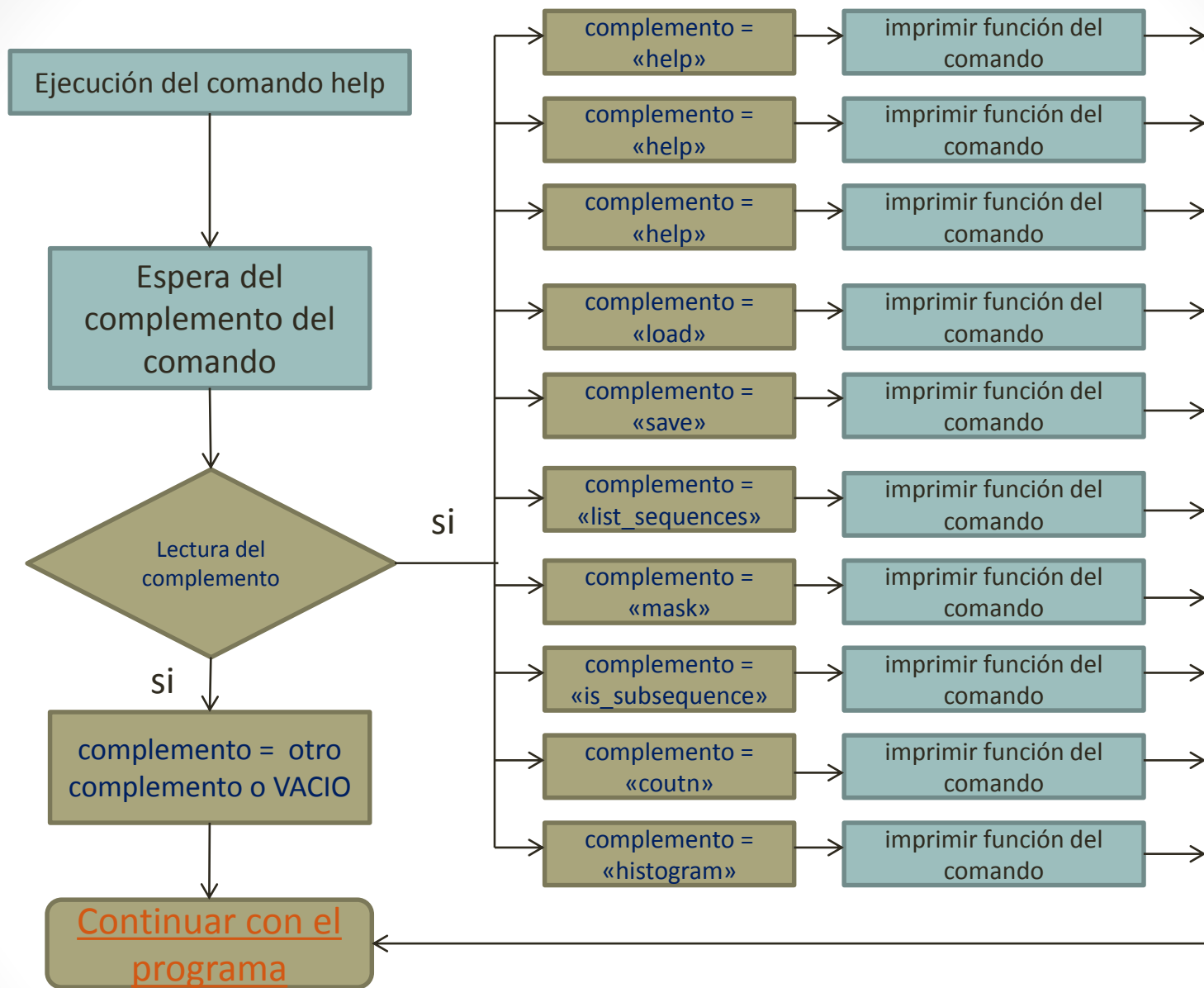


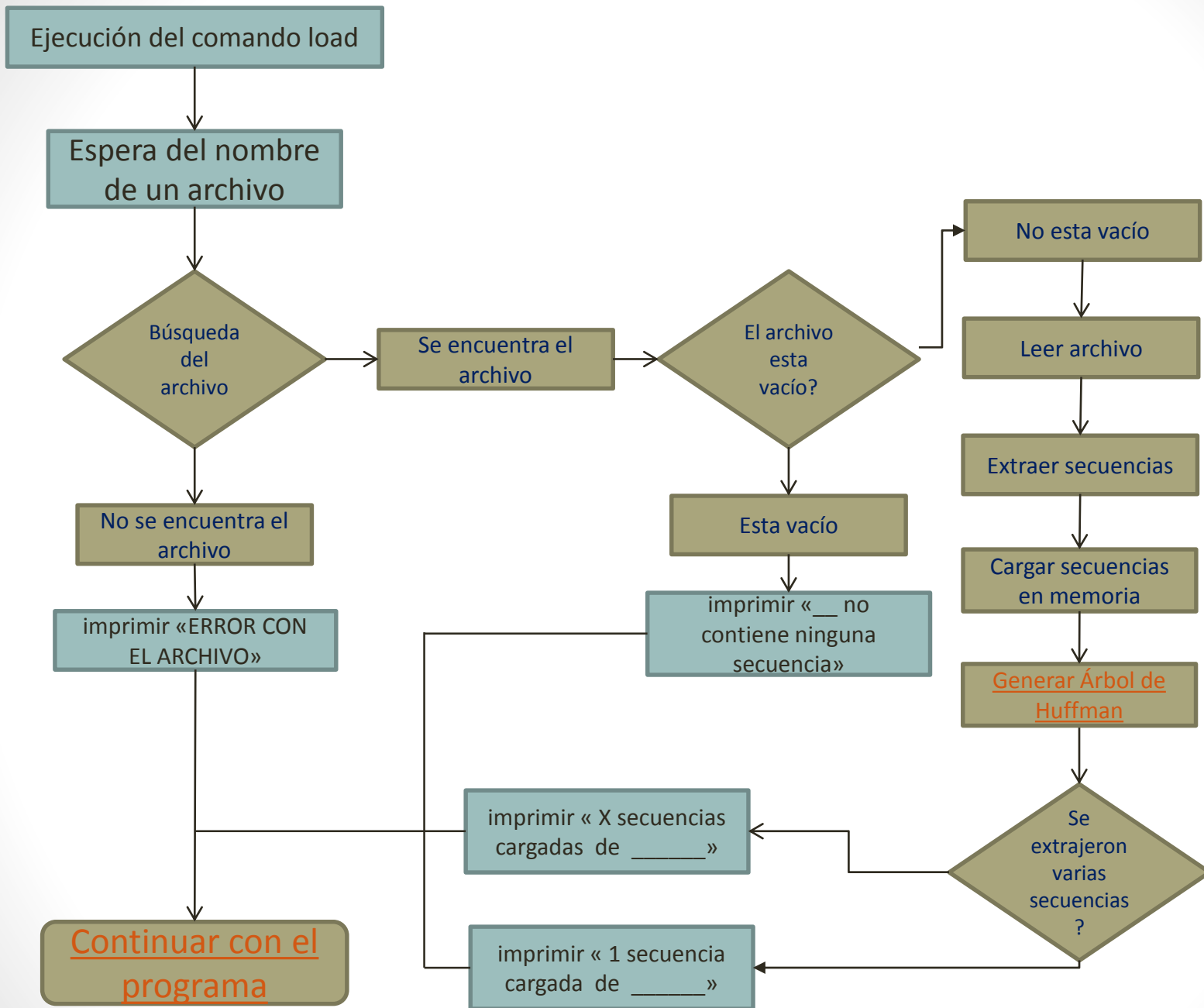
## Esquemático del funcionamiento del proyecto (entrega2)

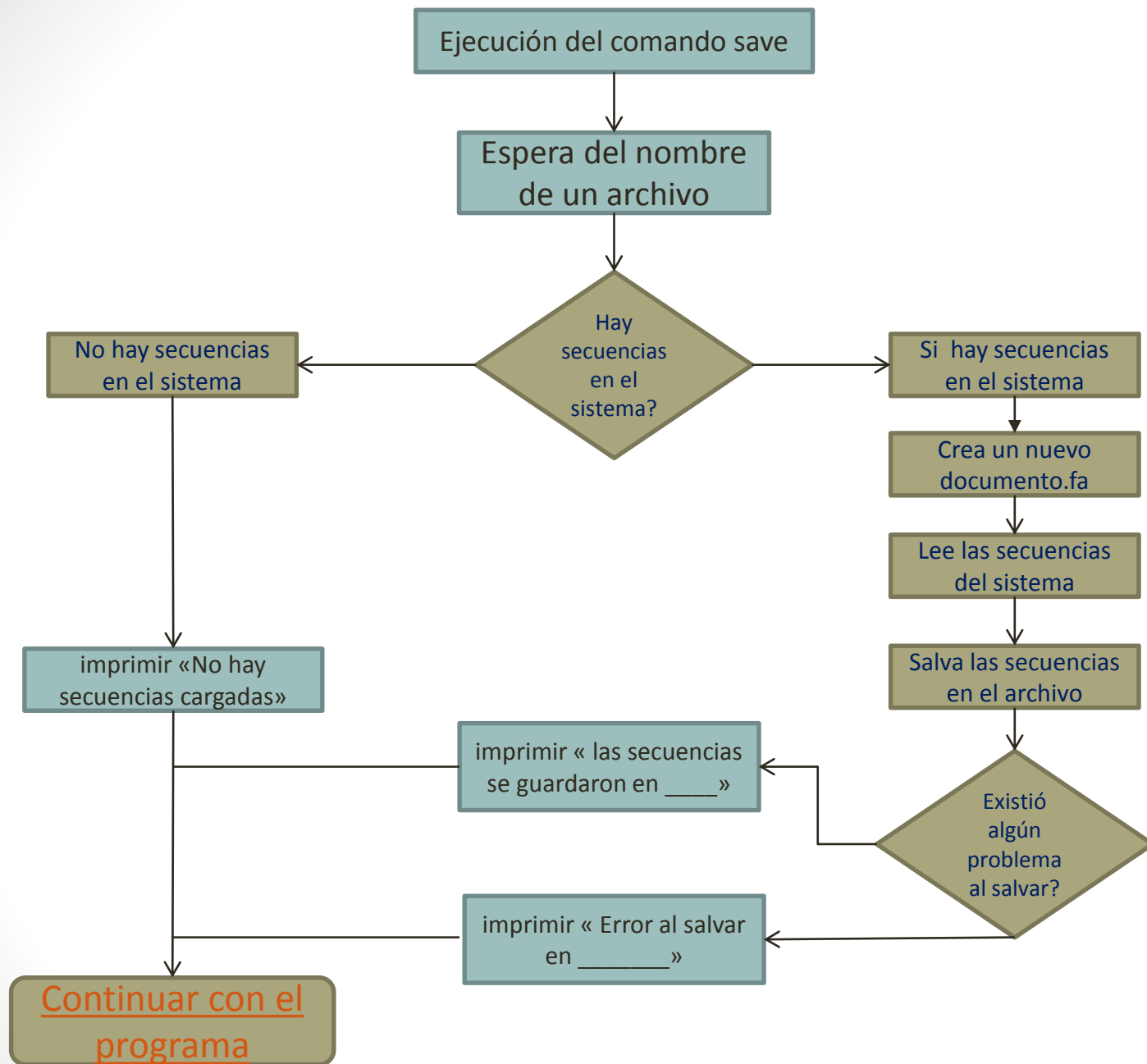
Juan Pablo Rodríguez  
Luis Felipe Urdaneta  
2015

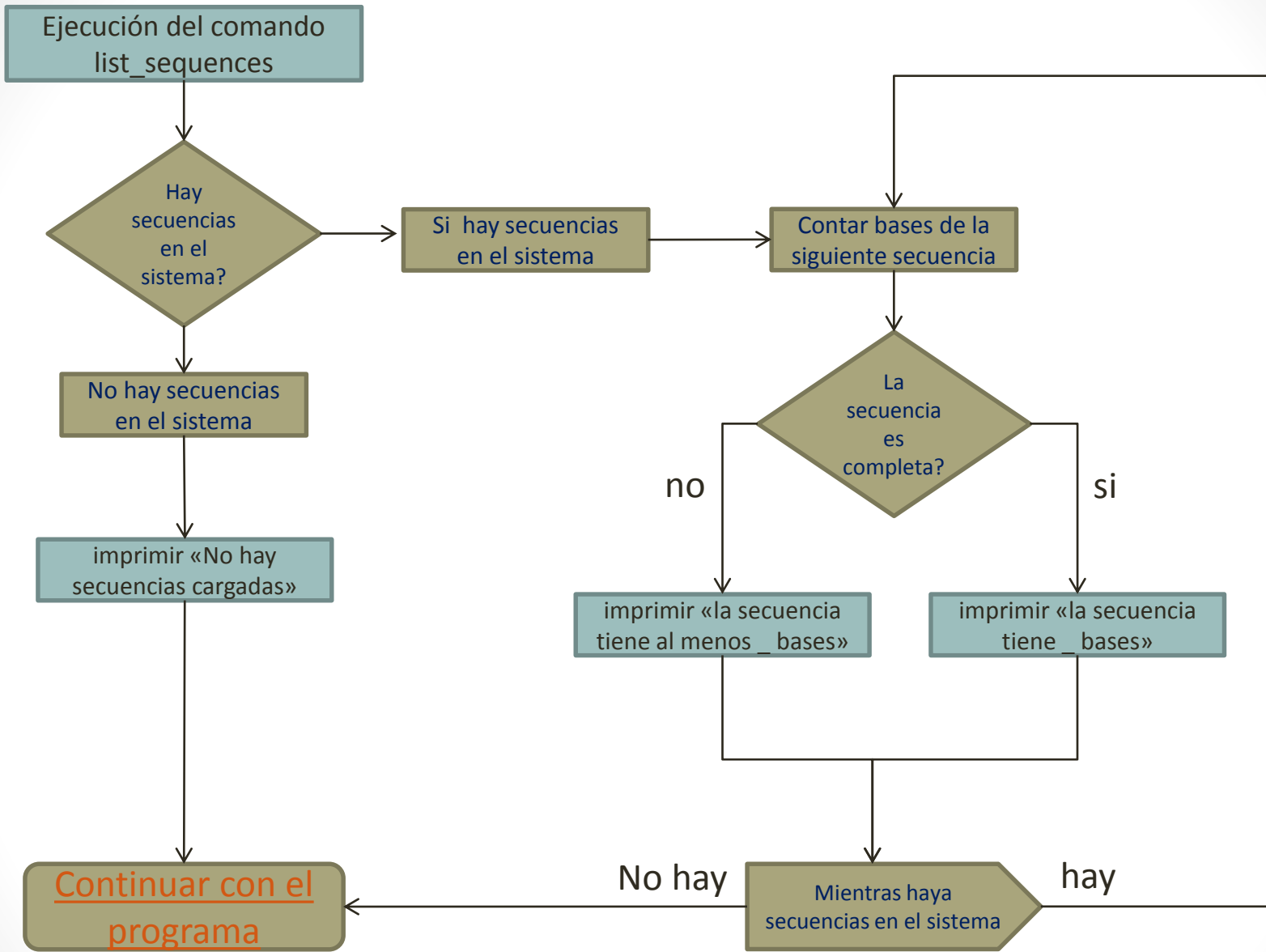
Nota: para navegar utilice los hipervínculos dentro de cada segmento.

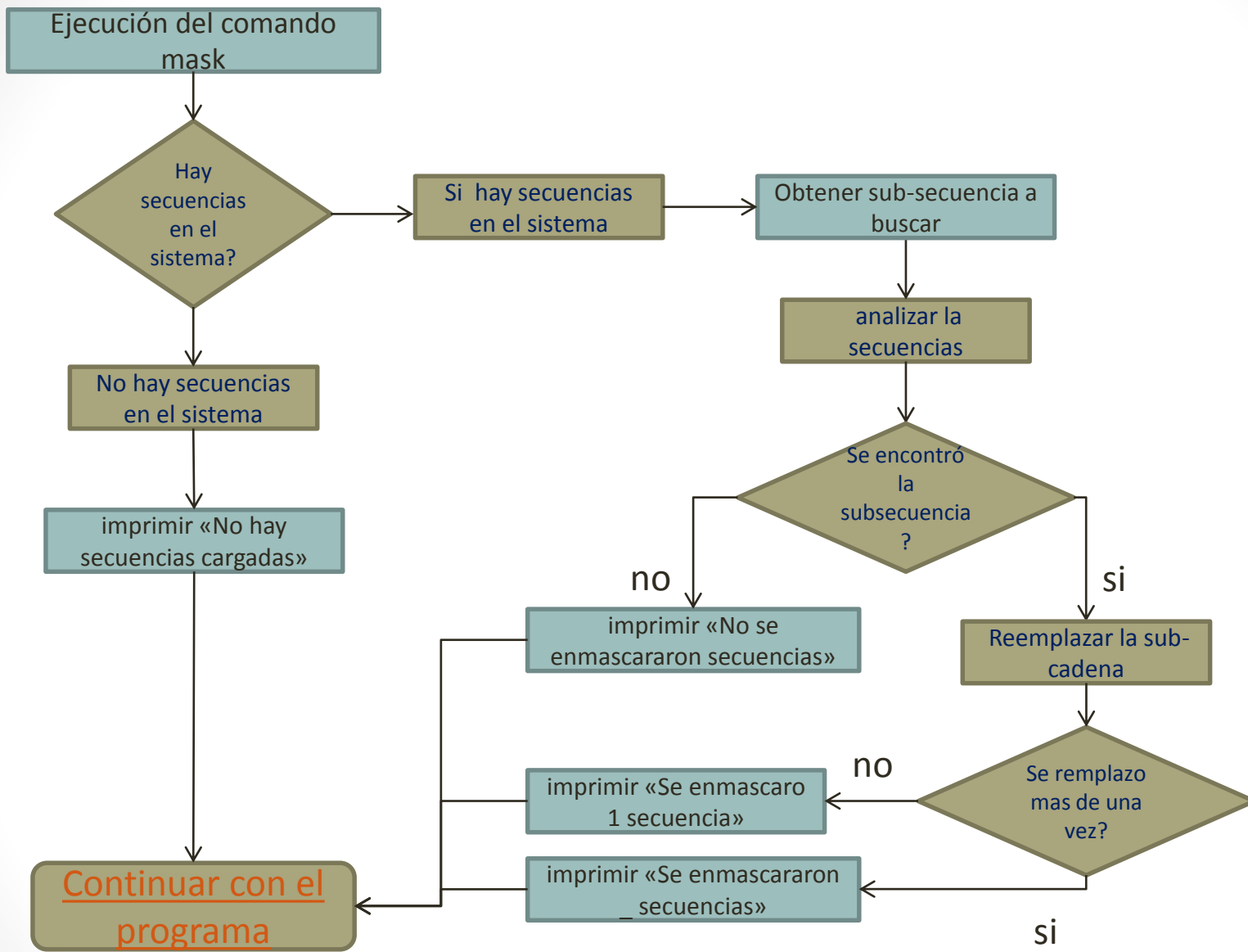
[Como se forma el Árbol de Huffman.](#)  
[Como esta codificado el archivo binario.](#)

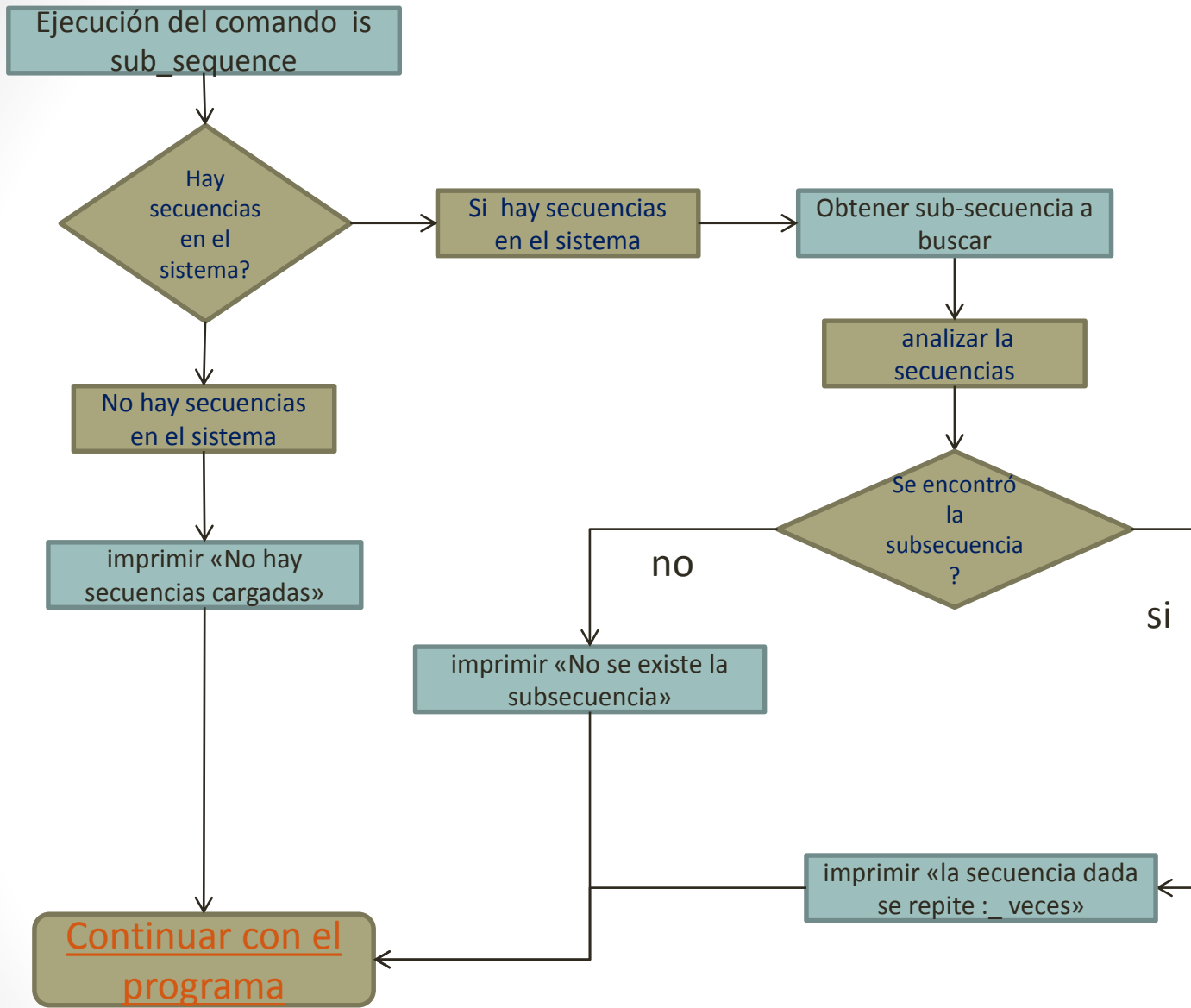


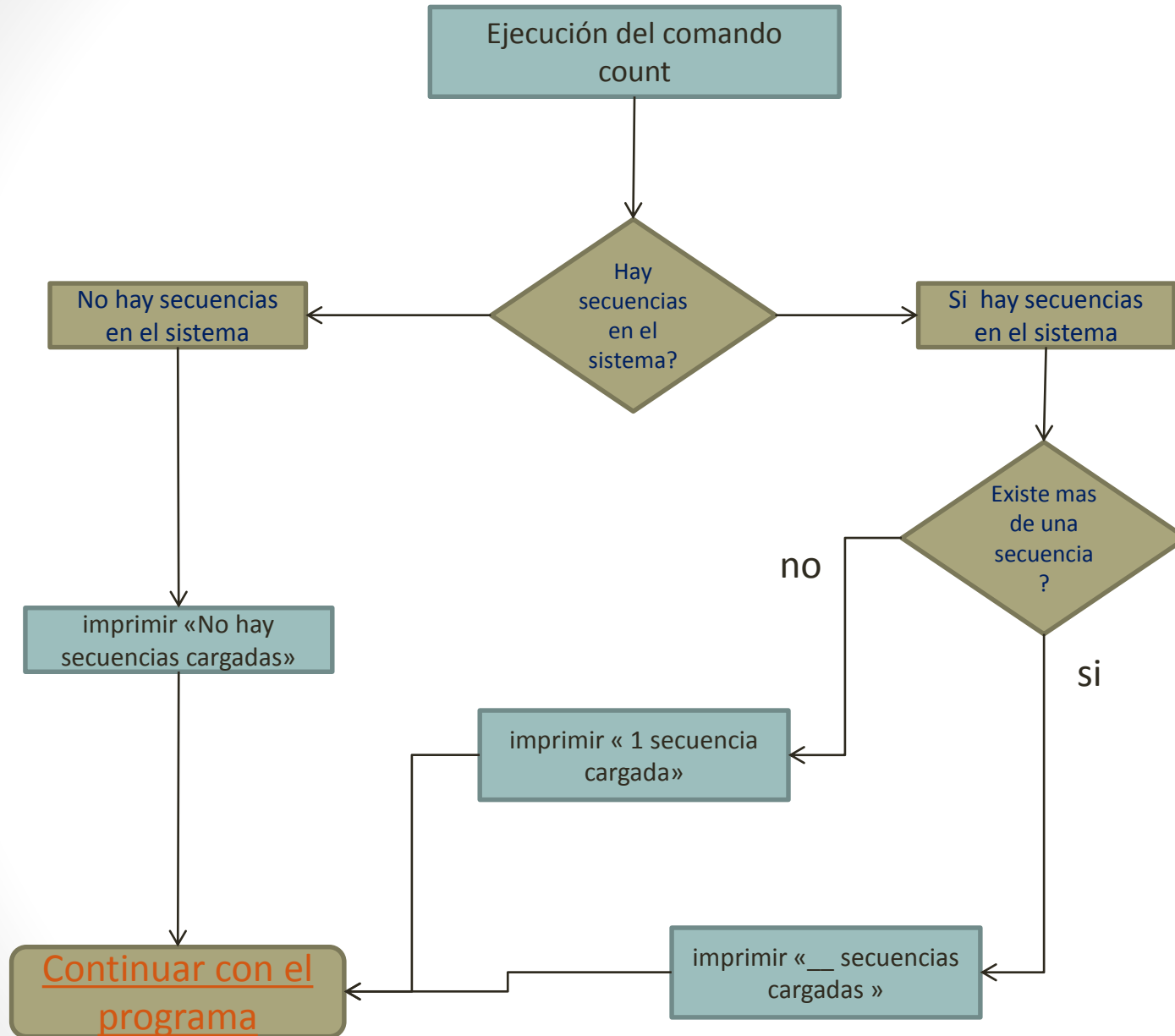




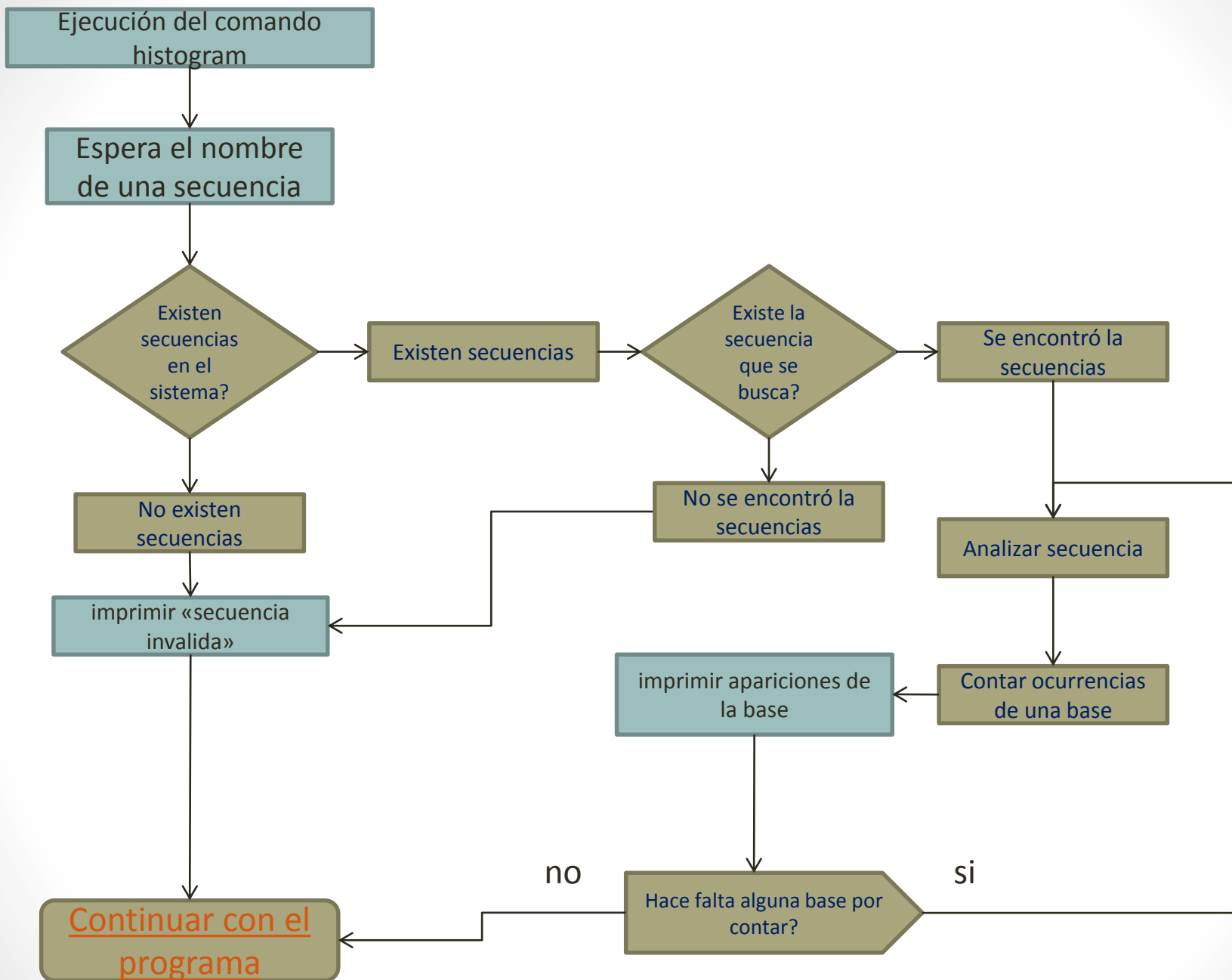


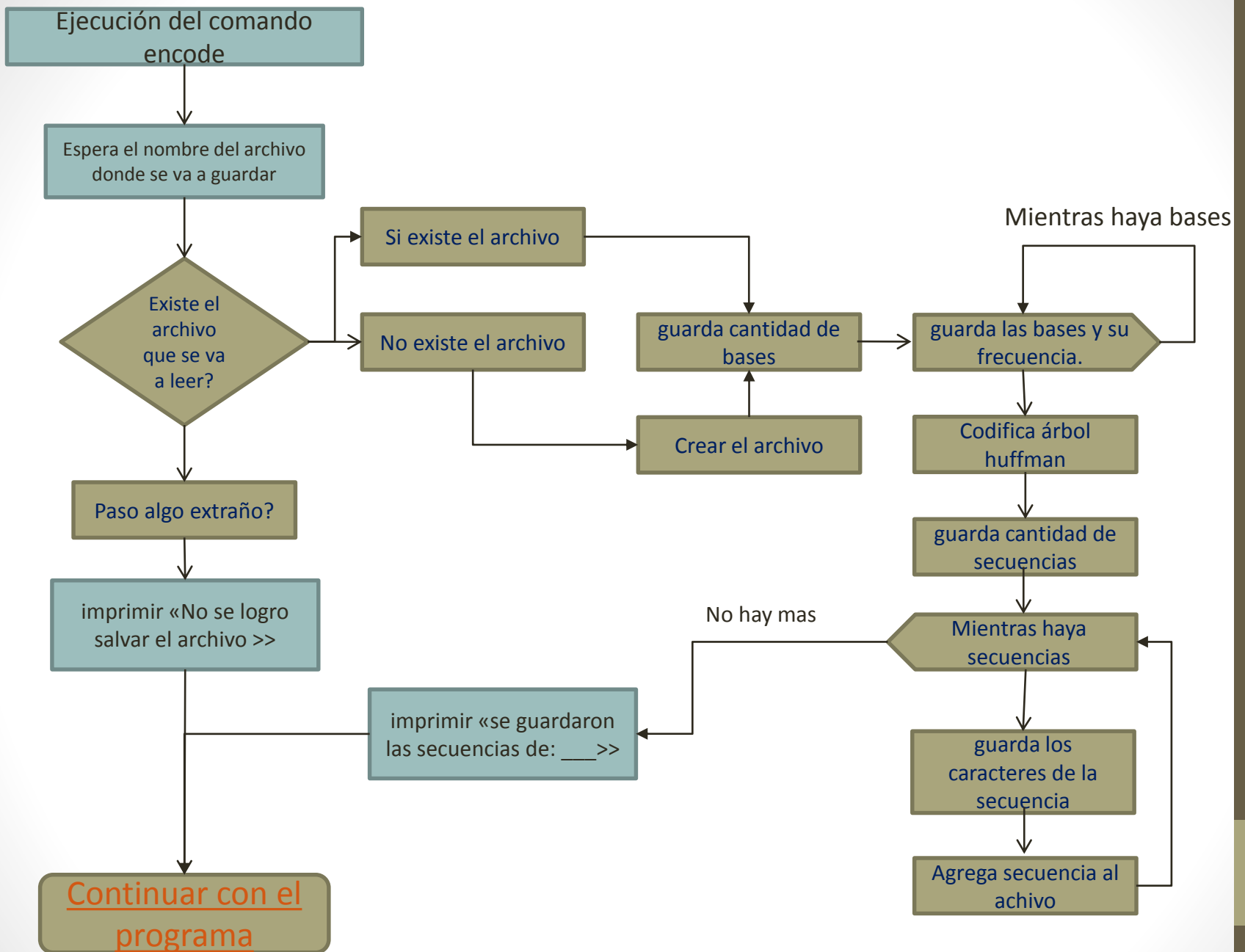


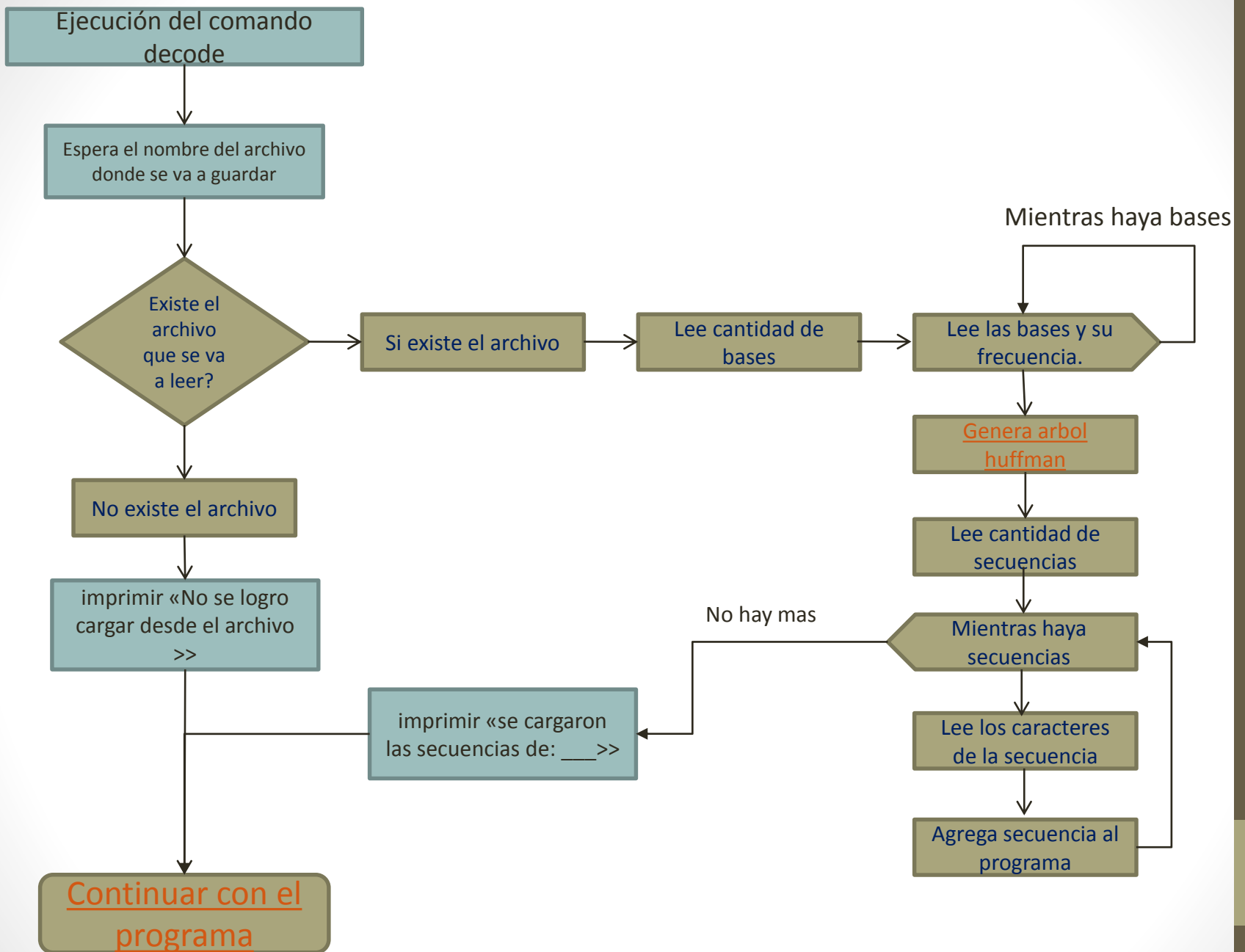




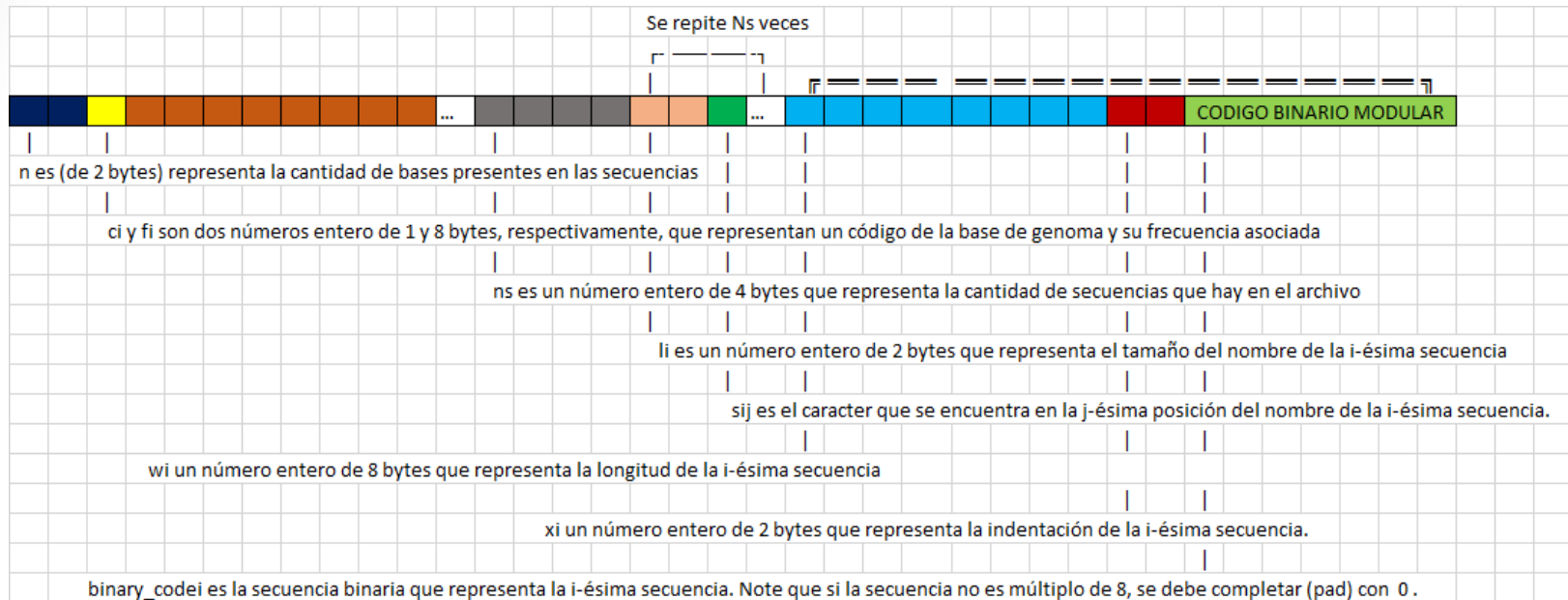








# Como esta codificado el archivo binario:

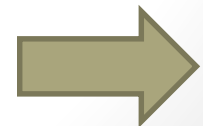


Nota: con código binario modular nos referimos a la secuencia de bits `binary_code`, donde un modulo individual representa cada letra dentro de una secuencia

[Volver](#)

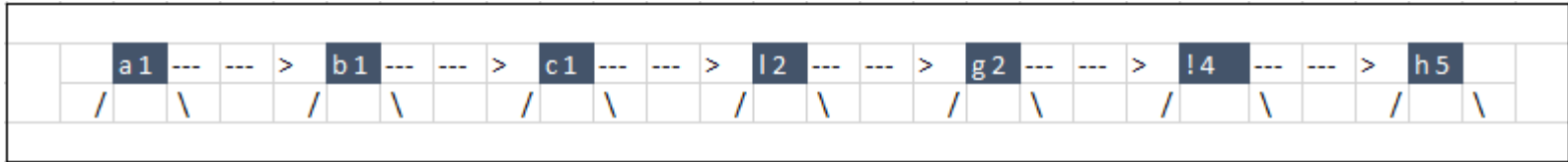
# Como se genera el árbol de Huffman:

- Al momento de leer los datos, estos se agregan a un Map de carácter-frecuencia, donde se cuenta la cantidad total de todas las bases presentes en todas las secuencias del archivo que se lee.
- Por medio de una función, cada dato del Map se convierte en un NodoHuffman (remítase al documento adjunto del diseño del proyecto).
- Estos nodos se insertan en una Priority queue, que por medio de un comparator organiza todos los elementos al momento que se ingresa un nuevo nodo.
- A continuación se muestra como se forma el árbol a partir de esta priority queue, después de que todos los datos han sido insertados.



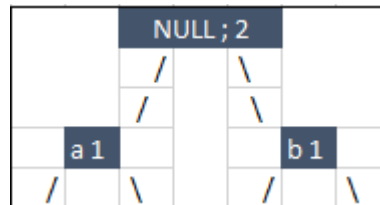
# Generar el árbol a partir de la priority queue

Tenemos nuestra priority queue ya ordenada, el elemento mas a la derecha es El tope de la cola

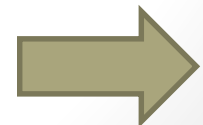
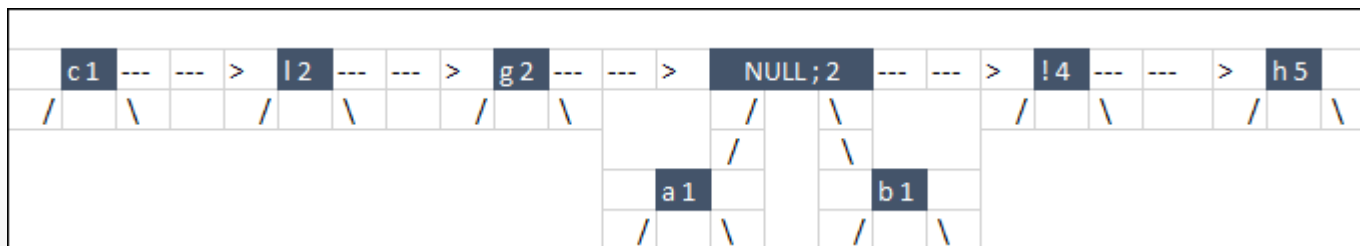


Los 2 primeros elementos de la cola se extraen y se les asigna un padre de carácter que nunca va a ser una base, para la explicación se usara un NULL para mostrar esta idea.

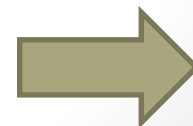
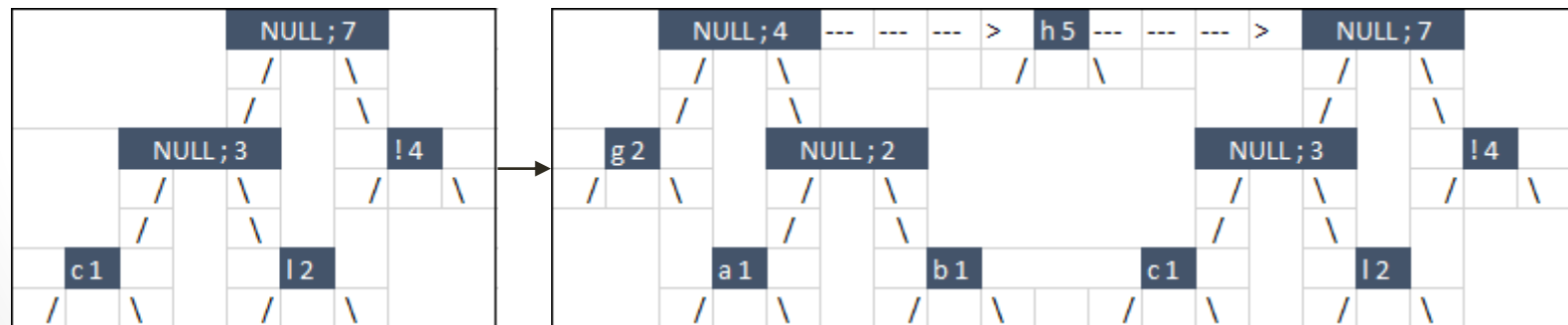
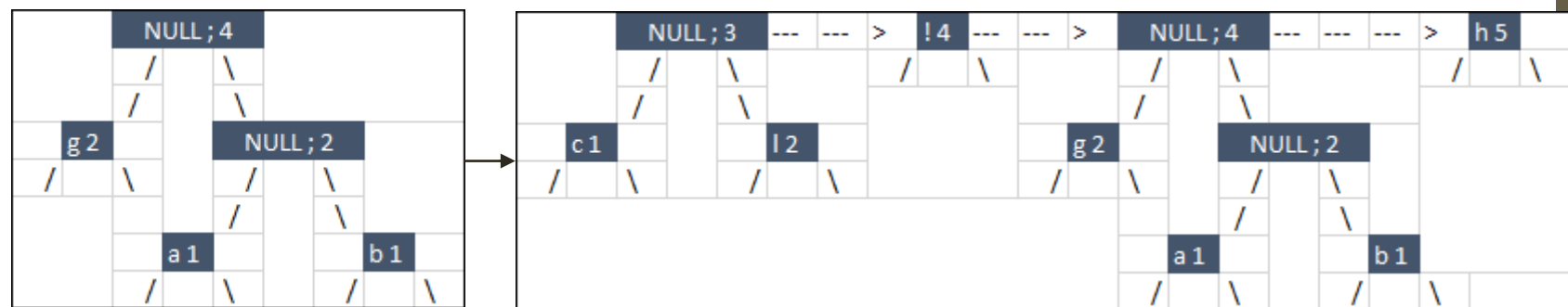
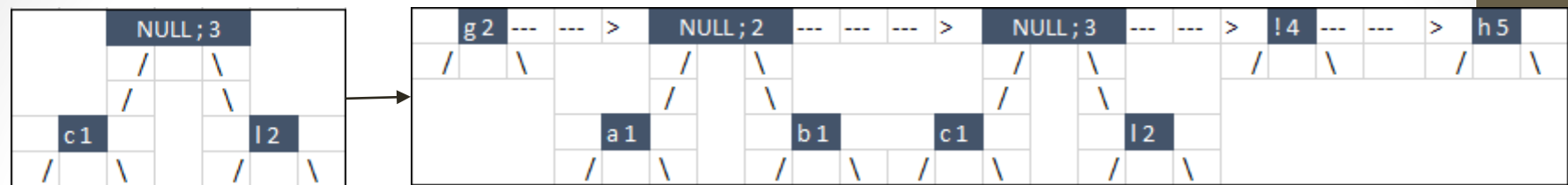
la frecuencia de este nodo padre es la suma de la frecuencia de sus 2 hijos:

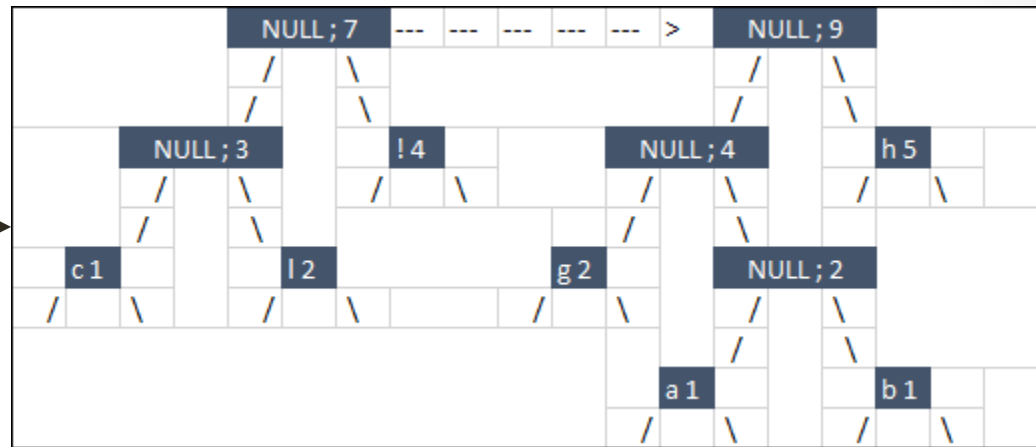
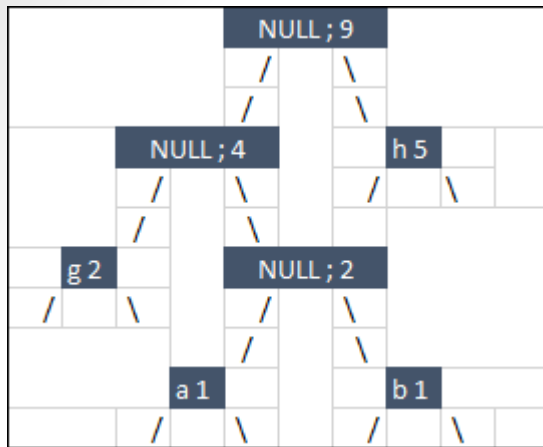


Este nuevo nodo se vuelve meter a la cola, y esta se ordena sola basado en la frecuencia:

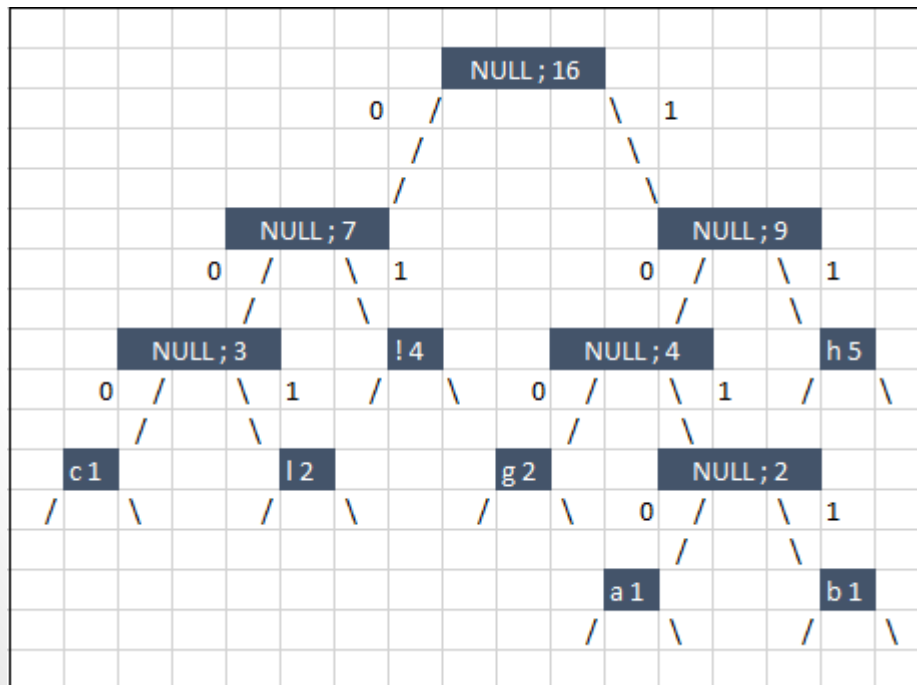


Se repite este proceso N veces:





Cuando quedan solo 2 nodos, se crea un padre que se convierte en la raíz de mi ArbolHuffman.



Cada letra tiene un código:

A	1010
B	1011
C	000
!	01
G	100
H	11
L	001

Y un mensaje como

a l ! ! c

Quedaría codificado así:

1010|001|0  
1|01|000|00

[Volver](#)



## **TAD Cadena**

### **Conjunto mínimo de datos**

- tipo, identifica el tipo de cadena que es.
- cadena, es lo que contiene la cadena.
- tam, es el tamaño de la cadena
- ident, es el tamaño de indentación.
- complete, dice si esta completa o no la cadena
- bases, tiene las bases con su frecuencia en la cadena

### **Operaciones del objeto**

- Cadena (tipo1,cadena1), Constructor de Cadena, le asigna el tipo y una cadena dada.
- getTipo(), Retorna el tipo de la cadena.
- getCadena(), Retorna la cadena.
- setTipo(tipo), Recibe un tipo por parámetro y cambia el que estaba antes.
- setCadena(cadena), Recibe una cadena por parámetro y cambia el que estaba antes.
- contarBases(), Cuenta las bases que tiene la cadena y retorna la cantidad.
- subCadenas(subCadena,bandera1), Encuentra una subCadena, si bandera1 es verdadero me cuenta las subCadenas que hay, si bandera1 es falso me enmascara esas subcadenas.

## **TAD CódigoGenetico**

### **Conjunto mínimo de datos**

- listaCadenas, vector de tipo cadena, contiene un conjunto de cadenas.
- basesTotales, mapa, contiene las bases de todas las secuencias con su correspondiente frecuencia.
- tree, ArbolHuffman, el arbol asociado a ese código genetico.

### **Operaciones del objeto**

- CodigoGenetico(), Constructor de CodigoGenetico.

- getListasCadenas(), Retorna la lista de cadenas.
- contarSecuencias(), Retorna la cantidad de secuencias que tiene almacenadas.
- cargarDatos(nombreArchivo), Carga las cadenas desde un archivo y las almacena.
- listaSecuencias(), Muestra el tipo de las cadenas y además cuantas bases tiene.
- buscarCadena(descripción), Busca una cadena con su descripción y la retorna.
- mostrarHistograma(descripción), Muestra el histograma de una cadena dada su descripción.
- guardarDatos(nombreArchivo), Guarda los datos que tiene almacenados.
- subCadenas(subCadena,bandera), Retorna la cantidad de subCadenas que se tiene.
- generarArbol(), Genera el correspondiente árbol de HuffMan de las secuencias.
- encode(), Se encarga de codificar (siguiendo el formato) los datos que se almacenan en código genético y posteriormente los guarda en un archivo binario.
- decode(), Se encarga de decodificar los datos que se almacenan en un archivo binario, con un formato establecido.

### **Operaciones Adicionales**

ingresarBases(), Retorna un vector con las bases ingresadas.

buscarBasePos(letra), Dada una letra busca en el vector la base . Si la encuentra retorna la posición, si no retorna -1.

load(), Imprime en pantalla la ayuda del comando load.

countM(), Imprime en pantalla la ayuda del comando count.

list\_sequences(), Imprime en pantalla la ayuda del comando list\_sequences

histogram(), Imprime en pantalla la ayuda del comando histogram.

is\_subsequence(), Imprime en pantalla la ayuda del comando is\_subsequence().

mask(), Imprime en pantalla la ayuda del comando mask.

save(), Imprime en pantalla la ayuda del comando save.

encode(), Imprime en pantalla la ayuda del comando encode.

decode(), Imprime en pantalla la ayuda del comando decode.

cargar(código,nombreArchivo), Es la función que se llama cuando se recibe el comando load, recibe CódigoGenetico y el nombre del archivo para cargarlo.

contar(código), Es la función que se llama cuando se recibe el comando count, recibe CódigoGenetico para saber de cual código genético debe saber la longitud de sus cadenas.

listaSecuencias(código), Es la función que se llama cuando se recibe el comando list\_sequences, recibe CódigoGenetico para mostrar las secuencias que pertenecen a ese CódigoGenetico.

subSecuencia(código,secuencia), Es la función que se llama cuando se recibe el comando is\_subsequence, recibe CódigoGenetico y el nombre del tipo de la secuencia. Me muestra en pantalla si esa sub secuencia existe en una secuencia cargada.

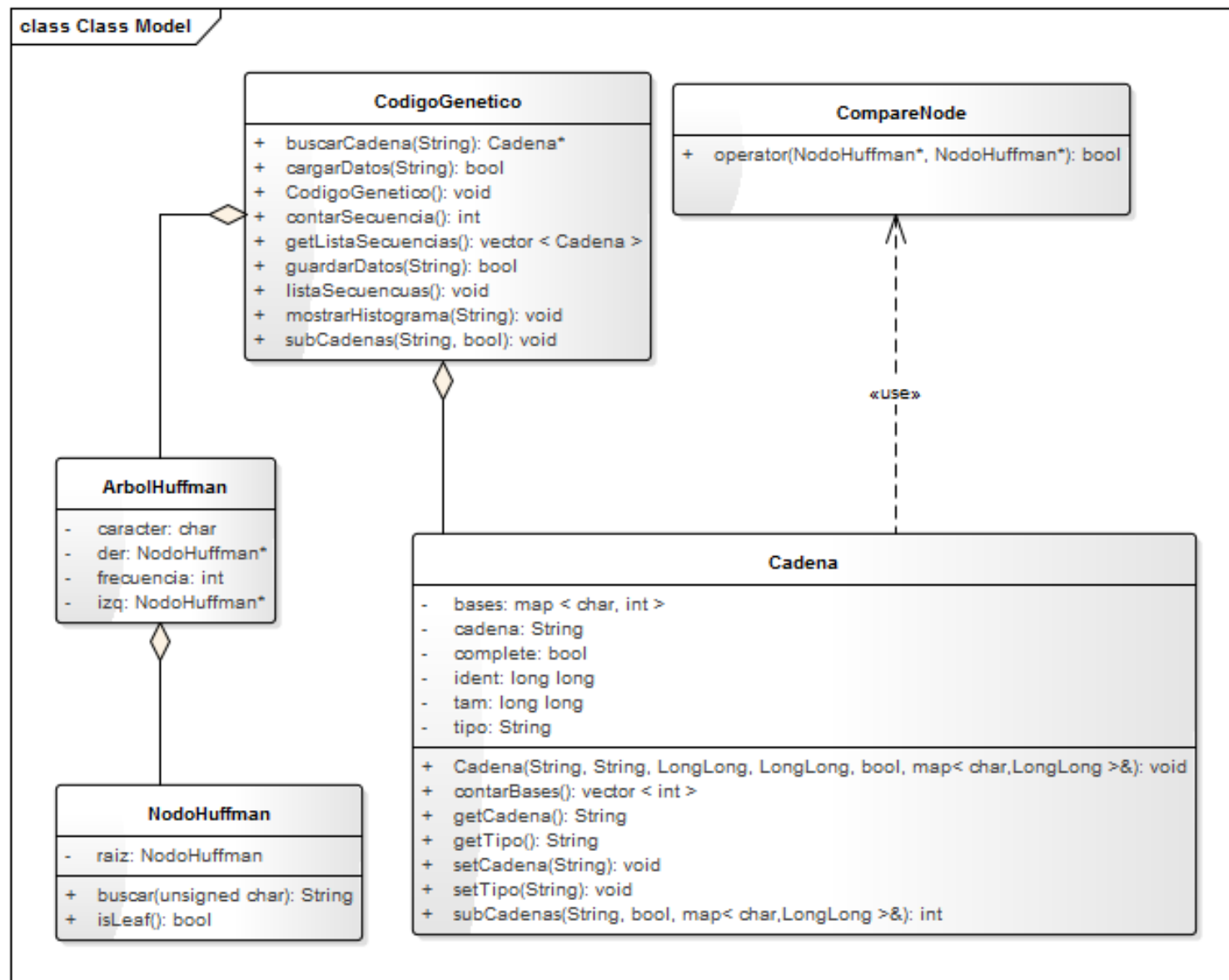
mask(código,secuencia), Es la función que se llama cuando se recibe el comando mask, recibe CódigoGenetico y el nombre del tipo de la secuencia. Muestra en pantalla la cantidad de cadenas que fueron enmascaradas.

guardar(código,nombreArchivo), Es la función que se llama cuando se recibe el comando save, recibe CódigoGenetico y el nombre del archivo al cual se va a mandar los datos y posteriormente guardarlos.

encode(), Es la función que se llama cuando se recibe el comando encode, recibe CódigoGenetico y el nombre del archivo al cual se va a guardar la codificación.

decode(), Es la función que se llama cuando se recibe el comando decode, recibe CódigoGenetico y el nombre del archivo al cual se va a cargar la codificación.

class Class Model



### ***Comentarios Primera entrega:***

Documento de diseño:

- El documento de diseño debería tener todo en un sólo archivo, para facilidad de lectura y que yo pueda ver a organización lógica de la concepción de la solución.
- En el diagrama de relación entre TADs sólo se incluye lo que es un TAD, es decir, que tiene datos Y operaciones asociadas. El main no es un TAD, clases auxiliares que solo provean métodos no son TADs.

Código fuente:

- El comando load no carga los archivos cuando le indico la extensión, con el nombre de archivo solo si lo hace.
- Hay que revisar la demora al listar las secuencias del archivo na\_arms.fa, parece estar afectado por la longitud del archivo y de las secuencias.
- En el histograma me debe aparecer el conteo del – también.
- El comando guarda el archivo agregándole la extensión .fa, lo que hace que quede con doble extensión.

### ***Correcciones Realizadas:***

Con respecto al documento de diseño, se unieron todos los elementos solicitados (Esquemático, diagrama de clases y Diseño de los TAD), y se corrigieron en el diagrama de relación de los TAD la aparición de elementos innecesarios.

Con respecto al código fuente, se llegó a un acuerdo en el que el enunciado no era lo suficientemente claro con respecto al tema de las extensiones de los archivos, cosa que se corregirá en la segunda entrega. Se agregó el conteo del carácter “-” al histograma y se explicó que la demora al momento de cargar se debe a que se genera todo lo necesario para el programa en la ejecución del comando, para qué sea más rápida la entrega de la información al usuario.