

**Estructuras de datos**

**Documentación**

Primera entrega

**Realizado por:**

**John García**

Ingeniería de sistemas

**Nicolás Peralta**

Ingeniería de sistemas

**Iván Ramírez**

Ingeniería de sistemas

**05 de Mayo del 2020**

Contenido

[**ESTRUCTURAS DE DATOS UTILIZADAS**](#_heading=h.gjdgxs) **3**

[1.1 Secuencia](#_heading=h.30j0zll) 3

[1.2 Sistema:](#_heading=h.1fob9te) 3

[**FUNCIONES Y PROCEDIMIENTOS**](#_heading=h.3znysh7) **4**

[2.1 menu\_aiuda](#_heading=h.2et92p0) 4

[2.2 menu\_ppal](#_heading=h.tyjcwt) 4

[2.3 histograma](#_heading=h.3dy6vkm) 4

[2.4 subsecuencia](#_heading=h.1t3h5sf) 4

[2.5 cargarArchivo](#_heading=h.4d34og8) 5

[2.6 guardar](#_heading=h.2s8eyo1) 5

[2.7 enmascarar](#_heading=h.17dp8vu) 6

[2.8 Conteo](#_heading=h.3rdcrjn) 6

[2.9 cantbases](#_heading=h.26in1rg) 6

[2.10 listar\_secuencias](#_heading=h.lnxbz9) 7

[**DIAGRAMAS Y MODELOS**](#_heading=h.35nkun2) **7**

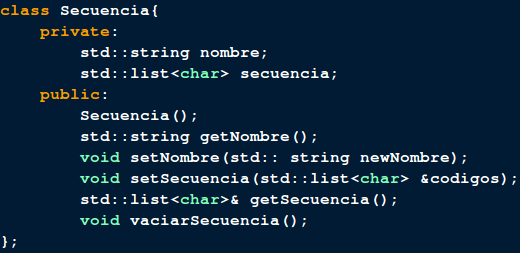
[3.1 Diagrama de Casos de Uso](#_heading=h.1ksv4uv) 7

[3.2 Diagrama de Clases](#_heading=h.44sinio) 8

# **ESTRUCTURAS DE DATOS UTILIZADAS**

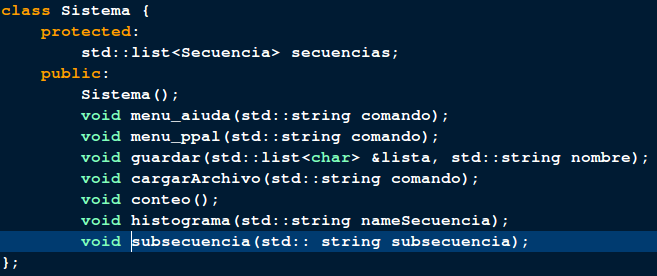
La estructura predilecta para el funcionamiento del sistema fue la lista anidada, siendo la única estructura provista por las librerías estándar. A continuación, se describen las estructuras de datos, y la manera en la que fueron utilizadas:

## 1.1 Secuencia



Es la estructura que maneja toda la información de las secuencias provistas por los archivos FASTA.

## 1.2 Sistema:



Será la estructura que jugará el papel de “middle man” en el sistema, teniendo todas las funcionalidades que utiliza el programa principal, y los datos referentes a las secuencias ya cargadas en el sistema.

# **FUNCIONES Y PROCEDIMIENTOS**

## 2.1 menu\_aiuda

Este procedimiento contiene toda la información de ayuda para el usuario, con respecto a los comandos ofrecidos por el sistema.

* **Parámetro de entrada:** cadena de texto con el comando deseado.
* **Prototipo:** void menu\_aiuda(string comando)

## 2.2 menu\_ppal

Este procedimiento contiene la correcta distribución del menú de opciones que el usuario podrá utilizar.

* **Parámetro de entrada:** cadena de texto con el comando deseado.
* **Prototipo:** void menu\_ppal(string comando):

## 2.3 histograma

El procedimiento presenta la frecuencia del código que compone la secuencia ingresada como parámetro.

* **Parámetro de entrada:** cadena de texto con el nombre de la secuencia deseada.
* **Prototipo:** void histograma (string nameSecuencias):

## 2.4 subsecuencia

El procedimiento analiza las secuencias cargadas en el sistema, y presenta el número de ocurrencias de una subsecuencia ingresada por el usuario.

* **Parámetro de entrada:** cadena de texto con la subsecuencia que se desea analizar.
* **Prototipo:** void subsecuencia(string subsecuencia):

## 2.5 cargarArchivo

* **Prototipo**: Void cargarArchivo(string comando):

* **Funcionalidad:** la función carga el archivo FASTA a la memoria
* **Descripción:** la función recibe un comando en el cual se encuentra la dirección del archivo que se va a cargar en memoria. Abre el archivo lo lee y guarda su información en la estructura de secuencias de la clase sistemas.

* **Uso de Parámetros:** string comando
* **Donde se usa:** en la función menu\_ppal
* **Retorno:** la función no tiene ningún retorno

## 2.6 guardar

* **Prototipo:** Void guardar(list<char> &lista, string nombre)
* **Funcionalidad:** la función guarda una secuencia en la lista de secuencias
* **Descripción:** la función recibe una lista la cual representa una secuencia, y el nombre de la secuencia, lo asigna a una estructura secuencia y lo agrega a la lista de secuencias.
* **Uso de Parámetros:** lista, nombre que se asignan a la estructura secuencia
* **Donde se usa:** en la función cargarArchivo
* **Retorno:** la función no tiene ningún retorno

## 2.7 enmascarar

Este procedimiento : Enmascara una secuencia dada por el usuario, si existe. Los elementos que pertenecen a la subsecuencia se enmascaran cambiando cada código por el código ’X’..

* **Parámetro de entrada:** cadena de texto con el comando deseado.
* **Prototipo:** void enmascarar (string comando):

## 2.8 Conteo

**Prototipo:** Void conteo()

**Funcionalidad:** la función muestra la cantidad de secuencias

**Descripción:** la función accede a la lista de secuencias y muestra la cantidad de secuencias

**Uso de Parámetros:** lista secuencias

**Donde se usa:** en la función menu\_ppa

**Retorno:** la función no tiene ningún retorno

## 2.9 cantbases

* **Prototipo:** int cantbases(list<char> lista)
* **Funcionalidad:** devuelve el numero de bases que se encuentran en una lista
* **Descripción:** la función recorre la lista y compara cada elemento con las bases y si es una base la suma a el contador finalmente cuando se termina de recorrer la lista devuelve el contador
* **Uso de Parámetros:** lista
* **Donde se usa:** en la función listar\_secunecias
* **Retorno:** la función retorna un 0 si no obtuvo ninguna base o las n bases que encontro.

## 2.10 listar\_secuencias

* **Prototipo:** Void listar\_secuencias()
* **Funcionalidad:** la función muestra las secuencias y el número de sus bases
* **Descripción:** la función accede a la lista de secuencias la recorre y por cada una de ellas muestra su descripción y la secuencia.
* **Uso de Parámetros:** lista secuencias
* **Donde se usa:** en la función menu\_ppa
* **Retorno:** la función no tiene ningún retorno

2.11 codificar

* **Prototipo:** Void codificar(string nombre\_archivo)
* **Funcionalidad:** La función codifica el archivo .fasta cargando previamente en el sistema.
* **Descripción:** El comando debe generar el archivo binario con la correspondiente codificación de Huffman en el formato descrito más arriba, almacenando en disco bajo el nombre nombre\_archivo.fabin .
* **Uso de Parámetros:** nombre que se le dara al archivo
* **Donde se usa:** en la función menu\_ppa
* **Retorno:** la función no tiene ningún retorno

2.12 listarfrecuencias

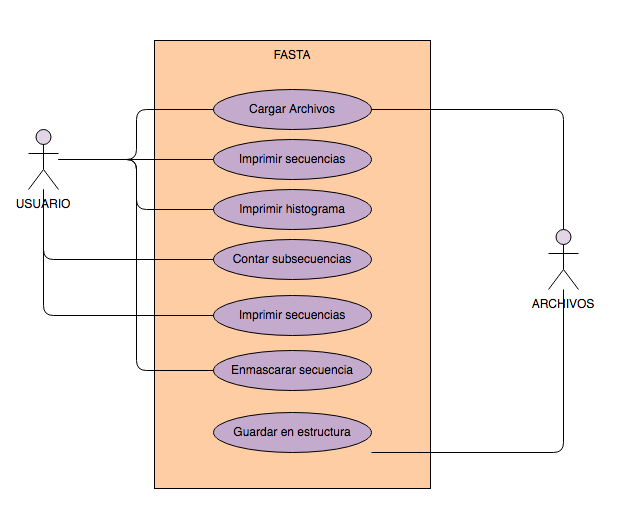
* **Prototipo:** Void listarFrecuencias()
* **Funcionalidad:** es una función auxiliar de la función codificar.
* **Descripción:** La función utiliza datos cargados en memoria para construir una lista de frecuencias.
* **Uso de Parámetros:** sin parametros.
* **Donde se usa:** en la función menu\_ppa
* **Retorno:** la función no tiene ningún retorno

2.13 algoritmoHuffman

* **Prototipo:** Void algoritmoHuffman()
* **Funcionalidad:** es una función auxiliar de la función codificar.
* **Descripción:** El comando genera una list de TablasHuffman que ayudarán a la codificación y luego estos datos se guardan en un árbol binario.
* **Uso de Parámetros:** sin parametros
* **Donde se usa:** en la función menu\_ppa
* **Retorno:** la función no tiene ningún retorno

# **DIAGRAMAS Y MODELOS**

## 3.1 Diagrama de Casos de Uso

****

## 3.2 Diagrama de Clases

