**Centro Educativo**

**Autorizado por ORT Uruguay**

**Analista Programador**

**Semestre I**

**Introducción a la computación**

**María Kapek**

**ByNarii**

**Equipo**

**Fabián Delmonte**

**Dardo De León**

TEMA:

Sistema que simula

el almacenamiento interno de datos

**Presentación del problema.**

Diseñar e implementar un sistema que permita simular la forma de almacenar internamente un valor. Debe permitir almacenar tanto valores numéricos (originalmente en cualquier base, que deberá indicarse) como una palabra o texto (utilizando un sistema específico de codificación, por ejemplo ASCII) y, en cada caso, sugerir las opciones posibles en cuanto a tamaño de estructura (byte, palabra o palabra doble) y forma de almacenamiento (números con o sin signo, número entero o fraccionario). Una vez seleccionadas se deberá mostrar su representación interna correspondiente a un conjunto de bits.

Consideraciones:

* Ingreso de un valor:
  + Permitir ingresar valores variados según la explicación.
  + Verificación del valor ingresado.
* Controles básicos que limiten errores en la selección de opciones:
  + Elementos gráficos controladores.
  + Mostrar información en pantalla según la selección realizada.
* Implementación del proceso de almacenamiento interno:
  + Mostrar el resultado en binario agrupado de a 8 bits.
  + Indicar si el valor no es almacenable en la estructura seleccionada.

**Introducción**

El propósito de esta aplicación es ilustrar la manera en la que una computadora almacena datos, tanto los ingresados por el usuario como los generados a partir de procesos internos.

Dentro de la computadora, la información se representa mediante el sistema de numeración binario, unos y ceros. La unidad básica de información es el bit (BInary digiT), es la unidad más pequeña y puede tomar solo uno de los valores antes mencionados. Los bits forman grupos de 8 que se denominan “byte”, por lo cual cada byte puede contener 8 dígitos binarios. A su vez, los grupos de dos bytes se denominan “palabra” y “palabra doble” a los de cuatro, (32 bits).

La computadora es capaz de construir, almacenar y representar distintos tipos de información. Entre otros, se distinguen dos tipos de representación, de los cuales se ocupa la presente aplicación: representación de textos y representación de valores numéricos.

La codificación va a facilitar y optimizar las tareas de almacenamiento, representación, comunicación y procesamiento de los datos. Algunas de los factores involucrados al momento de codificar dichos datos son, primero, que la memoria de la computadora es limitada, por lo que forzosamente tienen que acotarse las representaciones; segundo, las unidades funcionales del computador, están diseñadas para trabajar con cadenas de bits de cierto tamaño; tercero, los buses, que se encargan de la comunicación entre dichas unidades funcionales pueden transmitir simultáneamente un numero limitado de bits.

Como se dijo al principio, el objetivo de esta aplicación, es ilustrar la representación y almacenamiento de los datos teniendo en cuenta limitaciones de tamaño en el espacio destinado para tal fin, realizando las conversiones necesarias para realizar dicho almacenamiento habiendo posibilidad de desborde en caso de que los datos a almacenar superen el espacio disponible.

**Descripción del programa**

La aplicación presentará al usuario una interface como la que se ilustra en la figura 1, a través de la cual se le facilitarán al programa, los valores de los datos numéricos o alfanuméricos que se desea representar, así también como su signo y el sistema de numeración en el que están expresados.

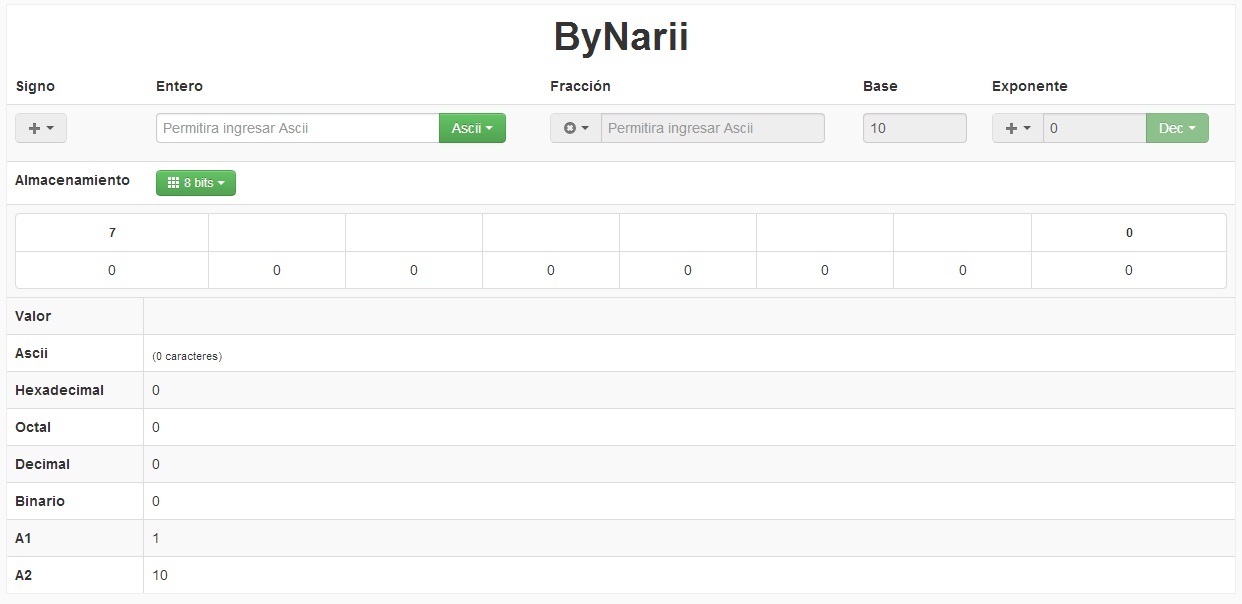


Figura 1

La interface está dividida a grandes rasgos en dos. La región ilustrada en la figura 2, es donde se realiza el ingreso de los datos.



Figura 2

En el menú desplegable bajo el título “Signo” (figura 3), se elige el signo del número que se desea almacenar. Bajo el título “Entero” se encuentra la casilla de texto que contendrá la parte entera del número a almacenar. Por medio del menú desplegable que se encuentra a la derecha de la casilla de texto, se puede elegir el sistema de numeración entre hexadecimal, decimal y octal, en el que estará expresado dicho número en caso de ser tal, o el tipo de datos ASCII en el caso de que se desee almacenar letras o palabras.



Figura 3

Bajo el título “Fracción” como se muestra en la figura 4, se encuentra un menú desplegable en el que se selecciona la habilitación o no, del cuadro de texto adyacente que permitirá en caso de estar habilitado, el ingreso de la parte decimal del número a almacenar. En caso de no habilitar la opción “Con Fracción”, la aplicación almacenará solo parte entera expresada con anterioridad.



Figura 4

La última parte de la región de ingreso de datos que se muestra en la figura 5, contiene en primera instancia la base del número a almacenar, que siempre estará fijada en 10. En seguida, bajo el título “Exponente” se encuentra un cuadro de texto en el que se ingresará el valor del mismo. También hay dos menús desplegables a un lado y otro del cuadro. En el primero se selecciona el signo del exponente y en el de más a la derecha, el sistema numérico en el que está expresado, ya sea hexadecimal, decimal u octal.



Figura 5

La segunda parte de la interface (figura 6), es la que se encarga de mostrar las conversiones y la forma en que se almacenarían.

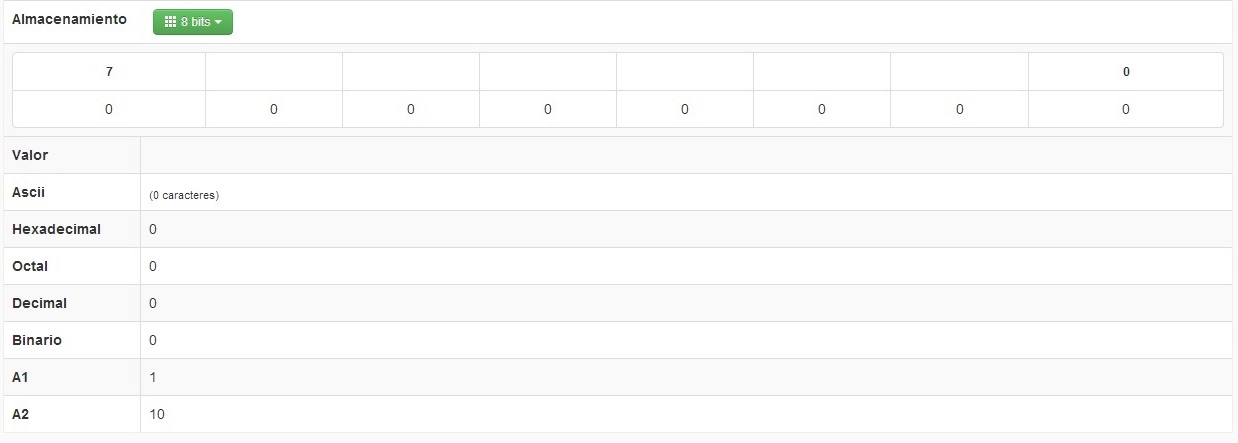


Figura 6

El primer apartado (figura 7), es el que muestra la forma en la que se almacenaría en memoria el numero ingresado. En el primer renglón se hace una separación del espacio total disponible en grupos de 8 bits (1 byte). Cada una de las casillas que se observan en el segundo renglón representa un bit y la cantidad de ellas depende de la precisión seleccionada en el menú desplegable al lado del título “Almacenamiento”. Las opciones son: “Simple” y “Doble”. En la primera se dispone de 32 bits que se dividen en 1 bit de signo, 8 de exponente y 23 de mantisa. En la segunda se dispone de 64 bits, los cuales se dividen en 1 bit de signo, 11 de exponente y 52 de mantisa.



Figura 7

En la figura 8 se indica lo que sucede cuando se selecciona “Con fracción” en el menú desplegable de la figura 4. El primer bit desde la izquierda del segundo renglón del almacenamiento queda reservado para el signo del número en coma flotante que se ingresa; además, a partir de la segunda casilla cambia el color de fondo, indicando las posiciones destinadas al almacenamiento del exponente. El resto de las casillas quedan destinadas al almacenamiento de la mantisa. La distribución de estas regiones siempre es la misma mientras el número a almacenar sea del tipo coma flotante, lo que sí cambian son las cantidades de bits que se destinan a cada una, dependiendo de la selección de precisión que se haya hecho.

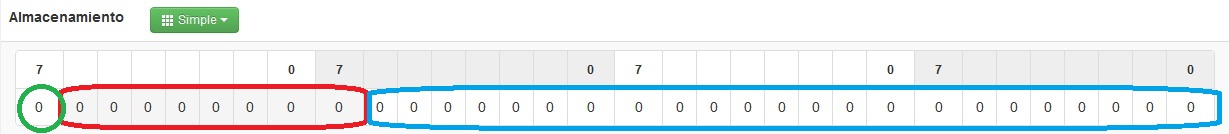


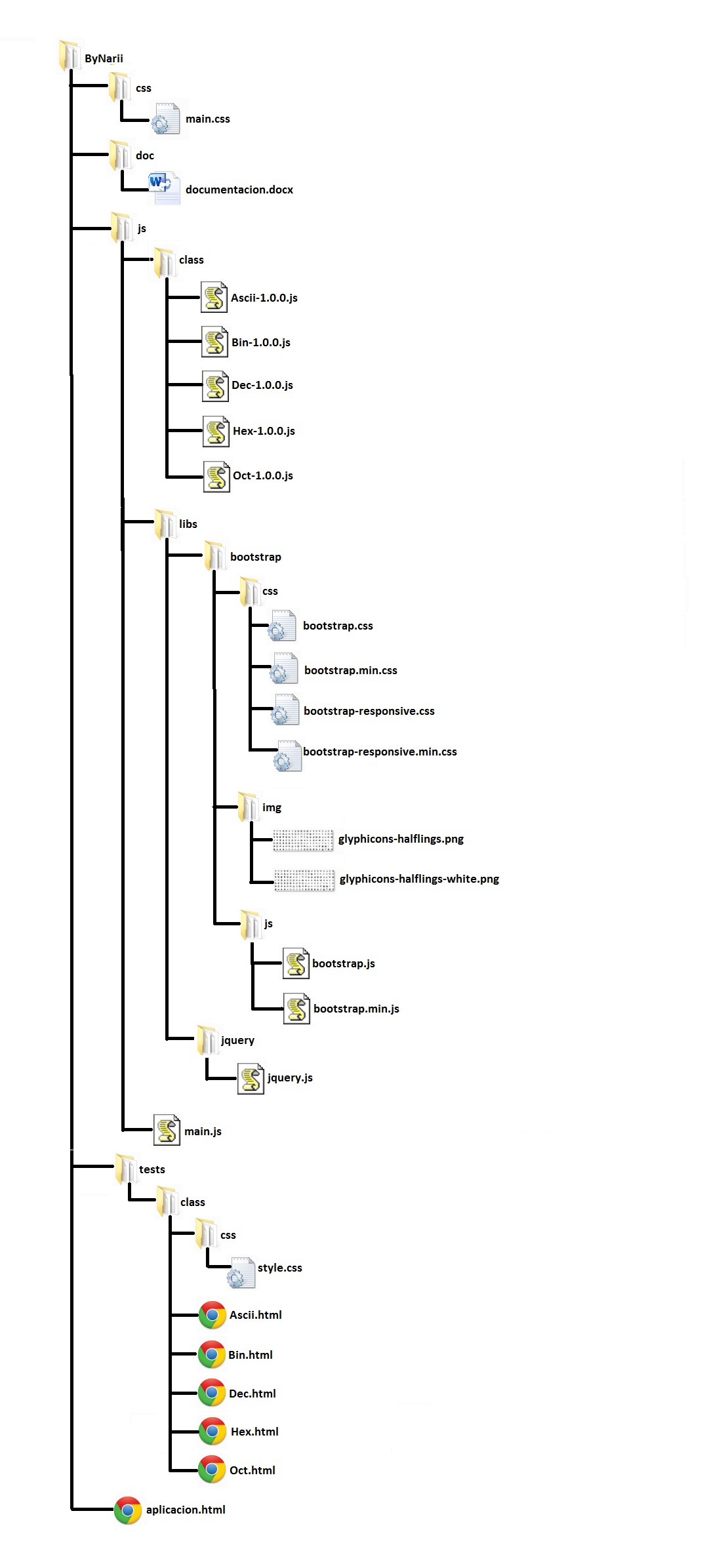
Figura 8

El apartado debajo del área de almacenamiento (figura 9) contiene las conversiones que se están realizando a medida que el usuario va ingresando los valores en los campos de la parte superior. En “Valor” se muestran los caracteres que el usuario está ingresando. En los campos ASCII, Hexadecimal, Octal, Decimal y Binario se pueden ver las conversiones de esos caracteres a cada uno de los respectivos sistemas de numeración. En A1 se muestra el resultado de realizar sobre los datos ingresados el complemento a 1, consistente en completar el número devuelto de la conversión a binario con ceros hasta llenar el espacio de almacenamiento disponible y luego cambiar cada cero por un uno y viceversa. El campo A2 se refiere al resultado de un proceso similar al descrito antes, con el añadido de que al resultado de A1 se le suma uno. Cabe agregar que estos dos procesos tienen validez cuando el número ingresado es negativo.



Figura 9

**Árbol de directorios**

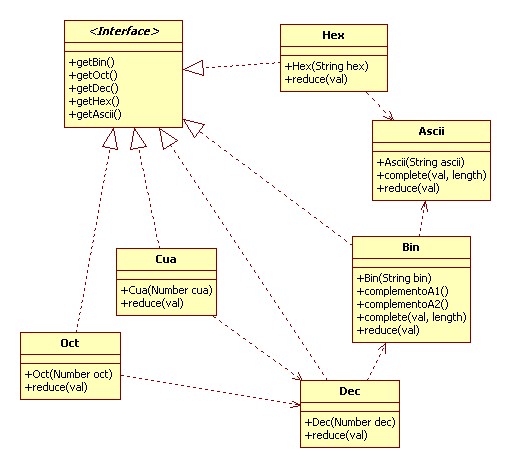


**Descripción de archivos y directorios**

El directorio **ByNarii** es el directorio raíz del proyecto, dentro de el se encuentran las carpetas **CSS**, **doc**, **js**, **tests** y el archivo **aplicacion.html** que es el que carga la interfaz del programa. Dentro de la carpeta **CSS** se encuentra el archivo **main.css** que es que contiene los estilos de la interfaz de la aplicación. La carpeta **doc**, contiene la documentación presente. El directorio **js** tiene dos subdirectorios, **class** y **libs**, y también contiene el archivo **main.js**. Este archivo es el que administra el funcionamiento de la interfaz, generando objetos contenedores que agrupan las variables a utilizar, y cargan en variables los valores de los elementos del DOM; maneja también los eventos disparados por las posibles elecciones del usuario, y las excepciones o errores que pudieran producirse; se encarga además de representar los datos ingresados por el usuario y generar en la interfaz el diseño necesario para tal función. El subdirectorio **class** contiene las clases **Ascii-1.0.0.js, Bin-1.0.0.js, Dec-1.0.0.js, Hex-1.0.0.js, y Oct-1.0.0.js**. La clase **Ascii** es la que se encarga justamente de manejar los valores ASCII, recibe una cadena de texto, recorre todos los caracteres que la componen, toma el valor Unicode de cada uno, lo convierte a decimal y almacena internamente su valor en colecciones cuyos elementos estarán en las bases 2, 8, 16 y Unicode. Las demás clases se encargan de manejar los valores binarios, decimales, hexadecimales y octales respectivamente. La carpeta **libs** contiene las carpetas **bootstrap** y **jquery**; la primera contiene los archivos necesarios para la implementación de Bootstrap y la segunda contiene la librería JQuery.

La carpeta **tests** contiene la carpeta **class**, y ésta, a los archivos de prueba de las diferentes clases descritas anteriormente y la carpeta **css** en la que se encuentra el archivo **style.css**, que define los estilos de los archivos de prueba.

**A continuación se muestra una imagen que muestra la relación entre las clases y la interfaz.**



**Descripción de Herramientas**

Para el desarrollo de esta aplicación se utilizó como lenguaje JavaScript y se hizo uso de la librería JQuery.

JQuery es una librería de Javascript que se utiliza para simplificar la manera de interactuar con los documentos HTML y permite manipular el DOM, así también como manejar eventos entre otras cosas. También ofrece funcionalidades basadas en JavaScript que agilitan el desarrollo ya que éstas requieren menos código que si fueran desarrolladas en JavaScript puro.

Para el front-end, se utilizó Bootstrap. Se trata de un framework de twitter que permite crear interfaces web con CSS y Javascript que adaptan la interfaz dependiendo del tamaño del dispositivo en el que se visualice, en otras palabras, se adapta automáticamente al tamaño de una computadora o de una Tablet sin que el usuario tenga que hacer nada, este tipo de diseño se conoce como “adaptativo” o Responsive Design.

La aplicación se realizó enteramente en NetBeans, un entorno de desarrollo integrado (IDE), que permite escribir, compilar, depurar y ejecutar programas. Se presenta en varias versiones dependiendo del tipo de desarrollo y lenguajes en los que se desea desarrollar, para la realización de este proyecto se utilizó el PHP Pack, enfocado especialmente al desarrollo web.

**Bibliografía**

* Organización de Computadoras – Andrew Tanembaum.
* Representación en Punto Flotante - Minaya Villasana

<http://prof.usb.ve/mvillasa/compcient/puntoflotante.pdf>

* Aritmética en coma flotante - Daniel Mozos, Marcos Sánchez-Elez, José Luis Risco <http://www.fdi.ucm.es/profesor/mozos/aec/aritm_pf.pdf>
* Wikipedia

**Índice**

Presentación del problema ---------------------------------------------------- página 3

Introducción ------------------------------------------------------------------------ página 4

Descripción del programa ------------------------------------------------------ página 5

Árbol de directorios -------------------------------------------------------------- página 9

Descripción de archivos y directorios -------------------------------------- página 10

Descripción de herramientas ------------------------------------------------ página 11

Bibliografía ------------------------------------------------------------------------ página 13