Proyecto: Estructuras de Datos (Componente 1)

Grupo #6

Jose David Calderon
Andrés Malagón
Julián Cárdenas

Estructuras de Datos
16 de marzo de 2021

Bogotá
Pontificia Universidad Javeriana

Objetivo: El objetivo del proyecto del semestre es construir un sistema que permita algunas manipulaciones sencillas sobre archivos, con formato FASTA, que contienen códigos genéticos.

Descripción general: La idea principal del programa es la lectura y edición de archivos fasta, estos mantienen una estructura de secuencias organizada que facilita manejo como cadenas de caracteres. Dentro de las principales funcionalidades implementadas se encuentran: Cargar, Conteo, Listar_Secuencias, Histograma, Es_Subsecuencia, Enmascarar, Guardar y Salir, todas estas serán explicadas en detalle más adelante.

Se presentará la documentación del código asignado para la primera entrega del proyecto la cual se encarga del primer componente de funciones las cuales a tintes generales se encargan de todas las acciones posibles con los archivos Fasta, ya que en estos se encuentra la debida secuencia de las diferentes bases nitrogenadas que conforman un genoma, en este caso específico el genoma humano por lo consiguiente se explicara en tres apartados diferentes el funcionamiento del código, las cuales son:

a. Descripción de entradas, salidas y condiciones para la función principal y las operaciones auxiliares
b. los TAD´s utilizados, debe proveerse una plantilla que lo describa brevemente y la lista de sus operaciones
c. Esquemáticos (diagramas, gráficos, dibujos) que ilustren el funcionamiento general de las operaciones principales.

En consecuencia, con estos tres apartadas daremos la mayor claridad al código posible para que el usuario final posea de una debida información y no se pierda en el manejo del software, esto con el fin de establecer la mejor experiencia hombre-máquina.

2. **Entradas**

   Cargar (string archivo): Recibe un archivo de tipo. FA (fasta).

   Conteo: No recibe.

   Enmascarar(vector<string> secuencias, string mascara, int *total): Recibe un string de secuencias de n cantidad de char.

   Es_subsecuencia(vector<string> secuencias, string secuencia, int *numSec): Recibe un string de secuencias de n cantidad de char.

   Listar_secuencia(vector<string> secuencias): No recibe.

   Histograma(vector<string> secuencias, string secuencia): Recibe un string de secuencias de n cantidad de char

   Guardar(vector<string> secuencias, string destino): Recibe un archivo de tipo. FA (fasta).

3. **Salidas:**

   Cargar (string archivo): Si imprime ““nombre archivo” no contiene ninguna secuencia.” en pantalla significa que el archivo no se cargó correctamente. Si imprime “1 secuencia cargada correctamente desde “nombre archivo””, se cargó correctamente.

   Conteo: Retorna la cantidad de secuencias de un archivo.

   Enmascarar(vector<string> secuencias, string mascara, int *total): Retorna la secuencia enmascarada con ‘X’.

   Es_subsecuencia(vector<string> secuencias, string secuencia, int *numSec): Retorna las subsecuencia de la secuencia seleccionada.

   Listar_secuencia(vector<string> secuencias): Si la secuencia contiene un carácter ‘- ‘, retorna la secuencia seleccionada y cada una de las bases de dicha secuencia con un mensaje en pantalla: “contiene al menos _ bases”. Si la secuencia no contiene un carácter ‘- ‘, retorna la secuencia seleccionada y cada una de las bases de dicha secuencia con un mensaje en pantalla: “contiene_ bases”. Si la secuencia no se encontró, retorna un mensaje en pantalla: “No hay secuencias cargadas en memoria”.

   Histograma(vector<string> secuencias, string secuencia): Retorna el conteo de cada código de una secuencia.

Guardar(vector<string> secuencias, string destino): Retorna un texto que define si el archivo fue guardado con éxito o no.

4. TAD´s

Se creó un TAD para el uso del mecanismo inicial del código el cual se desarrolló en el archivo principal, por esto se decidió llamarlo "*TAD main*" ya que este contiene las funciones principales del componente, este se sub divide en dos archivos funcionales:

i. Funciones.h:
   En este apartado se encuentra el enunciado de todas las funciones necesarias para el componente 1, las cuales están nombradas en la gráfica que se mostrara en este mismo apartado.
ii. Funciones.hxx:
   Este archivo contempla todo el código funcional correspondiente para la funcionalidad completa del archivo anterior, cuenta con los enunciados de las funciones y sus respectivos valores de entrada, si los necesita, el código desarrollado y los retornos pertinentes ya sea porque debe retornar una variable o sobrescribirla.

Se decidió la descomposición del código de esta forma para una mayor organización en términos de trabajo y no sobrecargar al usuario con un solo archivo, esto facilita la integración de partes y será una metodología usada en próximas entregas.
Por lo siguiente se adjunta una imagen con la explicación de este:

Ilustración 1 TAD Main

En consecuencia, a lo explicado con anterioridad, se presentan 8 diferentes funciones las cuales son encargadas de la carga del archivo Fasta, su conteo para saber en específico la cantidad de secuencias en él.
Las dos siguientes funciones se encargan encontrar similitudes con las secuencias introducidas de bases nitrogenadas y listar estas mismas para tener un manejo completo del documento.
En secuencia las siguientes se encargan de la modificación y conteo de las bases para por ultimo guardar un nuevo archivo completamente modificado para salir u ingresar otro documento, esto dependerá del usuario.

5. Funcionamiento y esquemas

Función Principal: ***main***

Dentro de la función main se encuentra toda la estructura general de nuestro código, se puede decir que define la estructura visual del programa y está en constante comunicación con el usuario a la espera de que este ingrese algún comando que llame a su funcionalidad definida. Una vez el usuario ingresa correctamente su comando, esta evalúa a través de condicionales que función debe llamar, ejecuta la operación y vuelve a la espera. Todo este proceso iterativo termina al recibir el comando Salir, que cancela cualquier operación y termina el programa inmediatamente.

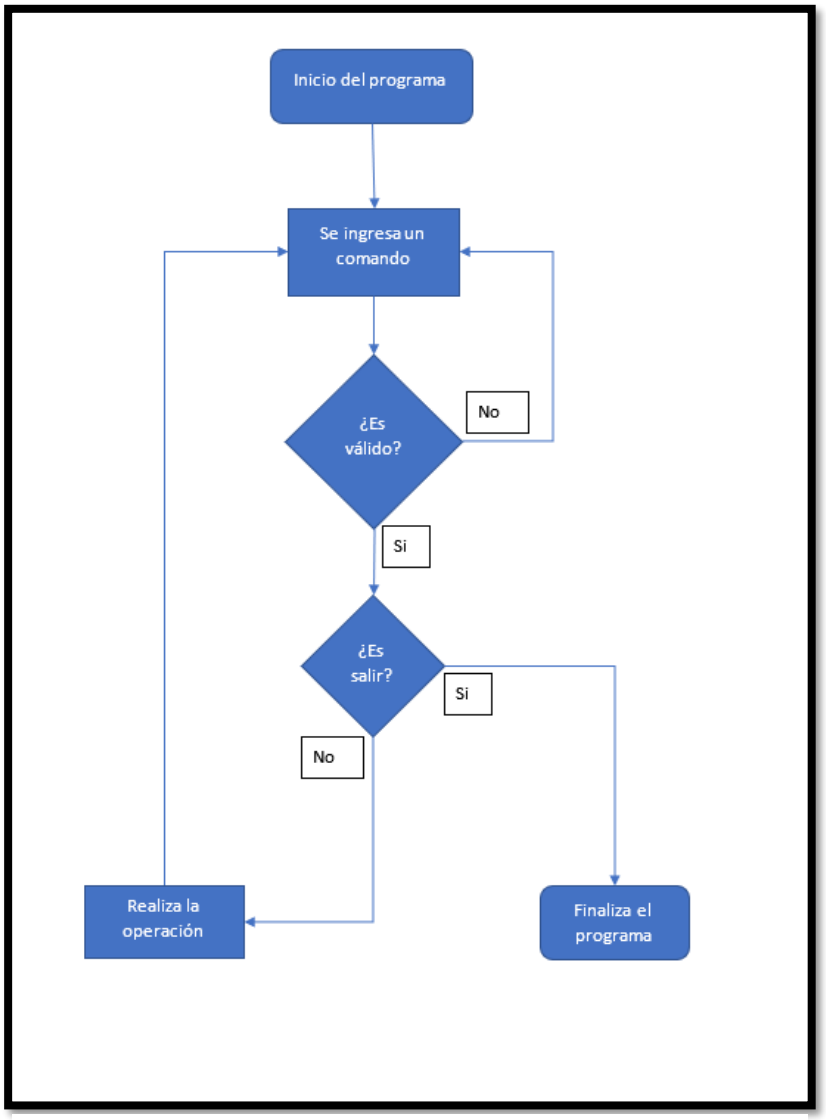

Figure 1 Función main

Función: ***Cargar***

La función cargar se encarga de leer el archivo y sus secuencias, para esto crea un vector de cadenas donde se guardará cada secuencia independientemente. Cabe aclarar que como una secuencia está formada por varias líneas y la lectura del archivo se lee por estas mismas, se decidió hacer una concatenación de líneas hasta que se encuentre el carácter que demarca el inicio de otra secuencia. Para así mantener la estructura general de una secuencia separada por cambios de línea en una sola cadena, El nombre del archivo a abrir llega como parámetro a la función. Y al final retorna un vector de secuencias con las secuencias del archivo.

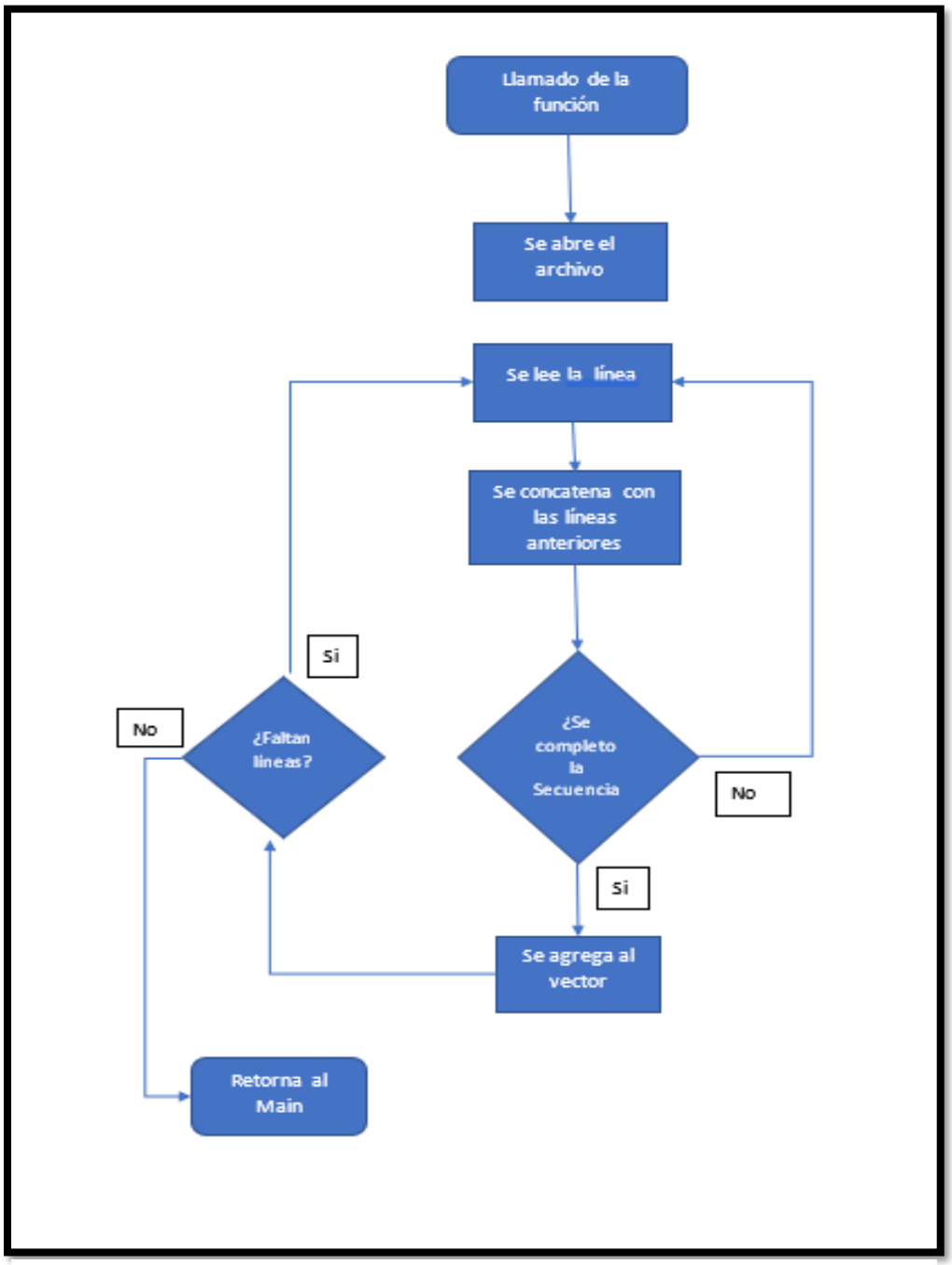

Figure 2Función cargar

Función: ***Conteo***

Esta operación es de las más sencillas debido a la manera en que decidimos guardar las secuencias dentro del programa, Al momento de cargar las secuencias estas quedan guardadas en el vector secuencias del main. por lo cual el conteo seria simplemente contar el tamaño del vector secuencias a través de la función. ***size.***

Función: ***Listar_secuencia***

Esta función lista cada una de las secuencias e indica la cantidad de bases que contiene, además identifica si la secuencia está completa o incompleta dependiendo si contiene el carácter "- "en su composición. Cabe aclarar que solo existen 5 bases por lo que este sería el máximo número en el conteo. Lo primero que se realiza es leer cada una de las posiciones del vector secuencias, las cuales contienen las cadenas, para después utilizar la función ***String::find ()*** que reconoce si esta cadena contiene el carácter o no. Una vez identificados los caracteres (A, C, T, G, U) se le suma al contador y se imprime.

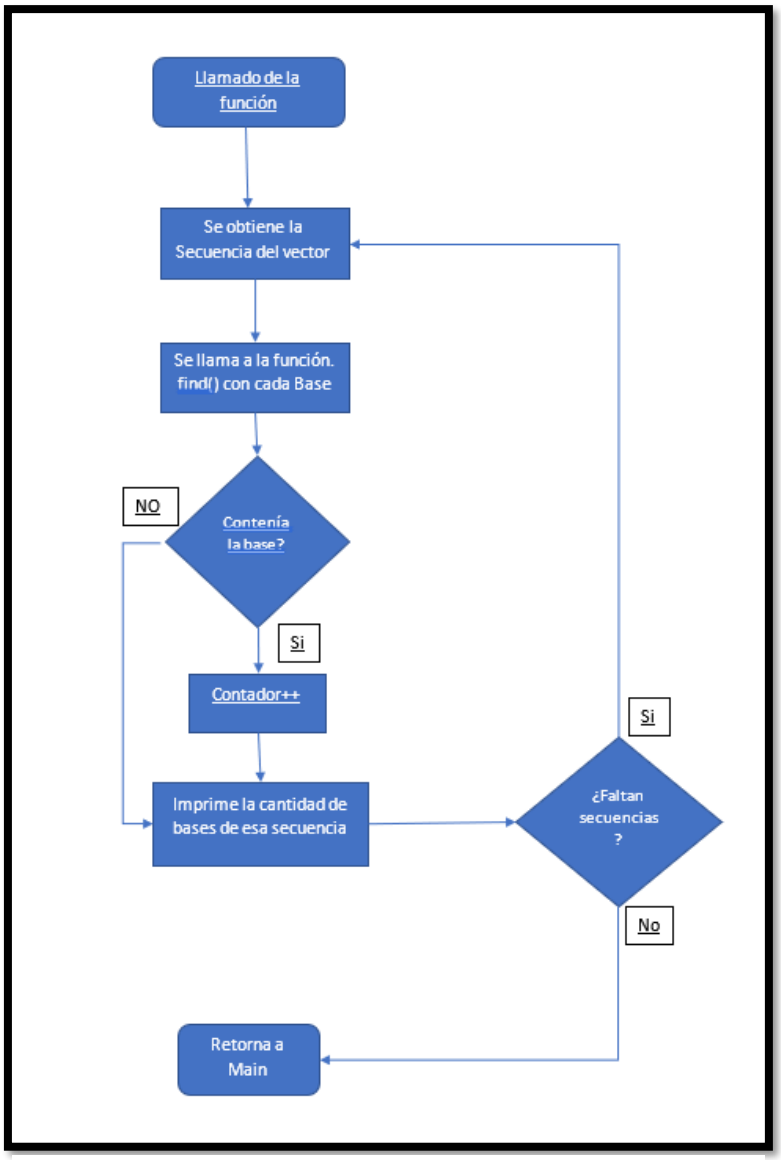

Figure 3Función listar secuencias

Función: ***histograma***

La función histograma realiza un conteo de cada código presente dentro de la secuencia, imprimiendo así una tabla ordenada con cada uno de los códigos y su frecuencia de aparición. Para esto nuevamente utilizamos el vector secuencias donde se guarda cada una de las secuencias cargadas. Posteriormente utilizamos 3 ciclos para recorrer correctamente cada secuencia. El primer ciclo recorre cada una de las secuencias, el segundo y tercer ciclo se encargan de comparar cada uno de los caracteres de la secuencia con cada uno de los caracteres que pueda tener. Una vez contado cada carácter de cada secuencia se imprime dicha tabla y el contador vuelve a 0 para empezar con la siguiente secuencia.

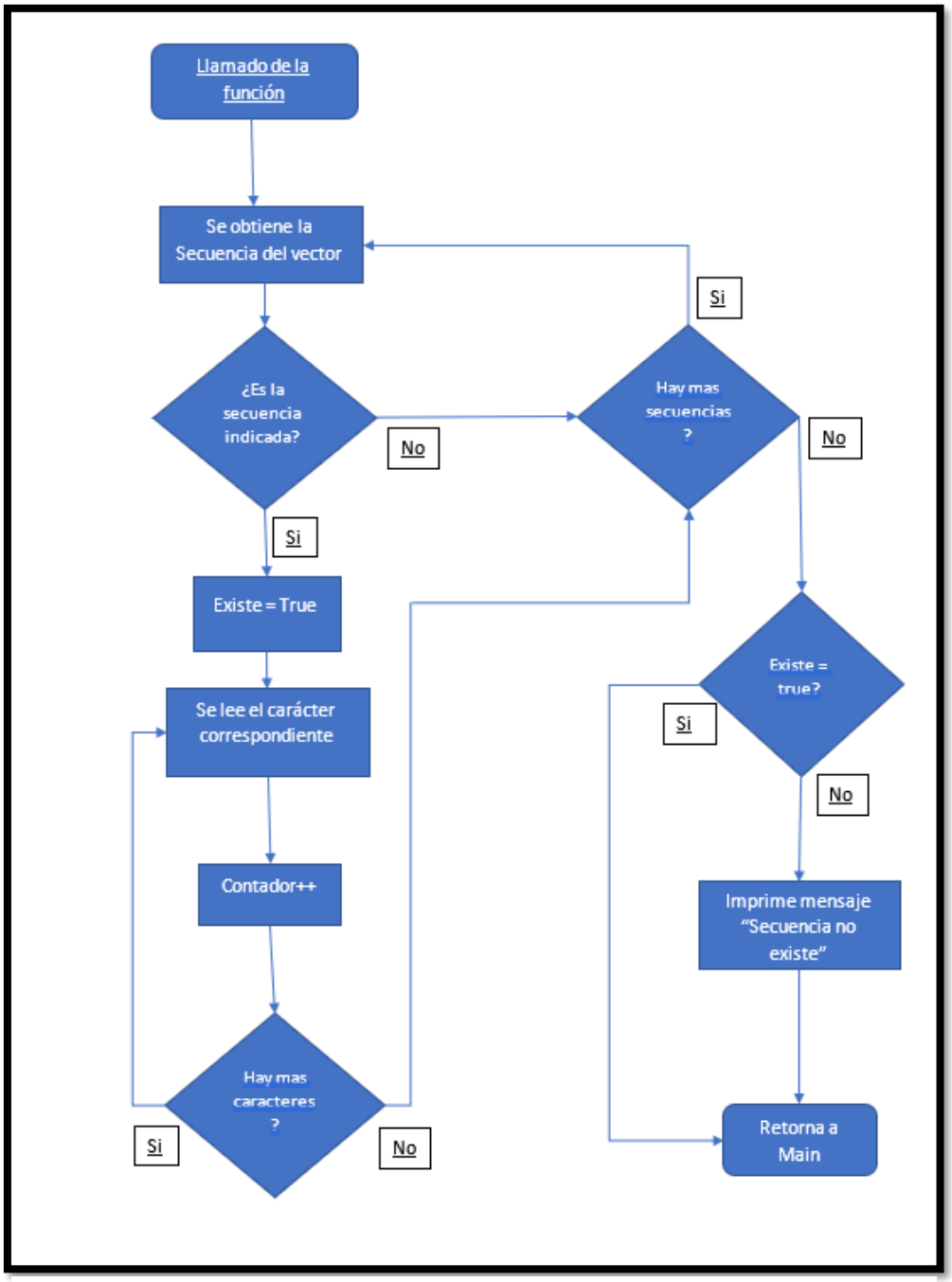

Figure 4 histograma

Función: ***Es_Subsecuencia***

Esta función nos indica si una secuencia dada por el usuario se repite en el archivo fasta y cuantas veces. para su implementación recorrimos con un ciclo cada una de las posiciones del vector secuencias y utilizamos la función. find () para encontrar la

sub secuencia en ella. En caso de encontrarla se le suma 1 al contador y este es retornado por referencia.

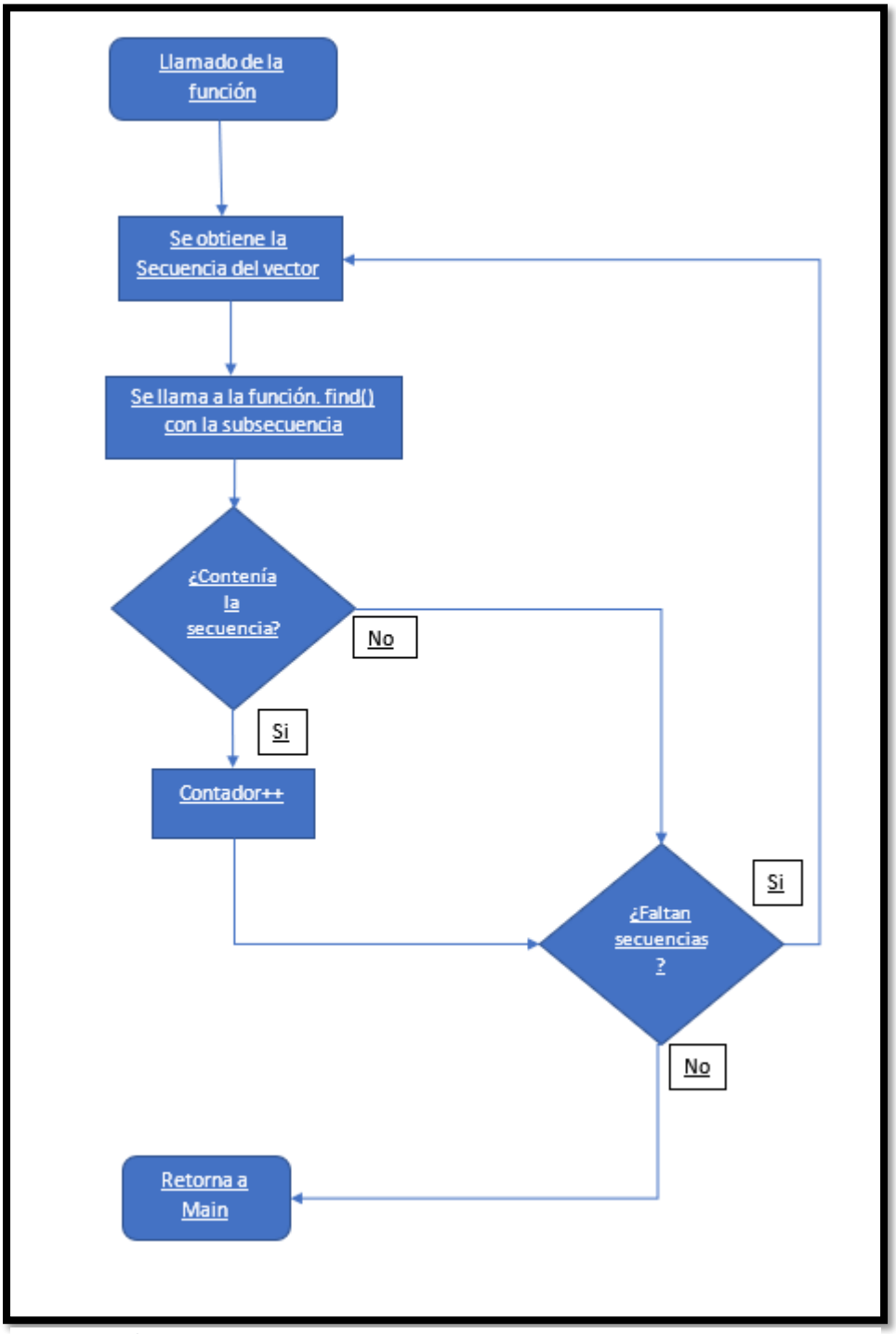

Figure 5 es sub secuencia

Función: ***Enmascarar***

La función enmascarar sigue el concepto base de las demás, recorre el vector de secuencias y retorna así mismo un vector de secuencias, pero con las cadenas enmascaradas. Lo primero que se hace es verificar el tamaño de la sub secuencia que se va a enmascarar para luego tomar tantos caracteres de la secuencia haya y compararlos. Al momento de encentrarla, reemplaza los caracteres con X y sigue así por toda la secuencia hasta terminarla. Repite el mismo proceso con cada secuencia del vector Secuencias.

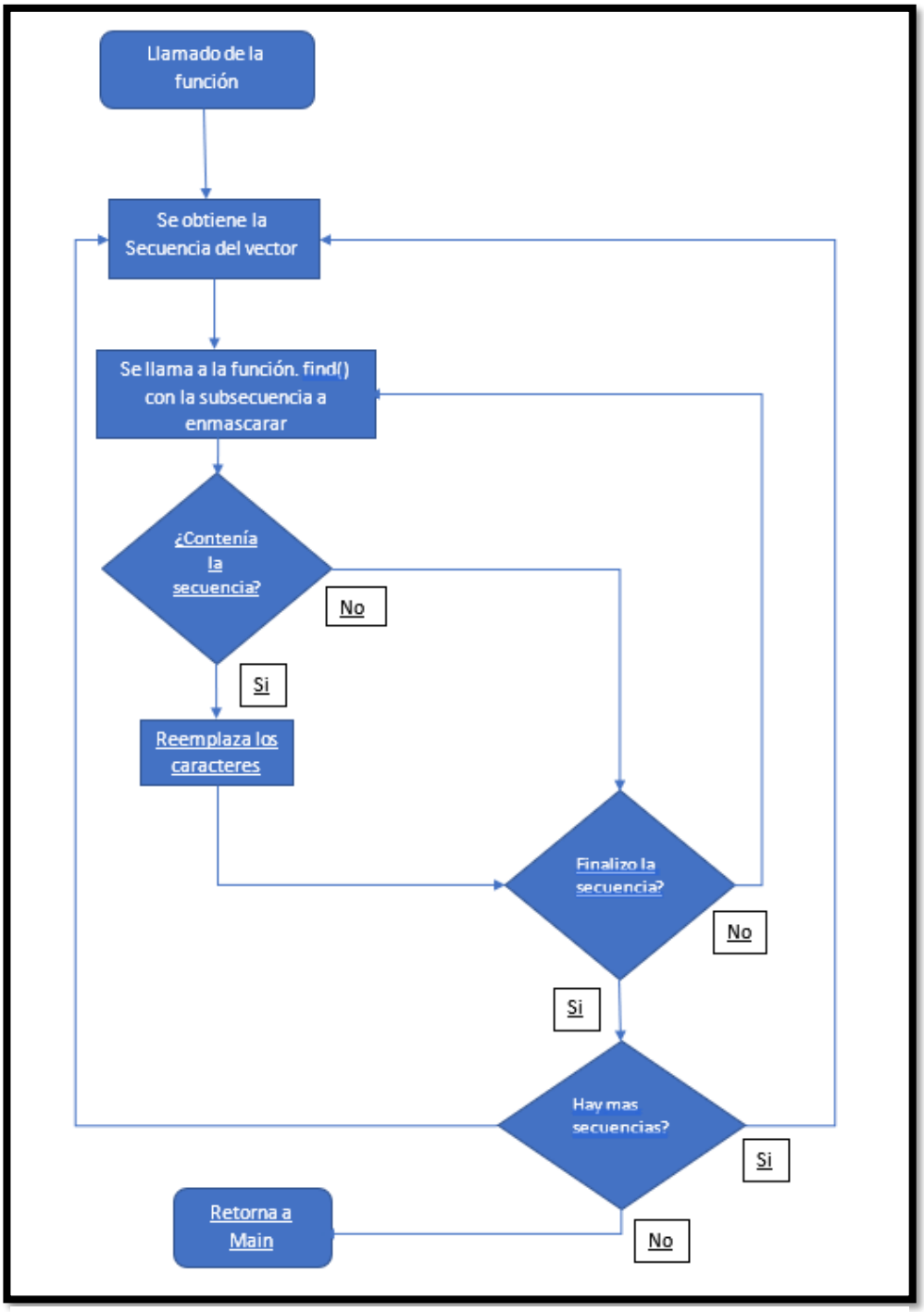

Figure 6 enmascarar

Función: ***Guardar ()***

Esta función es de las más sencillas de implementar ya que está basada netamente en la impresión de caracteres sobre un archivo. Lo primero que se realiza es la apertura de un archivo en modo escritura, luego se recorre cada cadena almacenada en

memoria en el vector secuencias y se imprime en el mismo. Dependiendo si la escritura fue correcta o no se le mostrara un mensaje al usuario en pantalla.

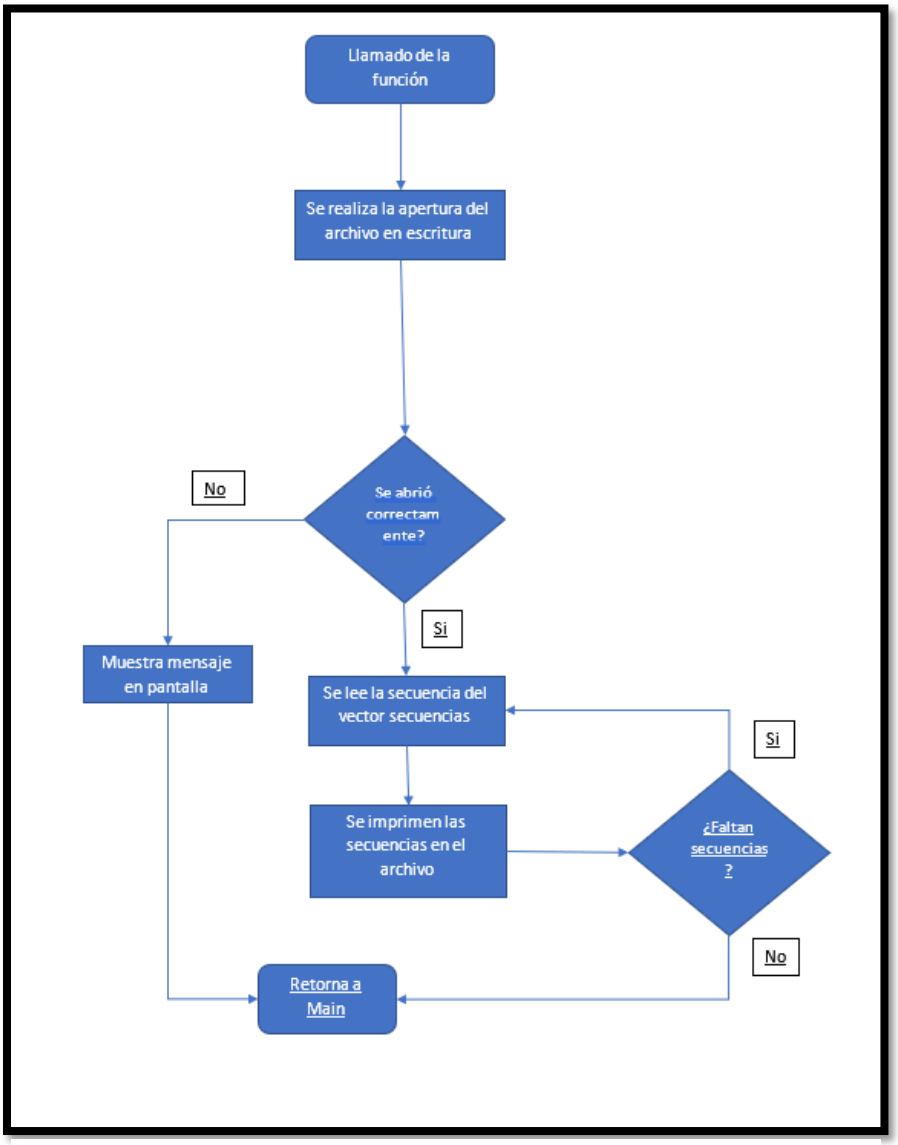

Figure 7 guardar

Función: ***salir***

Esta funcionalidad es la más sencilla de todas, en el main, una vez se detecta que el comando o cadena ingresado por el usuario es "**salir**", esta llama a la función ***exit***, y se cerrara el programa inmediatamente.