccionario al programa y se crea el árbol.

Proceso:

- Se crea un stream con el que se intenta abrir el archivo con el nombre recibido. Si esto no es posible, se imprime el error y termina la función.
- Si ya se ha leído el diccionario, se informa que este diccionario ya está cargado y termina la función.
- Al abrir el archivo se revisa linea por linea las palabras, si la palabra es valida esta se guarda como un camino en el árbol general.

Diseño:

Anexo 5

Inicializar Árbol Inverso:

Entradas:

- Cadena de caracteres con el nombre del archivo.

Salidas Posibles:

- Se imprime en pantalla error al cargar el archivo.
- Se informa que el diccionario elegido ya está cargado.
- Se carga el diccionario al programa y se crea el árbol.

Proceso:

- Se crea un stream con el que se intenta abrir el archivo con el nombre recibido. Si esto no es posible, se imprime el error y termina la función.
- Si ya se ha leído el diccionario, se informa que este diccionario ya está cargado y termina la función.
- Al abrir el archivo se revisa linea por linea las palabras, si la palabra es válida esta se guarda como un camino en el árbol inverso. Las palabras se invierten usando dos cadenas de caracteres y cambiando la posición del primer carácter de la cadena 1 por el ultimo de la cadena dos, aumentando de posición en la cadena 1 y decrementando en la cadena 2 hasta invertir la palabra.

Diseño:

Anexo 6

Palabras por Prefijo:

Entradas:

- Prefijo de las palabras a buscar.

Salidas Posibles:

- Se imprime en pantalla error no se encuentran palabras con ese prefijo.
- Se imprime en pantalla las palabras del prefijo con sus puntajes respectivos y el tamaño de cada una.

Proceso:

- Se recibe el prefijo con el que se buscan las posibles palabras en el árbol.
- Se busca el prefijo en el árbol, si el prefijo no equivale a ninguna palabra en el árbol se imprime el error y se termina la función. Si se encuentra el prefijo se recorre el subárbol del ultimo nodo encontrado.
- Al recorrer el subárbol, se almacenan las palabras en una lista y se retornan.
- Con la lista de palabras se calcula el tamaño de las palabras y su puntaje, finalmente esto se imprime en pantalla.

Diseño:

Anexo 7

Palabras por Sufijo:

Entradas:

- Sufijo de las palabras a buscar.

Salidas Posibles:

- Se imprime en pantalla error no se encuentran palabras con ese sufijo.
- Se imprime en pantalla las palabras del sufijo con sus puntajes respectivos y el tamaño de cada una.

Proceso:

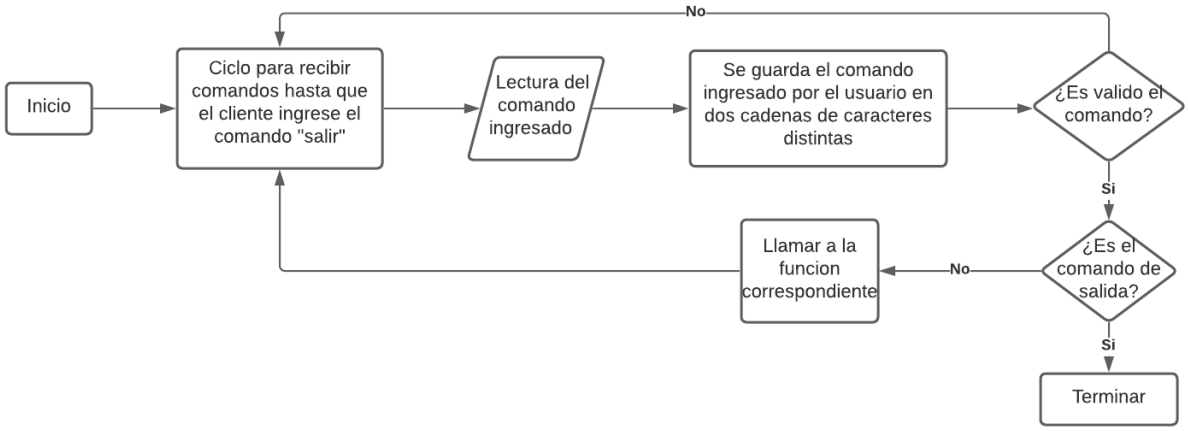
- Se recibe el sufijo con el que se buscan las posibles palabras en el árbol inverso.
- Se busca el sufijo en el árbol inverso, si el sufijo no equivale a ninguna palabra en el árbol inverso se imprime el error y se termina la función. Si se encuentra el sufijo se recorre el subárbol inverso del ultimo nodo encontrado.
- Al recorrer el subárbol inverso, se almacenan las palabras en una lista y se retornan.
- Con la lista de palabras se calcula el tamaño de las palabras y su puntaje, finalmente esto se imprime en pantalla.

Diseño:

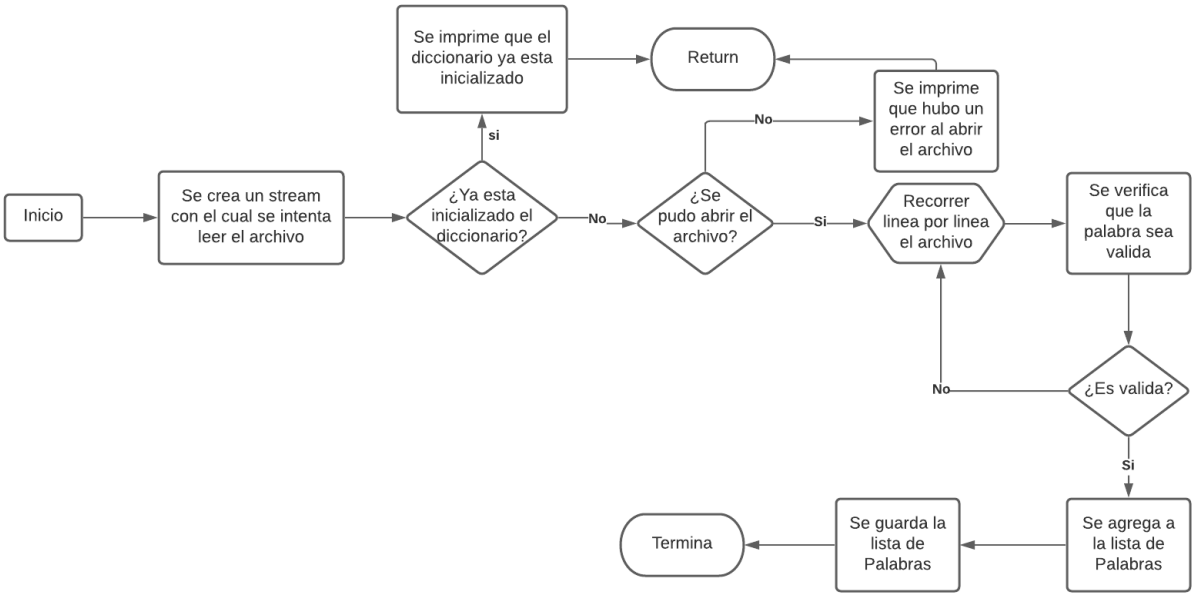
Anexo 8

Anexos:

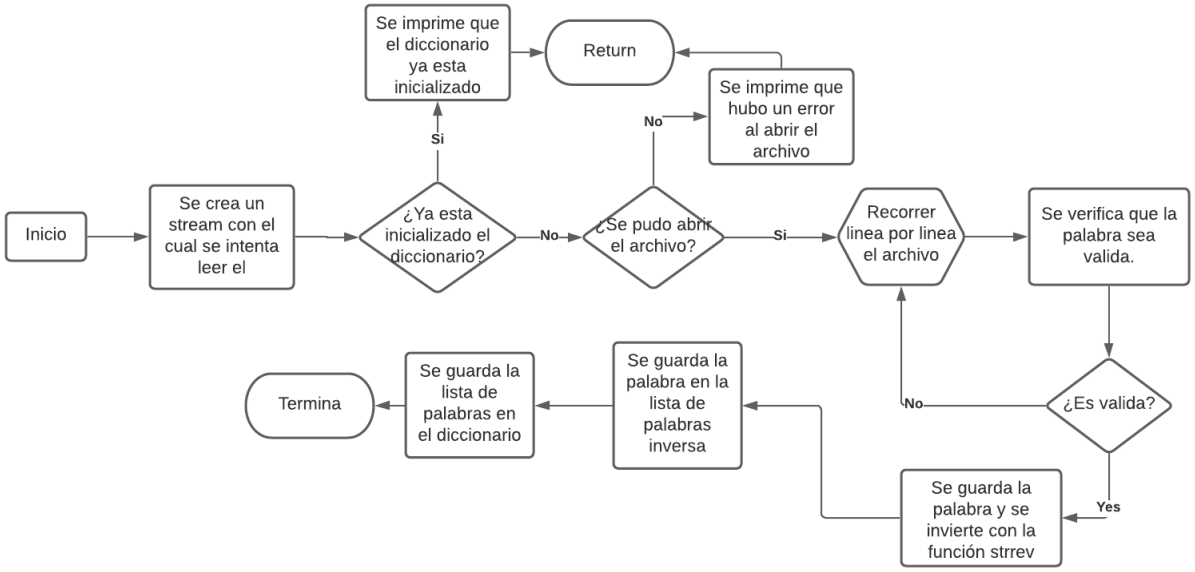
Anexo 1: Main



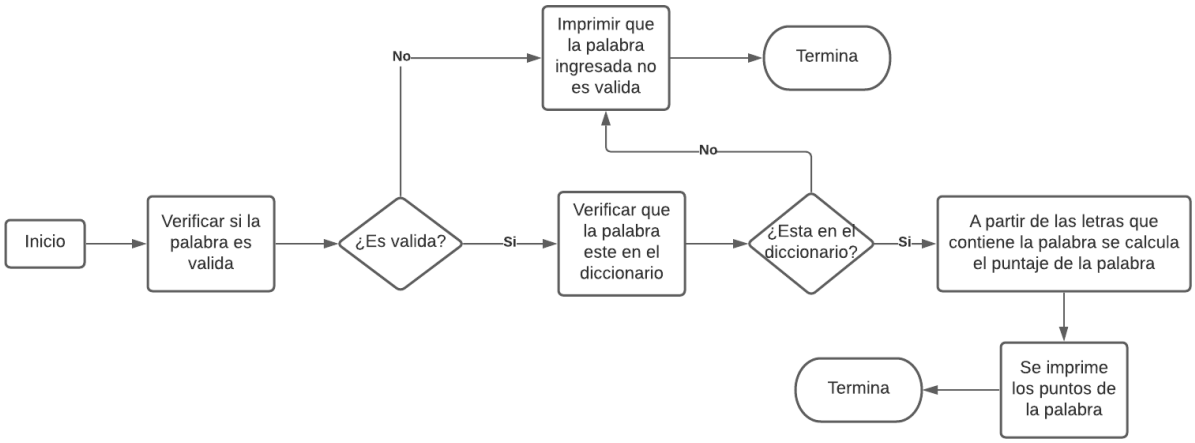
Anexo 2: Inicializar Diccionario



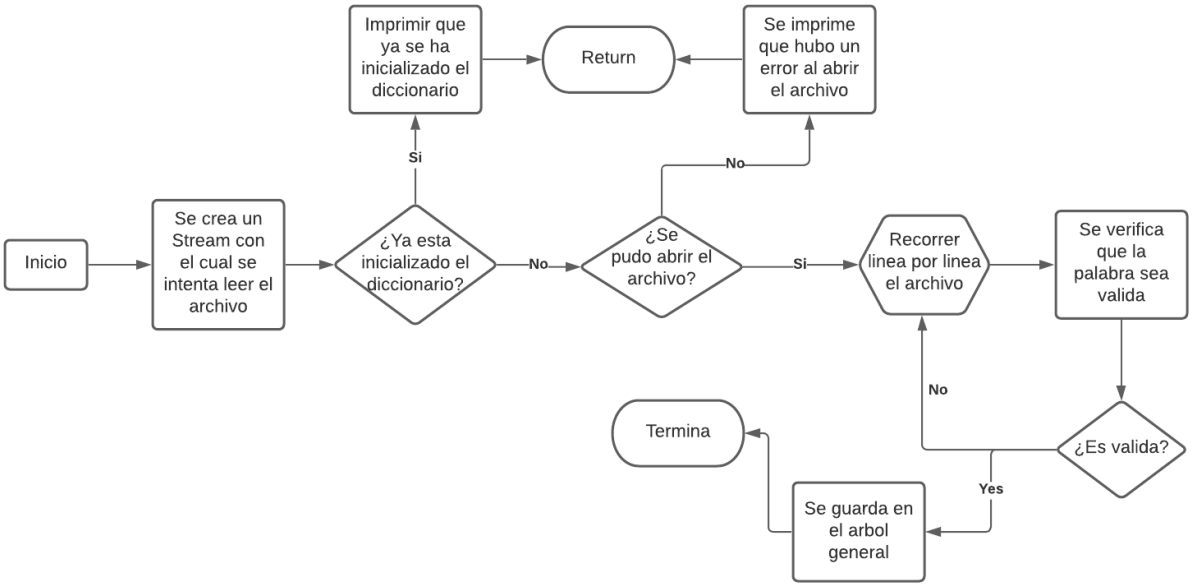
Anexo 3: Inicializar Diccionario Inverso



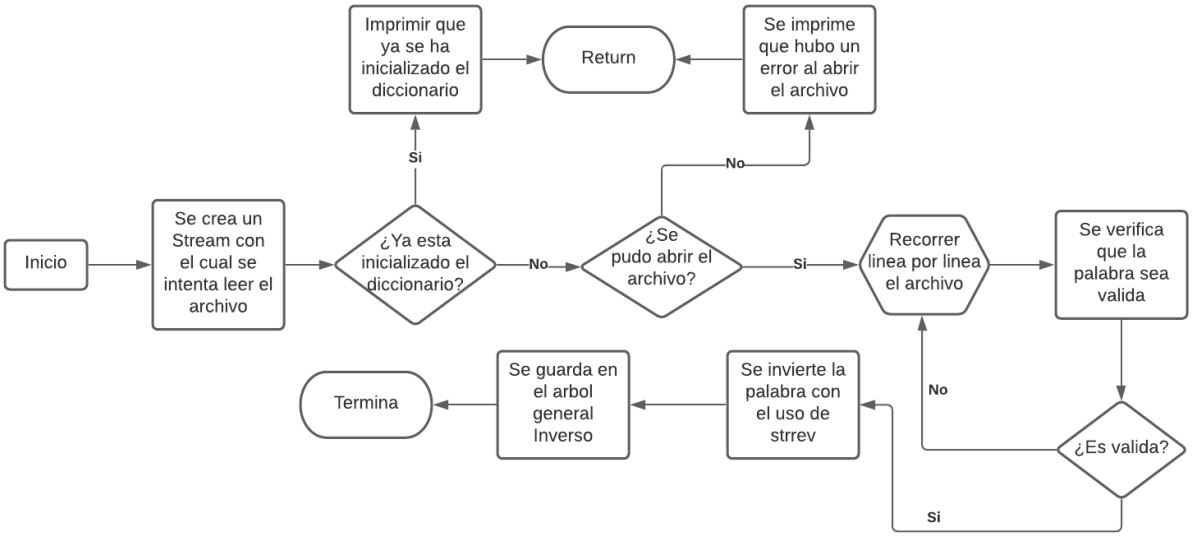
Anexo 4: Puntaje Palabra



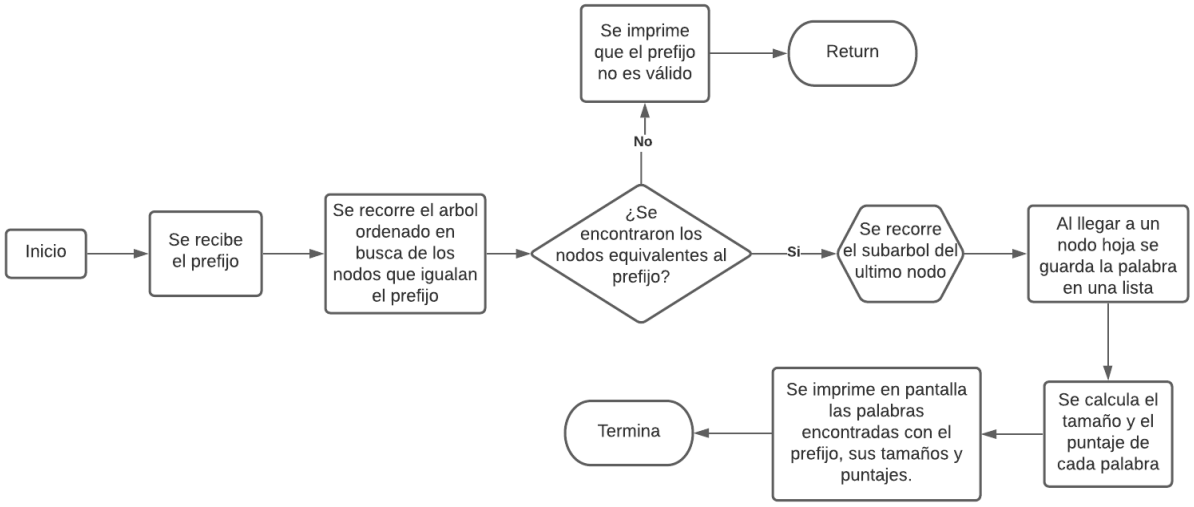
Anexo 5: Iniciar Árbol



Anexo 6: Iniciar Árbol Inverso



Anexo 7: Palabras por Prefijo



Anexo 8: Palabras por Sufijo