|  |  |  |
| --- | --- | --- |
| **CARRERA:**  Ingeniería de Software | **GUÍA**  No. 03 | **TIEMPO ESTIMADO:**  1h y 20 min. |
| **ASIGNATURA:**  Estructura de datos  NRC: 2967 | **FECHA DE ELABORACION:** 26-01-2020  **SEMESTRE**: septiembre 2019 – febrero 2020 | |
| **TÍTULO:**  Arboles Binarios | **DOCENTE:** Ing. Fernando Solis | |

**OBJETIVO**

Comprender el significado de Árboles y sus diferentes tipos para la utilización de un proyecto sencillo, como un traductor.

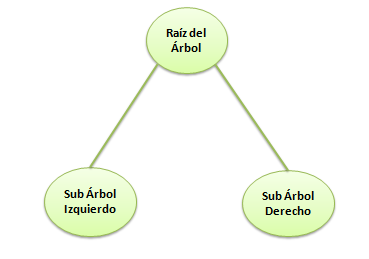
**INSTRUCCIONES**

**Arboles Binarios**

Los árboles binarios son estructuras de datos muy similares a las listas doblemente enlazadas, en el sentido que tienen dos punteros que apuntan a otros elementos, pero no tienen una estructura lógica de tipo lineal o secuencial como aquellas, sino ramificada.  Tienen aspecto de árbol, de ahí su nombre.

Un árbol binario es una estructura de datos no lineal en la que cada nodo puede apuntar a uno o máximo a dos nodos. También se suele dar una definición recursiva que indica que es una estructura compuesta por un dato y dos árboles. Esto son definiciones simples. Este tipo de árbol se caracteriza porque tienen un vértice principal y de él se desprende dos ramas. La rama izquierda y la rama derecha a las que también se les conoce como subárboles.

Una representación gráfica de la estructura general de un árbol binario se puede visualizar en la siguiente imagen:



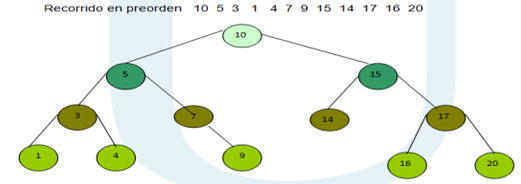
La rama izquierda y la derecha también son dos árboles binarios. El Vértice principal se denomina raíz y cada una de las ramas se puede denominar subárbol izquierdo y subárbol derecho.

#### Formas de Recorrer un árbol binario

Los árboles binarios, son estructuras de datos no lineales, son considerados como estructuras jerárquicas y como tal su forma de recorrerlos difiere sustancialmente en comparación con las listas enlazadas que son estructuras de datos de tipo lineal. En ese orden de ideas, el recorrido de un árbol binario se lleva a cabo en tres sentidos: Preorden, Inorden y Postorden. A continuación se detalla cada caso.

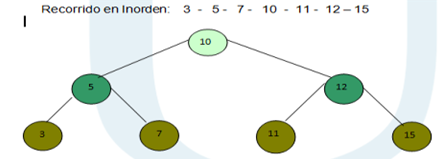
**Recorrido en  Preorden:**

Recorrer un árbol en preorden consiste en primer lugar, examinar el dato del nodo raíz, posteriormente se recorrer el subárbol izquierdo en preorden y finalmente se recorre el subárbol derecho en preorden. Esto significa que para cada subárbol se debe conservar el recorrido en preorden, primero la raíz, luego la parte izquierda y posteriormente la parte derecha.



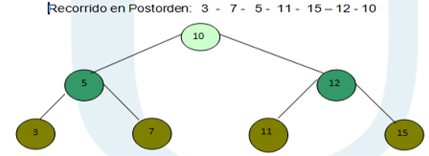
Recorrido en Inorden

Recorrer un árbol en Inorden consiste en primer lugar en recorrer el subárbol izquierdo en Inorden, luego se examina el dato del nodo raíz, y finalmente se recorre el subárbol derecho en  Inorden. Esto significa que para cada subárbol se debe conservar el recorrido en Inorden, es decir, primero se visita la parte izquierda, luego la raíz y posteriormente la parte derecha.



**Recorrido en Postorden:**

Recorrer un árbol en Postorden consiste en primer lugar en recorrer el subárbol izquierdo en Postorden, luego se recorre el subárbol derecho en Postorden y finalmente se visita el nodo raíz. Esto significa que para cada subárbol se debe conservar el recorrido en Postorden, es decir, primero se visita la parte izquierda, luego la parte derecha y por último la raíz.



**ACTIVIDADES**

1. **Ubicación de recursos**
2. El grupo esta conformado por Maria Belen Ceron y Alex Chicaiza
3. Se utilizo Visual Studio 2019
4. Para transformar el texto a .wav se utilizó la herramienta Balabolka.
5. **Planteamiento del problema**

Escribir un programa que pueda traducir palabras de ingles al español y viceversa, mostrar en pantalla la traducción y además tener la capacidad de reproducir la pronunciación de la palabra.

1. **Entregable (s)**

**Clase ABBI.h**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: ABBI.h

\* Modified: Sunday, January 19, 2020 4:00:49 PM

\* Purpose: Declaration of the class ABBI

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<string>

#if !defined(\_\_Traducctor\_Arbol\_Binario\_ABBI\_h)

#define \_\_Traducctor\_Arbol\_Binario\_ABBI\_h

class ABBI

**{**

public**:**

ABBI**(**std**::**string newIngles**,** std**::**string newEspanol**);**

**~**ABBI**();**

void insertarNodo**(**std**::**string newIngles**,** std**::**string newEspanol**,** ABBI**\*&** arbol**);**

void imprimirNodo**(**void**);**

void recorridoIn**(**ABBI**\*** arbol**);**

void recorridoPos**(**ABBI**\*** arbol**);**

void recorridoPre**(**ABBI**\*** arbol**);**

std**::**string buscar**(**ABBI**\*** arbol**,** std**::**string palabra**);**

//void imprimirArbol(ABBI\* arbol, int n);

protected**:**

private**:**

std**::**string ingles**;**

std**::**string espanol**;**

ABBI**\*** izquierda**;**

ABBI**\*** derecha**;**

**};**

#endif

**Clase ABBI.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: ABBI.cpp

\* Modified: Sunday, January 19, 2020 4:00:49 PM

\* Purpose: Implementation of the class ABBI

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "ABBI.h"

#include <string>

#include <iostream>

////////////////////////////////////////////////////////////////////////

// Name: ABBI::ABBI(char\* newIngles, char\* newEspanol)

// Purpose: Implementation of ABBI::ABBI()

// Parameters:

// - newIngles

// - newEspanol

// Return:

////////////////////////////////////////////////////////////////////////

ABBI**::**ABBI**(**std**::**string newIngles**,** std**::**string newEspanol**)**

**{**

**this->**ingles **=** newIngles**;**

**this->**espanol **=** newEspanol**;**

**this->**izquierda **=** **NULL;**

**this->**derecha **=** **NULL;**

**}**

////////////////////////////////////////////////////////////////////////

// Name: ABBI::~ABBI()

// Purpose: Implementation of ABBI::~ABBI()

// Return:

////////////////////////////////////////////////////////////////////////

ABBI**::~**ABBI**()**

**{**

// TODO : implement

**}**

////////////////////////////////////////////////////////////////////////

// Name: ABBI::insertarNodo(char\* ingles, char\* espanol, ABBI\*& arbol)

// Purpose: Implementation of ABBI::insertarNodo()

// Parameters:

// - ingles

// - espanol

// - arbol

// Return: void

////////////////////////////////////////////////////////////////////////

void ABBI**::**insertarNodo**(**std**::**string newIngles**,** std**::**string newEspanol**,** ABBI**\*&** arbol**)**

**{**

**if** **(**arbol **==** **NULL)** **{**

ABBI**\*** nuevo **=** **new** ABBI**(**newIngles**,**newEspanol**);**

arbol **=** nuevo**;**

//std::cout << "ingreso";

**}**

**else** **{**

**if** **(**newIngles**.**substr**(**0**,**1**)** **<** arbol**->**ingles**.**substr**(**0**,**1**))** **{**

**return** insertarNodo**(**newIngles**,**newEspanol**,** arbol**->**izquierda**);**

**}**

**else** **if** **(**newIngles**.**substr**(**0**,** 1**)** **>=** arbol**->**ingles**.**substr**(**0**,** 1**))** **{**

**return** insertarNodo**(**newIngles**,** newEspanol**,**arbol**->**derecha**);**

**}**

**}**

**}**

////////////////////////////////////////////////////////////////////////

// Name: ABBI::imprimirNodo()

// Purpose: Implementation of ABBI::imprimirNodo()

// Return: void

////////////////////////////////////////////////////////////////////////

void ABBI**::**imprimirNodo**(**void**)**

**{**

**if** **(this->**ingles**.**compare**(NULL))** **{**

**return;**

**}**

**else** **{**

std**::**cout **<<** "Dato del nodo: " **<<** **this->**ingles**;**

**return;**

**}**

**}**

////////////////////////////////////////////////////////////////////////

// Name: ABBI::recorridoIn(ABBI\* arbol)

// Purpose: Implementation of ABBI::recorridoIn()

// Parameters:

// - arbol

// Return: void

////////////////////////////////////////////////////////////////////////

void ABBI**::**recorridoIn**(**ABBI**\*** arbol**)**

**{**

**if** **(**arbol **!=** **NULL)** **{**

recorridoIn**(**arbol**->**izquierda**);**

std**::**cout **<<** arbol**->**ingles **<<** " "**<<** arbol**->**espanol**<<**std**::**endl**;**

recorridoIn**(**arbol**->**derecha**);**

**}**

**}**

////////////////////////////////////////////////////////////////////////

// Name: ABBI::recorridoPos(ABBI\* arbol)

// Purpose: Implementation of ABBI::recorridoPos()

// Parameters:

// - arbol

// Return: void

////////////////////////////////////////////////////////////////////////

void ABBI**::**recorridoPos**(**ABBI**\*** arbol**)**

**{**

**if** **(**arbol **!=** **NULL)** **{**

recorridoPos**(**arbol**->**izquierda**);**

recorridoPos**(**arbol**->**derecha**);**

std**::**cout **<<** arbol**->**ingles **<<** " " **<<** arbol**->**espanol **<<** std**::**endl**;**

**}**

**}**

////////////////////////////////////////////////////////////////////////

// Name: ABBI::recorridoPre(ABBI\* arbol)

// Purpose: Implementation of ABBI::recorridoPre()

// Parameters:

// - arbol

// Return: void

////////////////////////////////////////////////////////////////////////

void ABBI**::**recorridoPre**(**ABBI**\*** arbol**)**

**{**

**if** **(**arbol **!=** **NULL)** **{**

std**::**cout **<<** arbol**->**ingles **<<** " " **<<** arbol**->**espanol **<<** std**::**endl**;**

recorridoPre**(**arbol**->**izquierda**);**

recorridoPre**(**arbol**->**derecha**);**

**}**

**}**

////////////////////////////////////////////////////////////////////////

// Name: ABBI::buscar(ABBI\* arbol)

// Purpose: Implementation of ABBI::buscar()

// Parameters:

// - arbol

// Return: void

////////////////////////////////////////////////////////////////////////

std**::**string ABBI**::**buscar**(**ABBI**\*** arbol**,**std**::**string palabra**)**

**{**

**if** **(**arbol **==** **NULL)** **{**

**return** " "**;**

**}**

**else** **if** **(** palabra**.**substr**(**0**,**1**)<** arbol**->**ingles**.**substr**(**0**,**1**))** **{**

**if** **(**palabra**.**compare**(**arbol**->**ingles**)** **==** 0**)** **{**

std**::**cout **<<** "\n\tTRADUCCION: " **<<** arbol**->**espanol **<<** std**::**endl**;**

**return** arbol**->**espanol**;**

**}**

**else** **{**

**return** buscar**(**arbol**->**izquierda**,** palabra**);**

**}**

**}**

**else** **if** **(**palabra**.**substr**(**0**,**1**)** **>=** arbol**->**ingles**.**substr**(**0**,**1**))** **{**

**if** **(**palabra**.**compare**(**arbol**->**ingles**)** **==** 0**)** **{**

std**::**cout **<<** "\n\tTRADUCCION: " **<<** arbol**->**espanol **<<** std**::**endl**;**

**return** arbol**->**espanol**;**

**}**

**else** **{**

**return** buscar**(**arbol**->**derecha**,** palabra**);**

**}**

**}**

**}**

**Clase Cifrado.h**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Cifrado.h

\* Modified: Sunday, January 24, 2020 3:41:28 PM

\* Purpose: Declaration of the class Cifrado

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#if !defined(\_\_Cifrado\_Cifrado\_h)

#define \_\_Cifrado\_Cifrado\_h

#include <iostream>

#include <stdio.h>

#include <windows.h>

#include <conio.h>

#include<string>

**using** **namespace** std**;**

class Cifrado

**{**

public**:**

void encriptado**();**

void desencriptado**();**

private**:**

FILE**\*** archivo**,** **\*** nuevo**;**

**};**

#endif

**Clase Cifrado.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Cifrado.cpp

\* Modified: Sunday, January 24, 2020 3:41:28 PM

\* Purpose: Implementation of the class Cifrado

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "Cifrado.h"

////////////////////////////////////////////////////////////////////////

// Name: Cifrado::encriptado()

// Purpose: Implementation of Cifrado::encriptado()

// Return: void

////////////////////////////////////////////////////////////////////////

void Cifrado::encriptado() {

char car;

archivo = fopen("solucion.txt", "rb");

nuevo = fopen("1SolucioEncriptado.txt", "wb");

if (nuevo == NULL) {

cout << "Error al abrir el archivo nuevo\n";

}

if (archivo == NULL) {

cout << "Error al abrir el archivo\n";

}

while (fread(&car, 1, 1, archivo) != 0) {

car += 20;

fwrite(&car, 1, 1, nuevo);

}

fclose(nuevo);

fclose(archivo);

}

////////////////////////////////////////////////////////////////////////

// Name: Cifrado::desencriptado()

// Purpose: Implementation of Cifrado::desencriptado()

// Return: void

////////////////////////////////////////////////////////////////////////

void Cifrado::desencriptado() {

char car;

archivo = fopen("1SolucioEncriptado.txt", "rb");

nuevo = fopen("2SolucionDesncriptar.txt", "wb");

if (nuevo == NULL) {

cout << "Error al abrir el archivo nuevo\n";

}

if (archivo == NULL) {

cout << "Error al abrir el archivo\n";

}

while (fread(&car, 1, 1, archivo) != 0) {

car -= 20;

fwrite(&car, 1, 1, nuevo);

}

fclose(nuevo);

fclose(archivo);

}

**Clase Ingresar.h**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Universidad de las Fuerzas Armadas ESPE

\* Alex Chicaiza

\* Maria Belen Ceron

\* NRC: 2967

\* Fecha creacion: 19/01/2020

\* Fecha ultima modificacion: 26/01/2020

\* Docente: Ing. Fernando Solis

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

#include<stdlib.h>

#include<conio.h>

#include<string.h>

#include<stdio.h>

**using** **namespace** std**;**

class Ingreso**{**

public**:**

int ingresarEnteros**(**char **\*);**

char **\***ingresarEnterosChar**(**char **\*);**

string ingresarLetras**(**string**);**

**};**

/\*\*

**@brief** ingresarEnteros

**@param** \*msg

**@return** numero

\*/

int Ingreso **::** ingresarEnteros**(**char **\***msg**){**

bool bandera**=true;**

char caracter**[]=**""**;**

int contMenos**=**0**;**

int i**=**0**;**

int numero**=**0**;**

printf**(**"\n%s"**,**msg**);**

**do{**

caracter**[**0**]=**\_getch**();**

**if((**caracter**[**0**]>=**48 **&&** caracter**[**0**]<=**57**)){**

printf**(**"%c"**,**caracter**[**0**]);**

numero**=**numero**\***10**+**atoi**(**caracter**);**

i**++;**

**}else{**

**if(**caracter**[**0**]==**13**){**

bandera**=false;**

**}else{**

**if(**caracter**[**0**]==**8 **&&** i**>**0**){**

printf**(**"\b \b"**);**

i**--;**

numero**=**numero**/**10**;**

**if(**i**==**0**){**

contMenos**=**0**;**

numero**=**0**;**

**}**

**}**

**}**

**}**

**}while(**bandera**);**

**if(**contMenos**>**0**)**

numero**=**numero**\*(-**1**);**

**return** numero**;**

**}**

/\*\*

**@brief** ingresarLetras

**@param** \*msg

**@return** palabra

\*/

string Ingreso **::**ingresarLetras**(**string msg**){**

cout **<<** "\n\t" **<<** msg**;**

bool flag **=** **true;**

char caracter**[]** **=** ""**;**

string lectura**=**""**;**

int i **=** 0**;**

**while** **(**flag**)** **{**

caracter**[**0**]** **=** \_getch**();**

**if** **((**caracter**[**0**]** **>=** 65 **&&** caracter**[**0**]** **<=** 90**)** **||** **(**caracter**[**0**]** **>=** 97 **&&** caracter**[**0**]** **<=** 122**)** **||** caracter**[**0**]** **==** 32**)**

**{**

std**::**cout **<<** caracter**[**0**];**

lectura**+=** caracter**[**0**];**

i**++;**

**}**

**else**

**{**

**if** **(**caracter**[**0**]** **==** 13**)**

**{**

flag **=** **false;**

**}**

**else**

**{**

**if** **(**caracter**[**0**]** **==** 8 **&&** i **>** 0**)**

**{**

lectura **=** lectura**.**substr**(**0**,**lectura**.**length**()-**1**);**

std**::**cout **<<** "\b \b"**;**

i**--;**

**}**

**}**

**}**

**}**

//\*(lectura + i) = '\0';

**return** lectura**;**

**}**

char **\*** Ingreso **::** ingresarEnterosChar**(**char **\***msg**){**

bool bandera**=true;**

char caracter**[]=**""**;**

int contMenos**=**0**;**

int i**=**0**;**

int numero**=**0**;**

printf**(**"\n%s\n"**,**msg**);**

**do{**

caracter**[**0**]=**\_getch**();**

**if((**caracter**[**0**]>=**48 **&&** caracter**[**0**]<=**57**)){**

printf**(**"%c"**,**caracter**[**0**]);**

numero**=**numero**\***10**+**atoi**(**caracter**);**

i**++;**

**}else{**

**if(**caracter**[**0**]==**13**){**

bandera**=false;**

**}else{**

**if(**caracter**[**0**]==**8 **&&** i**>**0**){**

printf**(**"\b \b"**);**

i**--;**

numero**=**numero**/**10**;**

**if(**i**==**0**){**

contMenos**=**0**;**

numero**=**0**;**

**}**

**}**

**}**

**}**

**}while(**bandera**);**

**if(**contMenos**>**0**)**

numero**=**numero**\*(-**1**);**

char**\*** cad **=** **new** char**;**

sprintf**(**cad**,**"%i"**,**numero**);**

**return** cad**;**

**}**

**Clase Main**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Universidad de las Fuerzas Armadas ESPE

\* Alex Chicaiza

\* Maria Belen Ceron

\* NRC: 2967

\* Fecha creacion: 19/01/2020

\* Fecha ultima modificacion: 26/01/2020

\* Docente: Ing. Fernando Solis

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Module: Musica.cpp

\* Modified: Sunday, January 19, 2020 4:00:49 PM

\* Purpose: Implementation of the class ABBI

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <Windows.h>

#include <MMSystem.h>

#include <fstream>

#include <string>

#include <fstream>

#include <vector>

#include <stdio.h>

#include <conio.h>

#include <string.h>

#include "Ingresar.h"

#include "Cifrado.h"

#include "ABBI.h"

#define ARRIBA 72

#define IZQUIERDA 75

#define DERECHA 77

#define ABAJO 80

#define ENTER 13

**using** **namespace** std**;**

int cont **=** 16**;**

void gotoxy**(**short posicionx**,** short posiciony**);**

void dibujarCuadro**(**int x**,**int y**);**

void menu**();**

void reproducir**(**string cancion**);**

void leerArchivo**();**

void traductorInglesEspanol**();**

void traductorEspanolIngles**();**

void opcionSonido**(**string ingles**,** string espanol**);**

void empezarProgramas**(**string nombre**);**

void solucionTxt**();**

ABBI**\*** arbolIngles **=** **NULL;**

ABBI**\*** arbolEsp **=** **NULL;**

int main**()**

**{**

system**(**"Color 3F"**);**

leerArchivo**();**

solucionTxt**();**

system**(**"pause"**);**

int a**,** b**,** c**,** n **=** 0**,** letra **=** 39**,** pos **=** 1**,** cont **=** 39**,** aux**,** cont1 **=** 39**,** auxg**;**

char t**[**50**]** **=** "BIENVENIDO AL TRADUCTOR INGLES - ESPANOL"**,** auxt**[**39**]** **=** " "**;**

gotoxy**(**0**,** 0**);**

cout **<<** "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**;**

gotoxy**(**0**,** 2**);**

cout **<<** "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**;**

**do** **{**

**for** **(**a **=** 0**;** a **<** 30**;** a**++)** **{**

aux **=** pos**;**

**for** **(**b **=** 39**;** b **>** cont**;** b**--)** **{**

gotoxy**(**pos**,** 1**);**

cout **<<** t**[**b**];**

menu**();**

pos**--;**

**}**

aux**++;**

pos **=** aux**;**

cont**--;**

Sleep**(**100**);**

**if** **(**a **==** 40**)** **{**

**break;**

**}**

**}**

**for** **(**a **=** 3**;** a **<** 70**;** a**++)** **{**

gotoxy**(**a **-** 1**,** 1**);**

cout **<<** " "**;**

gotoxy**(**a**,** 1**);**

cout **<<** t**;**

menu**();**

Sleep**(**100**);**

**}**

pos **=** 70**;**

auxg **=** 69**;**

**for** **(**a **=** 0**;** a **<** 41**;** a**++)** **{**

gotoxy**(**auxg**,** 1**);**

cout **<<** " "**;**

aux **=** pos**;**

**for** **(**b **=** 0**;** b **<=** cont1**;** b**++)** **{**

gotoxy**(**pos**,** 1**);**

cout **<<** t**[**b**];**

pos**++;**

menu**();**

**}**

cont1**--;**

aux**++;**

pos **=** aux**;**

auxg**++;**

Sleep**(**100**);**

**}**

cont1 **=** 39**;**

letra **=** 39**;**

pos **=** 1**;**

cont **=** 39**;**

c **=** 0**;**

**}** **while** **(**c **=** 1**);**

system**(**"pause"**);**

**return** 0**;**

**}**

void reproducir**(**string cancion**)** **{**

**if** **(**cancion**.**compare**(**" "**)** **!=** 0**)** **{**

char sonido**[**20**];**

// copying the contents of the

// string to char array

strcpy**(**sonido**,** cancion**.**c\_str**());**

strcat**(**sonido**,** ".wav"**);**

wchar\_t wtext**[**20**];**

mbstowcs**(**wtext**,** sonido**,** strlen**(**sonido**)** **+** 1**);**//Plus null

LPWSTR ptr **=** wtext**;**

//Reproducir musica

PlaySound**(**ptr**,** **NULL,** SND\_SYNC**);**

**}**

**}**

void gotoxy**(**short posicionx**,** short posiciony**)** **{**

COORD coordenadaPosicion **=** **{** posicionx**,** posiciony **};**

SetConsoleCursorPosition**(**GetStdHandle**(**STD\_OUTPUT\_HANDLE**),** coordenadaPosicion**);**

**}**

void dibujarCuadro**(**int x**,** int y**)** **{**

gotoxy**(**x**,** y**++);** cout **<<** "=========================================================="**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "|| ||"**;**

gotoxy**(**x**,** y**++);** cout **<<** "=========================================================="**;**

**}**

void menu**()** **{**

//system("Color 7A");

Cifrado cifrar**;**

gotoxy**(**0**,** 0**);**

cout **<<** "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**;**

gotoxy**(**0**,** 2**);**

cout **<<** "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**;**

dibujarCuadro**(**17**,**3**);**

gotoxy**(**22**,** 5**);** cout **<<** "UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE"**;**

gotoxy**(**20**,** 7**);** cout **<<** "\*Alex Chicaiza"**;**

gotoxy**(**20**,** 8**);** cout **<<** "\*Maria Belen Ceron"**;**

gotoxy**(**20**,** 9**);** cout **<<** "NRC: 2967"**;**

gotoxy**(**20**,** 10**);** cout **<<** "Docente: Ing.Fernando Solis"**;**

dibujarCuadro**(**17**,** 14**);**

gotoxy**(**42**,** 15**);** cout **<<** "MENU"**;**

gotoxy**(**20**,** 16**);** cout **<<** " 1- ENGLISH to SPANISH"**;**

gotoxy**(**20**,** 17**);** cout **<<** " 2- ESPANOL a INGLES"**;**

gotoxy**(**20**,** 18**);** cout **<<** " 3- DESARROLLADORES"**;**

gotoxy**(**20**,** 19**);** cout **<<** " 4- GENERAR PDF"**;**

gotoxy**(**20**,** 20**);** cout **<<** " 5- ENCRIPTAR"**;**

gotoxy**(**20**,** 21**);** cout **<<** " 6- DESENCRIPTAR"**;**

gotoxy**(**20**,** 22**);** cout **<<** " 7- MOSTRAR IMAGEN EN CONSOLA"**;**

gotoxy**(**20**,** 23**);** cout **<<** " 8- AYUDA"**;**

gotoxy**(**20**,** 24**);** cout **<<** " 9- SALIR"**;**

gotoxy**(**20**,** cont**);**

//bran(cont);

cout **<<** "==>"**;**

**if** **(**\_kbhit**())** **{**

**switch** **(**\_getch**())** **{**

**case** ARRIBA**:**

**if** **(**cont **<** 17**)** **{**

cont **=** 25**;**

**}**

cont**--;**

**break;**

**case** ABAJO**:**

**if** **(**cont **>** 23**)** **{**

cont **=** 15**;**

**}**

cont**++;**

**break;**

**case** ENTER**:**

**switch** **(**cont**)** **{**

**case** 16**:**/\*1- ENGLISH to SPANISH\*/

system**(**"cls"**);**

traductorInglesEspanol**();**

system**(**"pause"**);**

**break;**

**case** 17**:**/\*2- ESPANOL a INGLES\*/

system**(**"cls"**);**

traductorEspanolIngles**();**

system**(**"pause"**);**

**break;**

**case** 18**:**/\*3- DESARROLLADORES\*/

system**(**"cls"**);**

empezarProgramas**(**"Desarrolladores.png"**);**

system**(**"pause"**);**

**break;**

**case** 19**:**/\*4- GENERAR PDF"\*/

system**(**"cls"**);**

system**(**"txt2pdf.exe solucion.txt respuestas.pdf -oao -pfs60 -pps43 -ptc0 -width3000 -height2000"**);**

system**(**"pause"**);**

**break;**

**case** 20**:**/\*5- INCRIPTAR\*/

system**(**"cls"**);**

cout **<<** "\n\n\tDOCUMENTO ENCRIPTADO\n"**;**

cifrar**.**encriptado**();**

Sleep**(**200**);**

system**(**"pause"**);**

**break;**

**case** 21**:**/\*6- DESINCRIPTAR\*/

system**(**"cls"**);**

cout **<<** "\n\n\tDOCUMENTO DESENCRIPTADO\n"**;**

cifrar**.**desencriptado**();**

Sleep**(**200**);**

system**(**"pause"**);**

**case** 22**:**/\*7- MOSTRAR IMAGEN EN CONSOLA\*/

system**(**"cls"**);**

system**(**"pause"**);**

**break;**

**case** 23**:**/\*8- AYUDA\*/

system**(**"cls"**);**

empezarProgramas**(**"Traductor.chm"**);**

system**(**"pause"**);**

**break;**

**case** 24**:**/\*9- SALIR\*/

system**(**"cls"**);**

std**::**cout **<<** "\tGracias por usar nuestra aplicacion" **<<** std**::**endl**;**

**delete** arbolEsp**;**

**delete** arbolIngles**;**

Sleep**(**1500**);**

exit**(**0**);**

**default:**

**break;**

**}**

system**(**"cls"**);**

menu**();**

gotoxy**(**3**,** 1**);**

**break;**

**}**

**}**

**}**

void leerArchivo**()** **{**

string ingles**;**

string espanol**;**

ifstream archivoI**;**

ifstream archivoE**;**

archivoI**.**open**(**"PalabrasIngles.txt"**,** ios**::**in**);**

archivoE**.**open**(**"PalabrasEspanol.txt"**,** ios**::**in**);**

**while** **(!**archivoI**.**eof**()** **&&** **!**archivoE**.**eof**())** **{**

getline**(**archivoI**,** ingles**);**

getline**(**archivoE**,** espanol**);**

arbolIngles**->**insertarNodo**(**ingles**,** espanol**,** arbolIngles**);**

arbolEsp**->**insertarNodo**(**espanol**,** ingles**,**arbolEsp**);**

**}**

archivoI**.**close**();**

archivoE**.**close**();**

**}**

void traductorInglesEspanol**()** **{**

system**(**"cls"**);**

Ingreso leer**;**

fstream enter**;**

enter**.**open**(**"solucion.txt"**,** fstream**::**app**);**

string ingles**;**

string espanol**;**

char**\*** buscar **=** **(**char**\*)(**malloc**(sizeof(**char**)));**

bool bandera **=** **true;**

cout **<<** endl **<<** endl**;**

string msg1 **=** "INGRESE LA PALABRA EN INGLES: "**;**

ingles **=** leer**.**ingresarLetras**(**msg1**);**

strcpy**(**buscar**,** ingles**.**c\_str**());**

\_strupr**(**buscar**);**

espanol **=** arbolIngles**->**buscar**(**arbolIngles**,** buscar**);**

**if** **(**espanol**.**compare**(**" "**)** **==** 0**)** **{**

espanol **=** "PALABRA NO EXISTE EN LA BASE DE DATOS"**;**

cout **<<** "\t" **<<** espanol **<<** endl**;**

bandera **=** **false;**

**}**

**if** **(!**bandera**)** **{**

opcionSonido**(**" "**,** " "**);**

**}**

**else** **{**

opcionSonido**(**buscar**,** espanol**);**

**}**

enter **<<** "\tTRADUCCION INGLES ESPANOL" **<<** endl**;**

enter **<<** "\t" **<<** buscar **<<** "\t-\t" **<<** espanol **<<** endl**<<**endl**;**

**}**

void traductorEspanolIngles**()** **{**

system**(**"cls"**);**

Ingreso leer**;**

fstream enter**;**

enter**.**open**(**"solucion.txt"**,** fstream**::**app**);**

string palabra**;**

string resultado**;**

char**\*** buscar **=** **(**char**\*)(**malloc**(sizeof(**char**)));**

bool bandera **=** **true;**

cout **<<** endl **<<** endl**;**

string msg1**=**"INGRESE LA PALABRA EN ESPANOL: "**;**

palabra**=**leer**.**ingresarLetras**(**msg1**);**

// copying the contents of the

// string to char array

strcpy**(**buscar**,** palabra**.**c\_str**());**

\_strupr**(**buscar**);**

resultado **=** arbolEsp**->**buscar**(**arbolEsp**,** buscar**);**

**if** **(**resultado**.**compare**(**" "**)** **==** 0**)** **{**

resultado **=** "PALABRA NO EXISTE EN LA BASE DE DATOS"**;**

cout **<<** "\n\t" **<<** resultado **<<** endl**;**

bandera **=** **false;**

**}**

**if** **(!**bandera**)** **{**

opcionSonido**(**" "**,** " "**);**

**}**

**else** **{**

opcionSonido**(**resultado**,** buscar**);**

**}**

enter **<<** "\tTRADUCCION ESPANOL INGLES" **<<** endl**;**

enter **<<** "\t" **<<** buscar **<<** "\t-\t" **<<** resultado **<<** endl **<<** endl**;**

**}**

void opcionSonido**(**string ingles**,** string espanol**)** **{**

int flecha **=** 6**;**

bool bandera **=** **true;**

**do** **{**

gotoxy**(**2**,** 6**);** cout **<<** " 1. Sonido espanol"**;**

gotoxy**(**2**,** 7**);** cout **<<** " 2. Sonido ingles"**;**

gotoxy**(**2**,** 8**);** cout **<<** " 3. Salir"**;**

gotoxy**(**2**,** flecha**);** cout **<<** "==>"**;**

**switch** **(**\_getch**())**

**{**

**case** ARRIBA**:**

flecha**--;**

**if** **(**flecha **<** 6**)** **{**

flecha **=** 8**;**

**}**

**break;**

**case** ABAJO**:**

flecha**++;**

**if** **(**flecha **>** 8**)** **{**

flecha **=** 6**;**

**}**

**break;**

**case** ENTER**:**

**switch** **(**flecha**)**

**{**

**case** 6**:**

reproducir**(**espanol**);**

**break;**

**case** 7**:**

reproducir**(**ingles**);**

**break;**

**case** 8**:**

bandera **=** **false;**

**break;**

**default:**

**break;**

**}**

**default:**

**break;**

**}**

**}** **while** **(**bandera**);**

gotoxy**(**0**,** 11**);**

**}**

void empezarProgramas**(**string nombre**)** **{**

char s**[]** **=** "open"**;**

wchar\_t wtext**[**5**];**

mbstowcs**(**wtext**,** s**,** strlen**(**s**)** **+** 1**);**

LPWSTR ptr **=** wtext**;**

char n**[**20**];**

strcpy**(**n**,** nombre**.**c\_str**());**

wchar\_t wtext2**[**20**];**

mbstowcs**(**wtext2**,** n**,** strlen**(**n**)** **+** 1**);**

LPWSTR ptr2 **=** wtext2**;**

ShellExecute**(NULL,** ptr**,** ptr2**,** **NULL,** **NULL,** SW\_SHOWNORMAL**);**

**}**

void solucionTxt**()** **{**

fstream enter**;**

enter**.**open**(**"solucion.txt"**,** fstream**::**out**);** //para leer in, para salir es out escribir

enter **<<** "\n\n\t\t\tUNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE\n" **<<** endl**;**

enter **<<** "\t\t\t\tINGENIERIA DE SOFTWARE" **<<** endl**<<**endl**;**

enter **<<** " Ingrantes: " **<<** endl**;**

enter **<<** "\t\t\t-Maria Belen Ceron" **<<** endl**;**

enter **<<** "\t\t\t-Alex Chicaiza" **<<** endl**;**

enter **<<** "\n\n\t\t\tSOLUCIONES " **<<** endl **<<** endl**;**

enter**.**close**();**

**}**

1. **Conclusión**

Después de las investigaciones y entendiendo como funciona cada tipo de árbol, se puedo observar que el árbol B es uno de los mejores para hacer un traductor, tiene una manera mas eficiente de ordenar los datos y también es más rápido al momento de buscar palabras, etc.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DOCENTE RESPONSABLE COORDINADOR DE ÁREA

Ing. Fernando Solis. MsC. PhD. Rodrigo Fonseca.