|  |  |  |
| --- | --- | --- |
| **CARRERA:**  Ingeniería De Software | **GUÍA**  No. 03 | **TIEMPO ESTIMADO:**  - |
| **ASIGNATURA:**  Estructura de Datos NRC: 2967 | **FECHA DE ELABORACION:** 27-01-2020  **SEMESTRE**: Septiembre 2019 – Febrero 2020 | |
| **TÍTULO:**  Proyecto Parcial 3   * Jorge Galarza * Kevin Zurita | **DOCENTE:** Ing. Fernando Solís. MsC. | |

**OBJETIVO**

* Desarrollar un programa en C++ que contenga todo el contenido aprendido en tercer parcial de la materia Estructura de Datos, aplicando árboles.

**INSTRUCCIONES**

1. Utilice como material principal, aquel indicado en clase por el docente.
2. Utilice información consultada en Internet y conocimiento adquirido en clase.

**ACTIVIDADES**

1. **Ubicación de recursos**

**p.e.**

1. Formar grupos de máximo 2 personas por computador
2. Instalar la herramienta Tiberius
3. **Planteamiento del problema**

**p.e.**

El programa consiste en realizar un diccionario de Español-Ingles e Inglés - Español que traduzca mediante árboles binarios en los dos lenguajes. Cada traducción será un nodo del árbol acompañado de un id.

**Marco Teórico**

**Árboles (Informática):** En ciencias de la computación y en informática, un árbol es un tipo abstracto de datos (TAD) ampliamente usado que imita la estructura jerárquica de un árbol, con un valor en la raíz y subárboles con un nodo padre, representado como un conjunto de nodos enlazados.

Una estructura de datos de árbol se puede definir de forma recursiva (localmente) como una colección de nodos (a partir de un nodo raíz), donde cada nodo es una estructura de datos con un valor, junto con una lista de referencias a los nodos (los hijos) , con la condición de que ninguna referencia esté duplicada ni que ningún nodo apunte a la raíz.

Alternativamente, un árbol se puede definir de manera abstracta en su conjunto como un árbol ordenado, con un valor asignado a cada nodo. Ambas perspectivas son útiles: mientras que un árbol puede ser analizado matemáticamente, realmente es representado como una estructura de datos en la que se trabaja con cada nodo por separado (en lugar de como una lista de nodos y una lista de adyacencia entre nodos, como un grafo). Mirando a un árbol como conjunto, se puede hablar de el nodo padre de un nodo dado, pero en general se habla de una estructura de datos de un nodo dado que sólo contiene la lista de sus hijos sin referencia a su padre (si lo hay).

**Árboles 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.

**Conclusión:**

* Se ha realizado un traductor aplicando árboles binarios, su búsqueda y sus operaciones, para el correcto funcionamiento del programa, mejorando el rendimiento gracias al uso de memoria dinámica.

**Código:**

//CLASE MAIN

/\* \*\*\*\*\* UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE \*\*\*\*\* \*\*

\*\* Estructura de Datos \*\*

\*\* Nombre: Jorge Galarza - Kevin Zurita \*\*

\*\* NRC: 2742 \*\*

\*\* Fecha de realizacion: 01/12/2019 \*\*

\*\* Fecha de modificacion: 12/12/2019 \*\*

\*\* Ing. Fernando Solis \*\*

\*/

#include <iostream>

#include <stdio.h>

#include <string>

#include <windows.h>

#include "DiccionarioG.h"

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

void cifrado**();**

void decifrado**();**

int main**(){**

DiccionarioG graf**;**

graf**.**diccionario**();**

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

cifrado**();**

decifrado**();**

**}**

void cifrado**()**

**{**

char arch\_in**[]=**"diccionario.txt"**;**

char arch\_out**[]=**"diccionariocifrado.txt"**;**

char c**;**

FILE **\***ofp**,\***ifp**;**

ifp**=**fopen**(**arch\_in**,**"r"**);**

ofp**=**fopen**(**arch\_out**,**"w"**);**

// system("txt2pdf.exe Datos.txt Datospdf.pdf -oao -pfs60 -pps43 -ptc0 -width3000 -height2000");

**while((**c**=** fgetc**(**ifp**))!=**EOF**)**

**{**

fputc**(**c**+**5**-**3**+**7**-**9**+**4**+**2**-**3**,**ofp**);**

**}**

fclose**(**ofp**);**

fclose**(**ifp**);**

**}**

void decifrado**()**

**{**

char arch\_in**[]=**"diccionariodecifrado.txt"**;**

char arch\_out**[]=**"diccionariocifrado.txt"**;**

char c**;**

FILE **\***ofp**,\***ifp**;**

ofp**=**fopen**(**arch\_in**,**"w"**);**

ifp**=**fopen**(**arch\_out**,**"r"**);**

**while((**c**=** fgetc**(**ifp**))!=**EOF**)**

**{**

fputc**(**c**-**5**+**3**-**7**+**9**-**4**-**2**+**3**,**ofp**);**

**}**

fclose**(**ofp**);**

fclose**(**ifp**);**

**}**

//CLASE MENU

/\* \*\*\*\*\* UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE \*\*\*\*\* \*\*

\*\* Estructura de Datos \*\*

\*\* Nombre: Jorge Galarza - Kevin Zurita \*\*

\*\* NRC: 2742 \*\*

\*\* Fecha de realizacion: 01/12/2019 \*\*

\*\* Fecha de modificacion: 12/12/2019 \*\*

\*\* Ing. Fernando Solis \*\*

\*/

#include <iostream>

#include <winbgim.h>

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <windows.h>

#include <fstream>

#define TECLA\_DERECHA 77

#define TECLA\_IZQUIERDA 75

#define ENTER 13

#define F1 59

void**\*** selloEspe**(**void **\*);**

static HWND hConWnd**;**

HWND BCX\_Bitmap**(**char**\*,** HWND **=** 0**,** int **=** 0**,** int **=** 0**,** int **=** 0**,** int **=** 0**,** int **=** 0**,** int **=** 0**,** int **=** 0**,** int **=** 0**);**

HWND GetConsoleWndHandle**(**void**);**

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

// Funcion gotoxy

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

HANDLE hcon**;**

hcon **=** GetStdHandle**(**STD\_OUTPUT\_HANDLE**);**

COORD dwPos**;**

dwPos**.**X **=** x**;**

dwPos**.**Y**=** y**;**

SetConsoleCursorPosition**(**hcon**,**dwPos**);**

**}**

// Marquesina

void**\*** marquesina**(**void **\***data**){**

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

char t**[**50**]** **=**"Diccionario: Proyecto Estructura de Datos"**,**auxt**[**39**]=**" "**;**

**do{**

**for** **(**a**=**0**;**a**<**42**;**a**++){**

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

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

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

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

pos**--;**

**}**

aux**++;**

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

cont**--;**

Sleep **(**75**);**

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

**break;**

**}**

**}**

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

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

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

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

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

Sleep **(**75**);**

**}**

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**++;**

**}**

cont1**--;**

aux**++;**

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

auxg**++;**

Sleep **(**75**);**

**}**

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

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

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

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

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

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

**}**

// Cambia el color de la letras y dondo de la consola

void SetConsoleColour**(**WORD**\*** Attributes**,** DWORD Colour**){**

CONSOLE\_SCREEN\_BUFFER\_INFO Info**;**

HANDLE hStdout **=** GetStdHandle**(**STD\_OUTPUT\_HANDLE**);**

GetConsoleScreenBufferInfo**(**hStdout**,** **&**Info**);**

**\***Attributes **=** Info**.**wAttributes**;**

SetConsoleTextAttribute**(**hStdout**,** Colour**);**

**}**

//

void ResetConsoleColour**(**WORD Attributes**){**

SetConsoleTextAttribute**(**GetStdHandle**(**STD\_OUTPUT\_HANDLE**),** Attributes**);**

**}**

// Remarca el metodo uno en la consola

void uno **(**WORD Attributes**){**

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

SetConsoleColour**(&**Attributes**,** FOREGROUND\_BLUE **|** BACKGROUND\_INTENSITY **|** BACKGROUND\_GREEN**);**

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

ResetConsoleColour**(**Attributes**);**

cout**<<**"\t CodigoQR \t Codigo de Barras \t Sello Pixeleado \t Salir "**<<**endl**;**

**}**

// Remarca el metodo dos en la consola

void dos**(**WORD Attributes**){**

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

cout**<<**"\tDiccionario \t "**;**

SetConsoleColour**(&**Attributes**,** FOREGROUND\_BLUE **|** BACKGROUND\_INTENSITY **|** BACKGROUND\_GREEN**);**

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

ResetConsoleColour**(**Attributes**);**

cout**<<**"\t Codigo de Barras \t Sello Pixeleado \t Salir"**<<**endl**;**

**}**

// Remarca el metodo tres en la consola

void tres**(**WORD Attributes**){**

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

cout**<<**"\tDiccionario \t CodigoQR \t "**;**

SetConsoleColour**(&**Attributes**,** FOREGROUND\_BLUE **|** BACKGROUND\_INTENSITY **|** BACKGROUND\_GREEN**);**

cout**<<**"Codigo de Barras"**;**

ResetConsoleColour**(**Attributes**);**

cout**<<**"\t Sello Pixeleado \t Salir"**;**

**}**

// Remarca el metodo cuatro en la consola

void cuatro**(**WORD Attributes**){**

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

cout**<<**"\tDiccionario \t CodigoQR \t Codigo de Barras \t "**;**

SetConsoleColour**(&**Attributes**,** FOREGROUND\_BLUE **|** BACKGROUND\_INTENSITY **|** BACKGROUND\_GREEN**);**

cout**<<**"Sello Pixeleado"**;**

ResetConsoleColour**(**Attributes**);**

cout **<<** "\t Salir"**;**

**}**

// Remarca el metodo cinco en la consola

void cinco**(**WORD Attributes**){**

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

cout**<<**"\tDiccionario \t CodigoQR \t Codigo de Barras \t Sello Pixeleado \t"**;**

SetConsoleColour**(&**Attributes**,** FOREGROUND\_BLUE **|** BACKGROUND\_INTENSITY **|** BACKGROUND\_GREEN**);**

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

ResetConsoleColour**(**Attributes**);**

**}**

// funcion principal

int main**()**

**{**

system**(**"color F1"**);**

// system("color e");

// declaracoion de variables

WORD Attributes**=**0**;**

pthread\_t thread1**;**

pthread\_create**(&**thread1**,** **NULL,** marquesina**,** **NULL);**

int met**,**tecla**,**opc**=**0**;**

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

cout**<<**"\t Seleccione el metodo que desea usar \t\t\t\t\t\t Pulse F1 para AYUDA"**<<**endl**;**

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

cout**<<**"\tDiccionario \t CodigoQR \t Codigo de Barras \t Sello Pixeleado \t Salir "**<<**endl**;**

**do{**

// lee la tecla que se pulsa

**do{**

tecla**=**getch**();**

**}while(**tecla**!=**TECLA\_DERECHA **&&** tecla**!=**TECLA\_IZQUIERDA **&&** tecla**!=**ENTER **&&** tecla**!=**F1**);**

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

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

cout**<<**"\t Seleccione el metodo que desea usar \t\t\t\t pulse F1 para ayuda"**<<**endl**;;**

// Dependiendo de la tecla que se pulse se usa un contador o abre la ventana de ayuda

**switch** **(**tecla**){**

cout **<<**"\n\n"**<<**opc**<<**endl**;**

**case** TECLA\_DERECHA**:**

opc**++;**

**if(**opc**==**6**){**

opc**=**1**;**

**}**

**break;**

**case** TECLA\_IZQUIERDA**:**

opc**--;**

**if(**opc**<=**0**){**

opc**=**5**;**

**}**

**break;**

**case** ENTER**:**

met**=**opc**;**

**break;**

// en caso de presionar la tecla f1 se abre una ventana de ayuda

**case** F1**:**

system**(**"AyudaDiccionario.chm"**);**

**break;**

**}**

// Dependiendo del contador entra a un caso en el que pinta el metodo que se seleccionara

**switch(**opc**){**

**case** 1**:**

uno**(**Attributes**);**

**break;**

**case** 2**:**

dos**(**Attributes**);**

**break;**

**case** 3**:**

tres**(**Attributes**);**

**break;**

**case** 4**:**

cuatro**(**Attributes**);**

**break;**

**case** 5**:**

cinco**(**Attributes**);**

**break;**

**}**

// Cuando se pulsa enter entra al metodo que se encuentre seleccionado

**switch(**met**){**

// ejecuta jugar

**case** 1**:**

SuspendThread**(&**thread1**);**

system**(**"Diccionario.exe"**);**

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

opc**=**0**;**

met**=**0**;**

**break;**

// Ejecuta código QR

**case** 2**:**

system**(**"CodigoQR.png"**);**

opc**=**0**;**

met**=**0**;**

**break;**

// Ejecuta codigo de barras

**case** 3**:**

system**(**"CodigodeBarras.png"**);**

opc**=**0**;**

met**=**0**;**

**break;**

// Ejecuta el sello pixeleado

**case** 4**:**

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

void **\***status**;**

pthread\_t thread2**;**

pthread\_create **(&**thread2 **,** **NULL** **,** selloEspe **,** **(**void **\*)** **&** thread1**);**

pthread\_join**(**thread2**,** **&**status**);**

opc**=**0**;**

met**=**0**;**

**break;**

**}**

**}while(**met**!=**5**);**

// Termina el programa

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

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

cout**<<**"Gracias por usar nuestro diccionario !!!"**<<**endl**;**

system**(**"PAUSE"**);**

**return** 0**;**

**}**

HWND GetConsoleWndHandle**(**void**)**

**{**

HWND hConWnd**;**

OSVERSIONINFO os**;**

char szTempTitle**[**64**],** szClassName**[**128**],** szOriginalTitle**[**1024**];**

os**.**dwOSVersionInfoSize **=** **sizeof(**OSVERSIONINFO**);**

GetVersionEx**(&**os**);**

// may not work on WIN9x

**if** **(**os**.**dwPlatformId **==** VER\_PLATFORM\_WIN32s**)** **return** 0**;**

GetConsoleTitle**(**szOriginalTitle**,** **sizeof(**szOriginalTitle**));**

sprintf**(**szTempTitle**,** "%u - %u"**,** GetTickCount**(),** GetCurrentProcessId**());**

SetConsoleTitle**(**szTempTitle**);**

Sleep**(**60**);**

// handle for NT and XP

hConWnd **=** FindWindow**(NULL,** szTempTitle**);**

SetConsoleTitle**(**szOriginalTitle**);**

// may not work on WIN9x

**if** **(**os**.**dwPlatformId **==** VER\_PLATFORM\_WIN32\_WINDOWS**)**

**{**

hConWnd **=** GetWindow**(**hConWnd**,** GW\_CHILD**);**

**if** **(**hConWnd **==** **NULL)** **return** 0**;**

GetClassName**(**hConWnd**,** szClassName**,** **sizeof(**szClassName**));**

// while ( \_stricmp( szClassName, "ttyGrab" ) != 0 )

**while** **(**strcmp**(**szClassName**,** "ttyGrab"**)** **!=** 0**)**

**{**

hConWnd **=** GetNextWindow**(**hConWnd**,** GW\_HWNDNEXT**);**

**if** **(**hConWnd **==** **NULL)** **return** 0**;**

GetClassName**(**hConWnd**,** szClassName**,** **sizeof(**szClassName**));**

**}**

**}**

**return** hConWnd**;**

**}**

HWND BCX\_Bitmap**(**char**\*** Text**,** HWND hWnd**,** int id**,** int X**,** int Y**,** int W**,** int H**,** int Res**,** int Style**,** int Exstyle**)**

**{**

HWND A**;**

HBITMAP hBitmap**;**

// set default style

**if** **(!**Style**)** Style **=** WS\_CLIPSIBLINGS **|** WS\_CHILD **|** WS\_VISIBLE **|** SS\_BITMAP **|** WS\_TABSTOP**;**

// form for the image

A **=** CreateWindowEx**(**Exstyle**,** "static"**,** **NULL,** Style**,** X**,** Y**,** 0**,** 0**,** hWnd**,** **(**HMENU**)**id**,** GetModuleHandle**(**0**),** **NULL);**

// Text contains filename

hBitmap **=** **(**HBITMAP**)**LoadImage**(**0**,** Text**,** IMAGE\_BITMAP**,** 0**,** 0**,** LR\_LOADFROMFILE **|** LR\_CREATEDIBSECTION**);**

// auto-adjust width and height

**if** **(**W **||** H**)** hBitmap **=** **(**HBITMAP**)**CopyImage**(**hBitmap**,** IMAGE\_BITMAP**,** W**,** H**,** LR\_COPYRETURNORG**);**

SendMessage**(**A**,** **(**UINT**)**STM\_SETIMAGE**,** **(**WPARAM**)**IMAGE\_BITMAP**,** **(**LPARAM**)**hBitmap**);**

**if** **(**W **||** H**)** SetWindowPos**(**A**,** HWND\_TOP**,** X**,** Y**,** W**,** H**,** SWP\_DRAWFRAME**);**

**return** A**;**

**}**

void**\*** selloEspe**(**void **\***arg**)**

**{**

pthread\_mutex\_t ptmutex1**;**

pthread\_mutex\_lock**(&**ptmutex1**);**

hConWnd **=** GetConsoleWndHandle**();**

BCX\_Bitmap**(**"logo.bmp"**,** hConWnd**,** 123**,** 150**,** 150**,** 0**,** 0**);**

Sleep**(**5000**);**

pthread\_mutex\_unlock**(&**ptmutex1**);**

**}**

//CLASE ARBOL BINARIO

/\* \*\*\*\*\* UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE \*\*\*\*\* \*\*

\*\* Estructura de Datos \*\*

\*\* Nombre: Jorge Galarza - Kevin Zurita \*\*

\*\* NRC: 2742 \*\*

\*\* Fecha de realizacion: 01/12/2019 \*\*

\*\* Fecha de modificacion: 12/12/2019 \*\*

\*\* Ing. Fernando Solis \*\*

\*/

#include <stdio.h>

#include <iostream>

#include <string>

#include <fstream>

#include <stdlib.h>

#include "NodoArbol.h"

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

**typedef** NodoArbol**\*** pnodo**;**

class ArbolBinario**{**

public**:**

void voz**(**string**);**

void llenarDiccionario**(**pnodo**&,** pnodo**&,** pnodo**&);**

void crearNodoArbol**(**int**,** string**,** pnodo**&);**

void insertarNodoArbol**(**pnodo**&,** int**,** string**);**

int CodigoPalabra**(**pnodo**,** string**);**

string PalabraCodigo**(**pnodo**,** int**);**

string Traduccion**(**pnodo**,** pnodo**,** string**);**

**};**

void ArbolBinario**::**crearNodoArbol**(**int id**,** string palabra**,** pnodo**&** tree**)**

**{**

pnodo nodoArbol **=** **new** NodoArbol**(**id**,** palabra**);**

tree **=** nodoArbol**;**

**}**

void ArbolBinario**::**insertarNodoArbol**(**pnodo**&** tree**,** int id**,** string palabra**){**

**if** **(**tree **==** **NULL)**

**{**

crearNodoArbol**(**id**,** palabra**,** tree**);**

**}**

**else** **if** **(**id **<** tree**->**getId**()){**

insertarNodoArbol**(**tree**->**izquierda**,** id**,** palabra**);**

**}**

**else** **if** **(**id **>=** tree**->**getId**()){**

insertarNodoArbol**(**tree**->**derecha**,** id**,** palabra**);**

**}**

**}**

string ArbolBinario**::**PalabraCodigo**(**pnodo tree**,** int code**)**

**{**

**while** **(**tree **!=** **NULL)**

**{**

**if** **(**tree**->**getId**()** **==** code**)**

**{**

**return** tree**->**getPalabra**();**

**}**

tree **=** tree**->**derecha**;**

**}**

**return** "No disponible."**;**

**}**

int ArbolBinario**::**CodigoPalabra**(**pnodo tree**,** string palabra**)**

**{**

**while** **(**tree **!=** **NULL)**

**{**

**if** **(**tree**->**getPalabra**()** **==** palabra**)**

**{**

**return** tree**->**getId**();**

**}**

tree **=** tree**->**derecha**;**

**}**

**return** 0**;**

**}**

string ArbolBinario**::**Traduccion**(**pnodo ArbolIngles**,** pnodo ArbolEspaniol**,**string word**)**

**{**

**return** PalabraCodigo**(**ArbolEspaniol**,** CodigoPalabra**(**ArbolIngles**,** word**));**

**}**

void ArbolBinario**::**llenarDiccionario**(**pnodo**&** ArbolIngles**,** pnodo**&** ArbolEspaniol**,** pnodo**&** ArbolTraduccion**){**

string line**;**

fstream out**;**

out**.**open**(**"diccionario.txt"**,**fstream**::**in**);**

ifstream myfile **(**"diccionario.txt"**);**

**if** **(**myfile**.**is\_open**())**

**{**

**while** **(** getline **(**myfile**,**line**)** **)**

**{**

int id **=** atoi**(**line**.**substr**(**0**,**line**.**find**(**"/"**)).**c\_str**());**

string words**=**line**.**substr**(**line**.**find**(**"/"**)+**1**,**line**.**length**());**

insertarNodoArbol**(**ArbolIngles**,**id**,**words**.**substr**(**0**,**words**.**find**(**"/"**)));**

words**=**words**.**substr**(**words**.**find**(**"/"**)+**1**,**words**.**length**());**

insertarNodoArbol**(**ArbolEspaniol**,**id**,**words**.**substr**(**0**,**words**.**find**(**"/"**)));**

insertarNodoArbol**(**ArbolTraduccion**,**id**,**words**.**substr**(**words**.**find**(**"/"**)+**1**,**words**.**length**()));**

**}**

myfile**.**close**();**

out**.**close**();**

**}**

**}**

void ArbolBinario**::**voz**(**string cad**){**

std**::**ofstream audio**(**"reproducir.vbs"**);**

audio**<<**"Dim SAPI"**<<**endl**;**

audio**<<**"set SAPI = CreateObject (\"sapi.spvoice\")"**<<**endl**;**

audio**<<**"SAPI.Speak"**<<**"\""**<<**cad**<<**"\""**<<**endl**;**

audio**.**close**();**

system**(**"reproducir.vbs"**);**

**}**

//CLASE DICCIONARIOG

/\* \*\*\*\*\* UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE \*\*\*\*\* \*\*

\*\* Estructura de Datos \*\*

\*\* Nombre: Jorge Galarza - Kevin Zurita \*\*

\*\* NRC: 2742 \*\*

\*\* Fecha de realizacion: 01/12/2019 \*\*

\*\* Fecha de modificacion: 12/12/2019 \*\*

\*\* Ing. Fernando Solis \*\*

\*/

#include <winbgim.h>

#include <iostream>

#include <stdio.h>

#include <string>

#include <windows.h>

#include "ArbolBinario.h"

#include "Ingreso.h"

class DiccionarioG**{**

public**:**

void lineas**();**

void defaulto**();**

void diccionario**();**

string minuscula**(**string**);**

**};**

string DiccionarioG**::**minuscula**(**string palabra**){**

string aux**;**

**for** **(**int i**=**0**;** i**<**palabra**.**length**();** i**++){**

aux**=**tolower**(**palabra**.**at**(**i**));**

palabra**.**replace**(**i**,**1**,**aux**.**c\_str**());**

**}**

**return** palabra**;**

**}**

void DiccionarioG**::**lineas**(){**

setcolor**(**0**);**

line**(**45**,**60**,**280**,**60**);**

line**(**45**,**100**,**280**,**100**);**

line**(**45**,**140**,**280**,**140**);**

line**(**45**,**180**,**280**,**180**);**

line**(**45**,**220**,**280**,**220**);**

line**(**45**,**260**,**280**,**260**);**

line**(**45**,**300**,**280**,**300**);**

line**(**45**,**340**,**280**,**340**);**

line**(**45**,**380**,**280**,**380**);**

line**(**45**,**420**,**280**,**420**);**

line**(**45**,**460**,**280**,**460**);**

line**(**45**,**500**,**280**,**500**);**

line**(**45**,**540**,**280**,**540**);**

line**(**310**,**60**,**545**,**60**);**

line**(**310**,**100**,**545**,**100**);**

line**(**310**,**140**,**545**,**140**);**

line**(**310**,**180**,**545**,**180**);**

line**(**310**,**220**,**545**,**220**);**

line**(**310**,**260**,**545**,**260**);**

line**(**310**,**300**,**545**,**300**);**

line**(**310**,**340**,**545**,**340**);**

line**(**310**,**380**,**545**,**380**);**

line**(**310**,**420**,**545**,**420**);**

line**(**310**,**460**,**545**,**460**);**

line**(**310**,**500**,**545**,**500**);**

line**(**310**,**540**,**545**,**540**);**

**}**

void DiccionarioG**::**defaulto**(){**

setcolor**(**8**);**

rectangle**(**15**,**15**,**575**,**575**);**

rectangle**(**30**,**30**,**560**,**560**);**

line**(**295**,**30**,**295**,**560**);**

setfillstyle**(**1**,**6**);**

floodfill**(**20**,**20**,**8**);**

setfillstyle**(**1**,**15**);**

floodfill**(**35**,**35**,**8**);**

floodfill**(**305**,**550**,**8**);**

**}**

void DiccionarioG**::**diccionario**(){**

NodoArbol**\*** aIngles**=NULL;**

NodoArbol**\*** aEspaniol**=NULL;**

NodoArbol**\*** aPronunciacion**=NULL;**

ArbolBinario tree**;**

tree**.**llenarDiccionario**(**aIngles**,**aEspaniol**,**aPronunciacion**);**

initwindow**(**600**,**600**);**

defaulto**();**

lineas**();**

setcolor**(**15**);**

settextstyle**(**8**,** 0**,** 4**);**

outtextxy**(**65**,**295**,**"Dictionary"**);**

outtextxy**(**95**,**260**,**"English"**);**

outtextxy**(**330**,**260**,**"Diccionario"**);**

outtextxy**(**355**,**295**,**"Espaniol"**);**

char tecla**;**

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

**do{**

tecla**=**getch**();**

**switch(**tecla**){**

**case** 75**:**

**if(**i**!=**1**){**

i**=**1**;**

setfillstyle**(**1**,**15**);**

floodfill**(**425**,**170**,**8**);**

setcolor**(**15**);**

outtextxy**(**330**,**260**,**"Diccionario"**);**

outtextxy**(**355**,**295**,**"Espaniol"**);**

lineas**();**

setcolor**(**1**);**

setfillstyle**(**1**,**1**);**

rectangle**(**150**,**150**,**180**,**200**);**

floodfill**(**165**,**170**,**1**);**

line**(**140**,**200**,**166**,**230**);**

line**(**190**,**200**,**164**,**230**);**

line**(**140**,**200**,**190**,**200**);**

floodfill**(**165**,**210**,**1**);**

**}**

**break;**

**case** 77**:**

**if(**i**!=**2**){**

i**=**2**;**

setcolor**(**15**);**

setfillstyle**(**1**,**15**);**

floodfill**(**165**,**170**,**8**);**

outtextxy**(**65**,**295**,**"Dictionary"**);**

outtextxy**(**95**,**260**,**"English"**);**

lineas**();**

setcolor**(**1**);**

setfillstyle**(**1**,**1**);**

rectangle**(**410**,**150**,**440**,**200**);**

floodfill**(**425**,**170**,**1**);**

line**(**400**,**200**,**426**,**230**);**

line**(**450**,**200**,**424**,**230**);**

line**(**400**,**200**,**450**,**200**);**

floodfill**(**425**,**210**,**1**);**

**}**

**break;**

**case** 13**:**

**if(**i**==**1**||**i**==**2**){**

Ingreso ingresar**;**

cleardevice**();**

defaulto**();**

lineas**();**

string traducir**;**

setcolor**(**15**);**

settextstyle**(**8**,** 0**,** 3**);**

**if(**i**==**1**){**

outtextxy**(**105**,**60**,**"Enter the word to translate"**);**

getch**();**

traducir**=**ingresar**.**ingresarString**(**"Enter the word to translate"**);**

cleardevice**();**

defaulto**();**

lineas**();**

setcolor**(**15**);**

outtextxy**(**50**,**60**,**"Original word"**);**

outtextxy**(**315**,**60**,**"Traduced word"**);**

outtextxy**(**50**,**110**,**traducir**.**c\_str**());**

outtextxy**(**315**,**110**,**tree**.**Traduccion**(**aIngles**,**aEspaniol**,**minuscula**(**traducir**)).**c\_str**());**

Sleep**(**3000**);**

**}else** **if(**i**==**2**){**

outtextxy**(**90**,**60**,**"Ingrese la palabra a traducir"**);**

getch**();**

traducir**=**ingresar**.**ingresarString**(**"Ingrese la palabra a traducir"**);**

cleardevice**();**

defaulto**();**

lineas**();**

setcolor**(**15**);**

outtextxy**(**50**,**60**,**"Palabra Original"**);**

outtextxy**(**315**,**60**,**"Palabra Traducida"**);**

outtextxy**(**50**,**110**,**traducir**.**c\_str**());**

outtextxy**(**315**,**110**,**tree**.**Traduccion**(**aEspaniol**,**aIngles**,**minuscula**(**traducir**)).**c\_str**());**

outtextxy**(**315**,**160**,**tree**.**Traduccion**(**aEspaniol**,**aPronunciacion**,**minuscula**(**traducir**)).**c\_str**());**

tree**.**voz**(**tree**.**Traduccion**(**aEspaniol**,**aIngles**,**minuscula**(**traducir**)));**

Sleep**(**3000**);**

**}**

cleardevice**();**

defaulto**();**

lineas**();**

setcolor**(**15**);**

settextstyle**(**8**,** 0**,** 4**);**

outtextxy**(**65**,**295**,**"Dictionary"**);**

outtextxy**(**95**,**260**,**"English"**);**

outtextxy**(**330**,**260**,**"Diccionario"**);**

outtextxy**(**355**,**295**,**"Espaniol"**);**

**}**

**}**

**}while(**tecla**!=**27**);**

**}**

// CLASE EXCLUIR

#include <iostream>

#include <string>

#include <bits/stdc++.h>

#include <stdlib.h>

//Universidad de las Fuerzas Armdas

//Nombre: Kevin Zurita

//NRC: 2967

//Fecha de creacion: 23/09/2019

//FecHa de ultima modificacion: 24/09/2019

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

class Excluir**{**

public**:**

bool control**(**string s**){**

int d**=**0**;**

**for(**int i**=**0**;**i**<**s**.**size**();**i**++)**

**{**

**if(!**isdigit**(**s**[**i**])){**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

bool controlS**(**string s**){**

int d**=**0**;**

**for(**int i**=**0**;**i**<**s**.**size**();**i**++)**

**{**

**if(**isdigit**(**s**[**i**])){**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

bool controlSN**(**string s**){**

int d**=**0**;**

**for(**int i**=**0**;**i**<**s**.**size**();**i**++)**

**{**

**if(!**isdigit**(**s**[**i**])){**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

bool controlCedula**(**string cedula**){**

**if** **(**cedula**.**length**()!=**10**)** **{**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**for** **(**int i **=** 0**;** cedula**.**length**()** **>** i**;** i**++)** **{**

**if** **(!**isdigit**(**cedula**[**i**]))** **{**

cout**<<**"Error"**<<**endl**;**

**return** **false;**

**}**

**}**

**return** **true;**

**}**

**};**

//CLASE INGRESO

#include <iostream>

#include "Excluir.h"

#include <string>

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

//Universidad de las Fuerzas Armdas

//Nombre: Kevin Zurita

//NRC: 2967

//Docente: Ing. Edgar Solis

//Fecha de creacion: 23/09/2019

//FecHa de ultima modificacion: 1/10/2019

class Ingreso**{**

public**:**

string ingresarString**(**char**\*** **);**

int ingresar**(**char**\*** **);**

string ingresarStringN**(**char**\*** **);**

string ingresarCedula**(**char **\*);**

bool validadCedula**(**string**);**

**};**

string Ingreso**::**ingresarString**(**char **\***msg**){**

string valor**;**

bool flag**;**

Excluir comprobante**;**

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

**do{**

cin**>>**valor**;**

flag**=**comprobante**.**controlS**(**valor**);**

**}while(**flag**==false);**

**return** valor**;**

**}**

string Ingreso**::**ingresarStringN**(**char **\***msg**){**

string valor**;**

bool flag**;**

Excluir comprobante**;**

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

**do{**

cin**>>**valor**;**

flag**=**comprobante**.**controlSN**(**valor**);**

**}while(**flag**==false);**

**return** valor**;**

**}**

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

string valor**;**

bool flag**;**

Excluir comprobante**;**

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

**do{**

cin**>>**valor**;**

flag**=**comprobante**.**control**(**valor**);**

**}while(**flag**==false);**

int num**;**

num **=** strtof**((**valor**).**c\_str**(),**0**);**

**return** num**;**

**}**

string Ingreso**::**ingresarCedula**(**char **\***msg**){**

string valor**;**

bool flag**;**

Excluir comprobante**;**

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

**do{**

cin**>>**valor**;**

flag**=**comprobante**.**controlCedula**(**valor**);**

**}while(**flag**==false);**

**return** valor**;**

**}**

bool Ingreso**::**validadCedula**(**string cedula**){**

int mul1**=**0**,**mul2**=**0**,**res**,**cont**=**1**,**sum1**=**0**,**sum2**=**0**,**sum3**=**0**,**ult\_n1**,**ult\_n2**;**

long int num**,**coc1**,**coc2**;**

num**=**strtoul**(**cedula**.**c\_str**(),NULL,**10**);**

mul1**=**0**;** mul2**=**0**;** sum1**=**0**;** sum2**=**0**;** sum3**=**0**;** cont**=**1**;**

**while** **(**num**<**0**)**

**{**

**return** **false;**

**}**

ult\_n1**=**num**%**10**;**

coc1**=**num**/**10**;**

num**=**coc1**;**

**do{**

**do{**

coc2**=**num**/**10**;**

res**=**num**%**10**;**

**if** **(**cont**%**2**==**0**)**

**{**

mul2**=**res**\***1**;**

**if** **(**mul2**>=**10**)**

mul2**-=**9**;**

sum2**+=**mul2**;**

**}**

**else**

**{**

mul1**=**res**\***2**;**

**if** **(**mul1**>=**10**)**

mul1**-=**9**;**

sum1**+=**mul1**;**

**}**

cont**++;**

num**=**coc2**;**

**}while** **(**coc2 **!=** 0**);**

sum3**=**sum1**+**sum2**;**

**if** **((**sum3**%**10**)!=**0**)**

**{**

ult\_n2**=**sum3**/**10**;**

ult\_n2**++;**

ult\_n2**\*=**10**;**

ult\_n2**-=**sum3**;**

**}**

**else**

ult\_n2**-=**ult\_n2**;**

**if(**ult\_n2**==**ult\_n1**)**

**while** **(**ult\_n2**==**ult\_n1**)**

**{**

**return** **true;**

**}**

**}while** **(**ult\_n2**==**ult\_n1**);**

**return** **false;**

**}**

//CLASE NODOARBOL

/\* \*\*\*\*\* UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE \*\*\*\*\* \*\*

\*\* Estructura de Datos \*\*

\*\* Nombre: Jorge Galarza - Kevin Zurita \*\*

\*\* NRC: 2742 \*\*

\*\* Fecha de realizacion: 01/12/2019 \*\*

\*\* Fecha de modificacion: 12/12/2019 \*\*

\*\* Ing. Fernando Solis \*\*

\*/

#include <stdio.h>

#include <iostream>

#include <string>

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

class NodoArbol**{**

private**:**

int id**;**

string palabra**;**

NodoArbol**\*** izquierda**;**

NodoArbol**\*** derecha**;**

public**:**

NodoArbol**(**int**,** string**);**

int getId**();**

string getPalabra**();**

NodoArbol**\*** getIzquierda**();**

NodoArbol**\*** getDerecha**();**

void setId**(**int**);**

void setPalabra**(**string**);**

void setIzquierda**(**NodoArbol**\*);**

void setDerecha**(**NodoArbol**\*);**

friend class ArbolBinario**;**

**};**

NodoArbol**::**NodoArbol**(**int code**,** string word**){**

id**=**code**;**

palabra**=**word**;**

izquierda**=NULL;**

derecha**=NULL;**

**}**

int NodoArbol**::**getId**(){**

**return** id**;**

**}**

string NodoArbol**::**getPalabra**(){**

**return** palabra**;**

**}**

NodoArbol**\*** NodoArbol**::**getIzquierda**(){**

**return** izquierda**;**

**}**

NodoArbol**\*** NodoArbol**::**getDerecha**(){**

**return** derecha**;**

**}**

void NodoArbol**::**setId**(**int code**){**

id**=**code**;**

**}**

void NodoArbol**::**setPalabra**(**string word**){**

palabra**=**word**;**

**}**

void NodoArbol**::**setIzquierda**(**NodoArbol**\*** left**){**

izquierda**=**left**;**

**}**

void NodoArbol**::**setDerecha**(**NodoArbol**\*** right**){**

derecha**=**right**;**

**}**