Diccionario Electrónico de Sinónimos

Díaz Rodríguez Felipe - Novoa Neisa Lorena

[andres.diazrodiguez@correo.usa.edu.co](mailto:andres.diazrodiguez@correo.usa.edu.co) – [lorena.novoa2@correo.usa.edu.co](mailto:lorena.novoa2@correo.usa.edu.co)

# resumen

El siguiente es un informe el cual contiene el procedimiento, resultados y conclusiones de un diccionario electrónico de sinónimos, que tiene la capacidad de agrega, buscar y listar palabra. Este diccionario se realizará por medio del programa Qt en lenguaje C y usando las librerías “stdio.h” y “string.h”

***Abstract:*** *The following is a report which contains the procedure, results and conclusions of an electronic dictionary of synonyms, which has the ability to add, search and list word. This dictionary will be done through the Qt program in C language and using the “stdio.h” and “string.h” libraries*

1. PALABRAS CLAVE

Memoria RAM, estructuras en C, librería “string.h”, librería “stdio.h”

# introducción

Para el correcto funcionamiento del programa planteado por la guía de laboratorio es necesario tener conocimientos de como al crear una variable esta ocupa una dirección y espacio en la memoria RAM. Las ventajas y funciones que tiene esta memoria en un pc, son la rápida lectura y escritura de los datos en ejecución por el O.S.

Como se encontró en la documentación, el uso la memoria RAM para evaluación de códigos en criptografía es de vital importancia ya que su escritura y lectura es mucho más rápida en comparación a los HDD.

*Áreas de memoria:*

|  |
| --- |
| Estática: en esta sección se almacenan variables globales, que no cambian y que son usadas durante la ejecución del programa[1] |
| Heap: permite almacenar variables adquiridas dinámicamente por medio de las funciones malloc y calloc. [1] |
| Stack: almacena argumentos y variables locales, durante la ejecución de las funciones en las que están definidas[1] |

*Estructuras:* Es un conjunto de variables que se referencian bajo el mismo nombre. La sintaxis es la siguiente:

struct nombre\_estructura {

unsigned tipo nombre\_variable;

unsigned tipo nombre\_variable;

…

unsigned tipo nombre\_variable;

}variable\_estructura;[2]

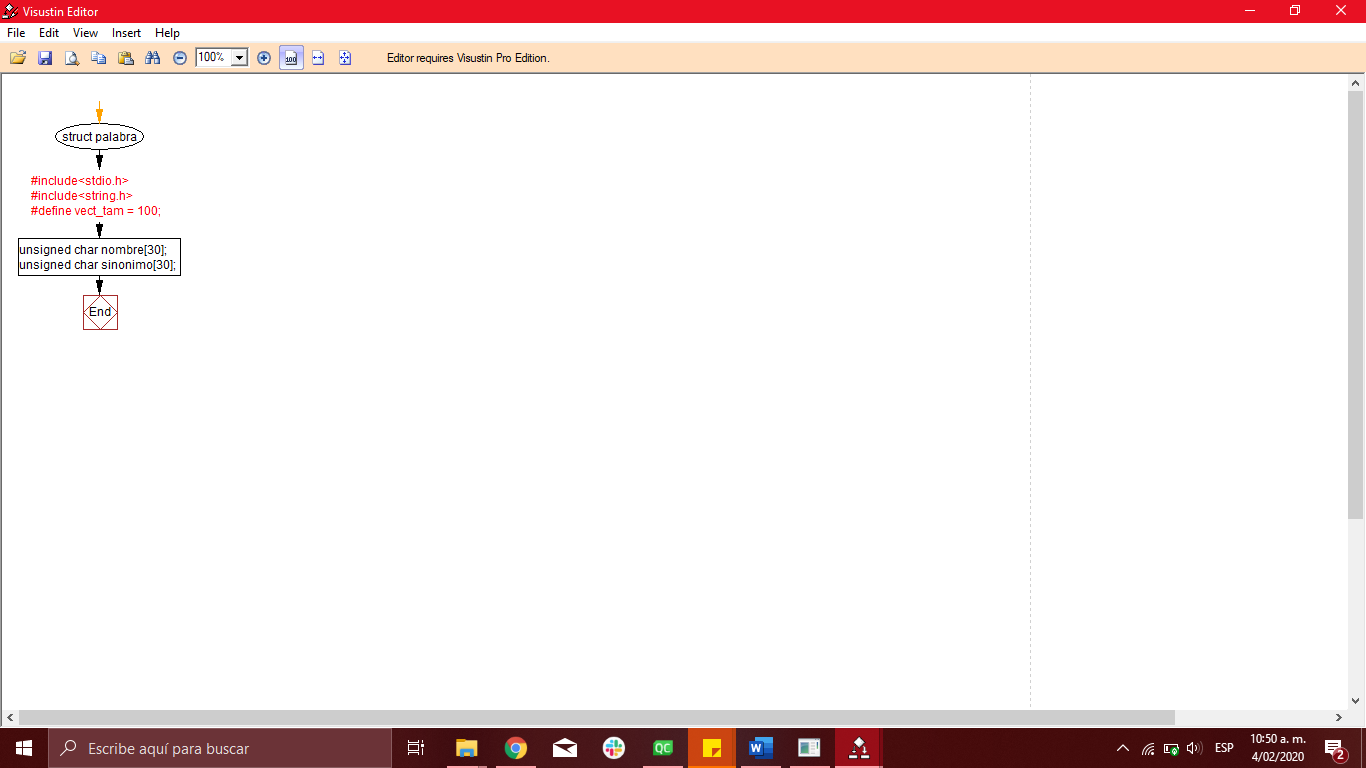
*Librerías:* contienen parámetros para las instrucciones del lenguaje.

* Librería “stdio.h”: contiene los prototipos de las funciones, macros y tipos para manipular datos de entrada y salida.[3]
* Librería “string.h”: contiene los prototipos de las funciones y macros de clasificación de caracteres.[3]

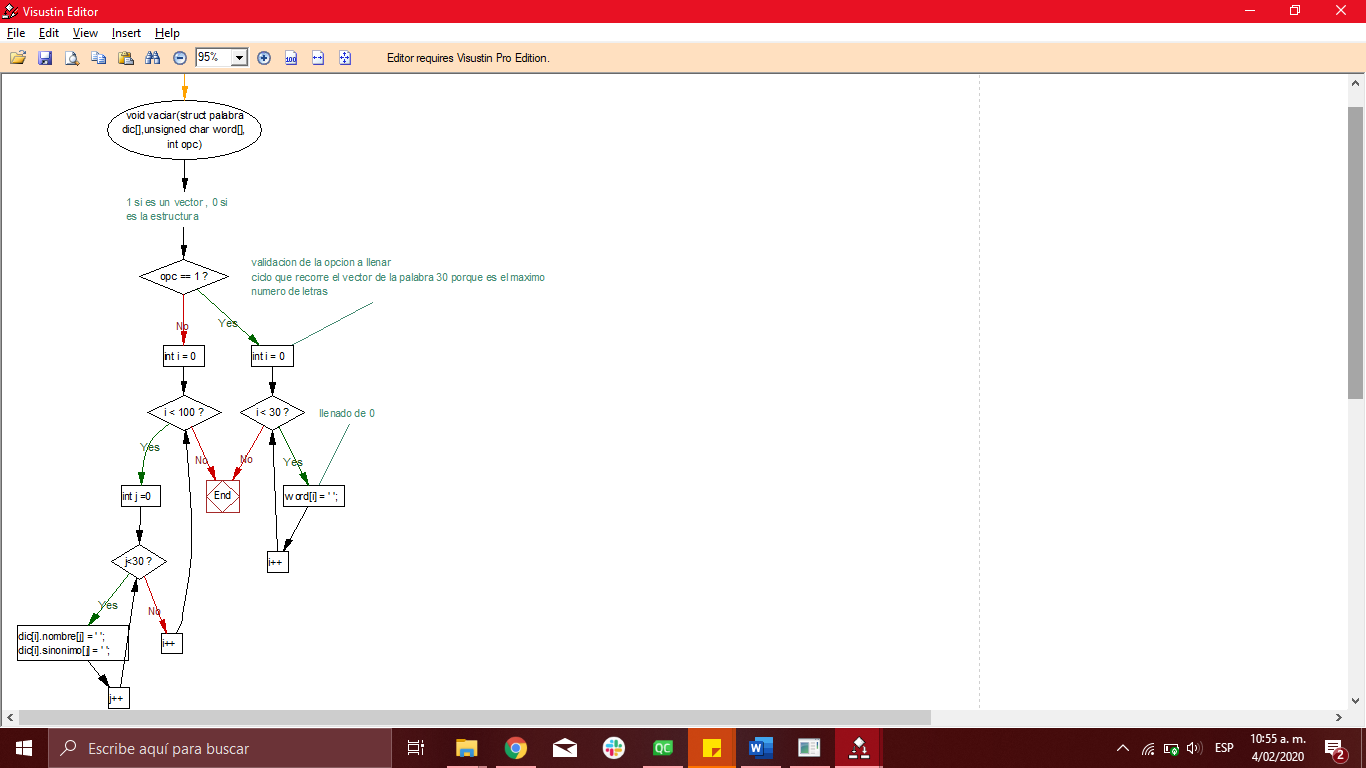
1. OBJETIVOS

* Realizar un diccionario electrónico el cual agregue, busque y liste palabras.
* Aprender a guardar datos en la memoria ram y stack, teniendo en cuenta el funcionamiento de cada una.
* Hacer uso de las librerías “stdio.h” y “string.h”

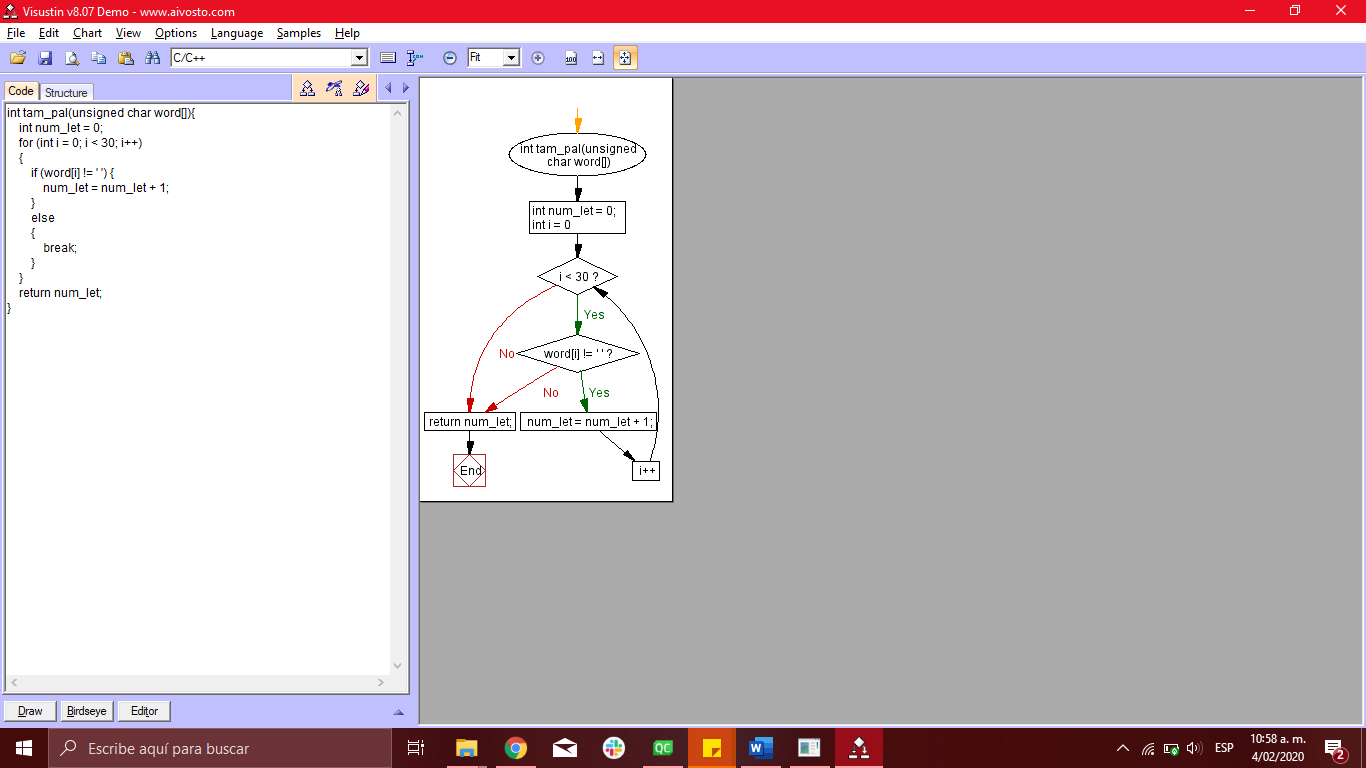
1. DIAGRAMAS DE FLUJO



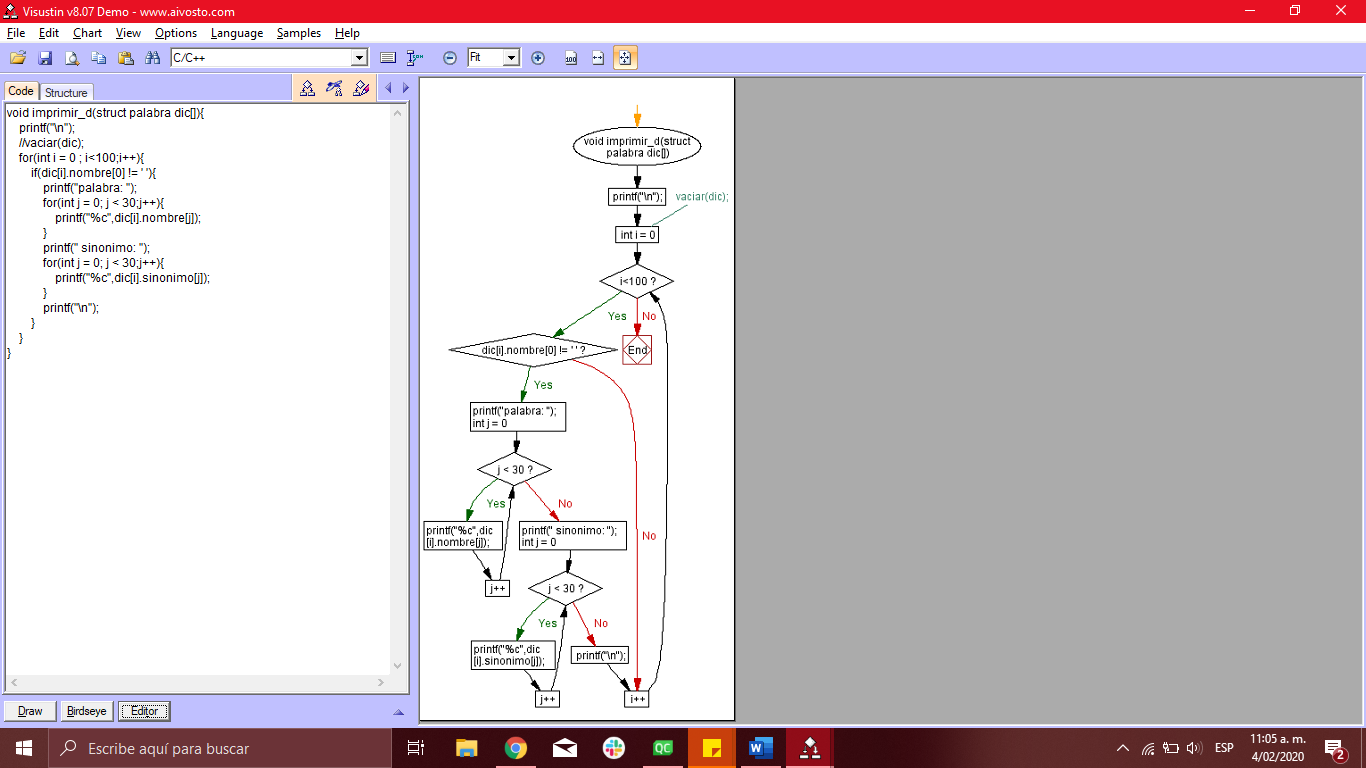
*Imagen 1: flujograma del código*



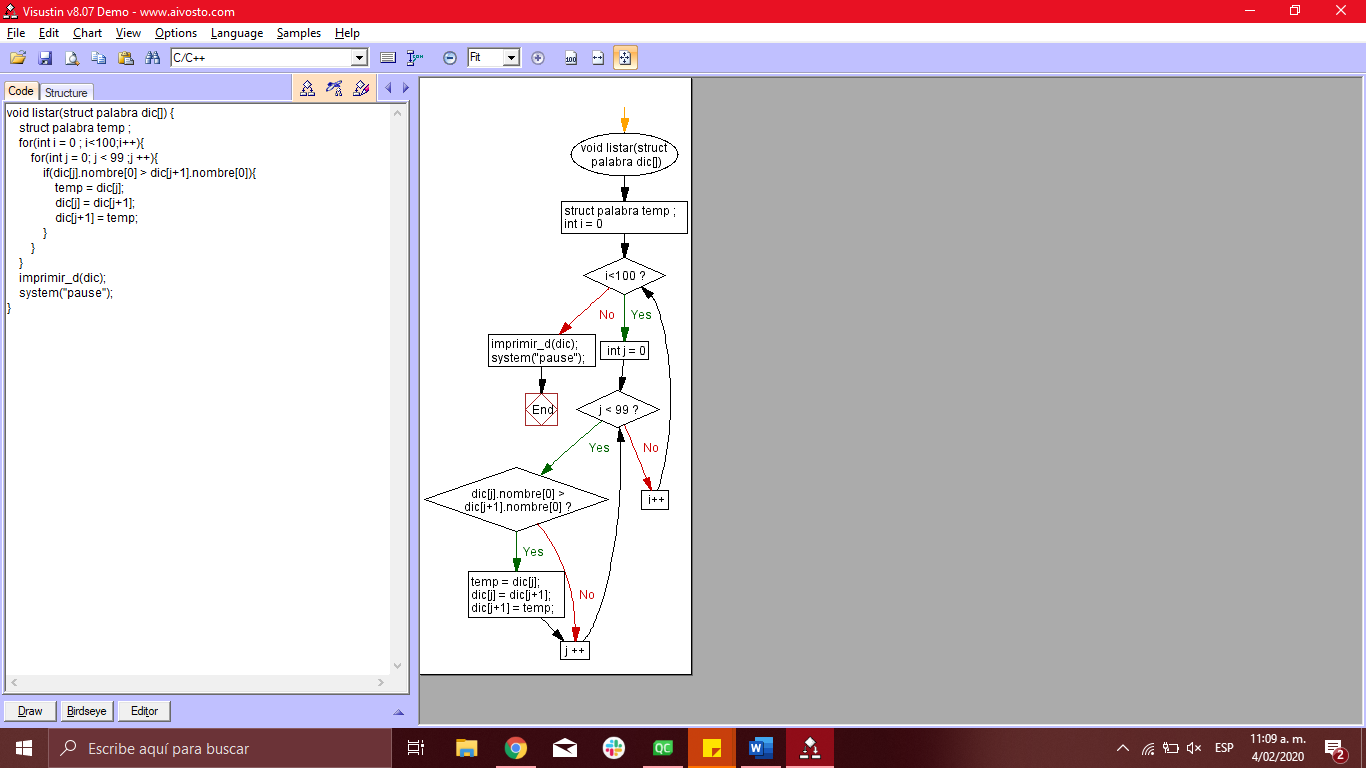
*Imagen 2: flujograma del código*



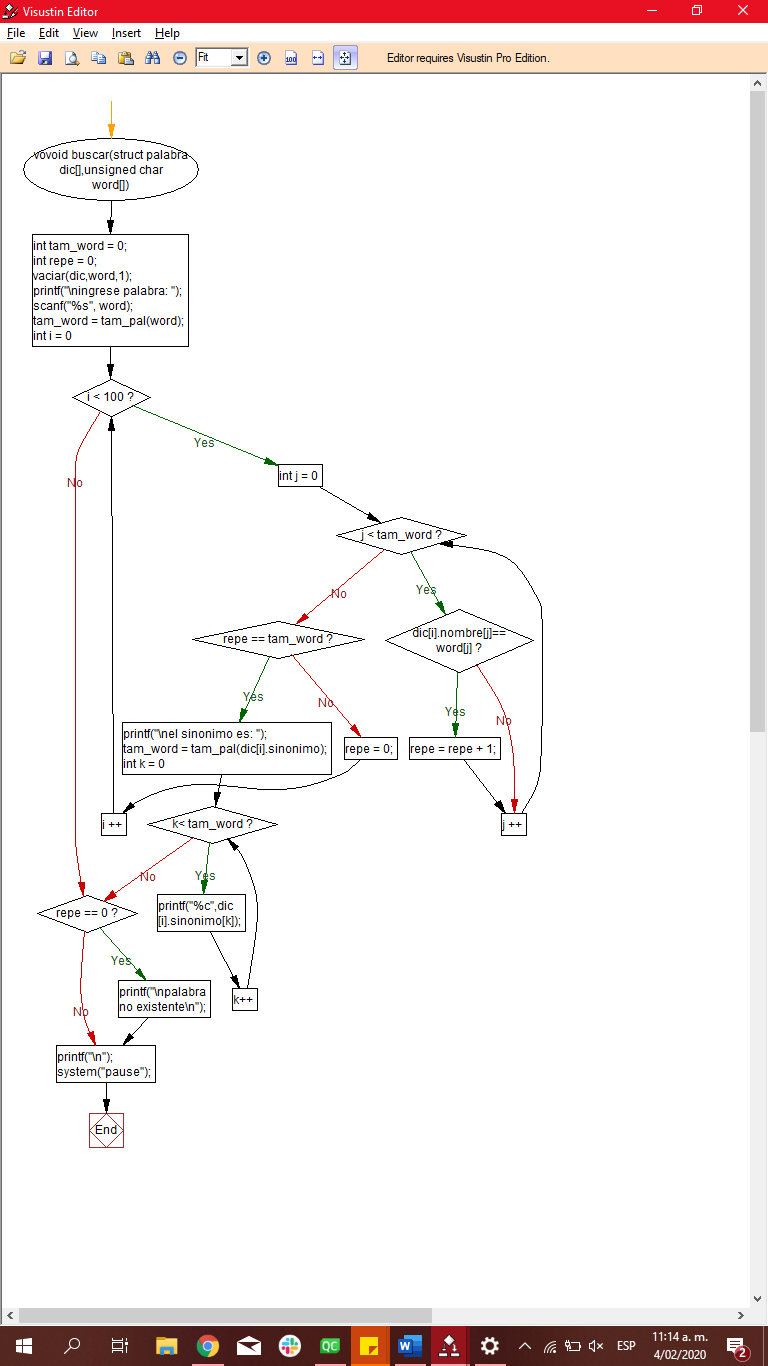
*Imagen 3: flujograma del código*



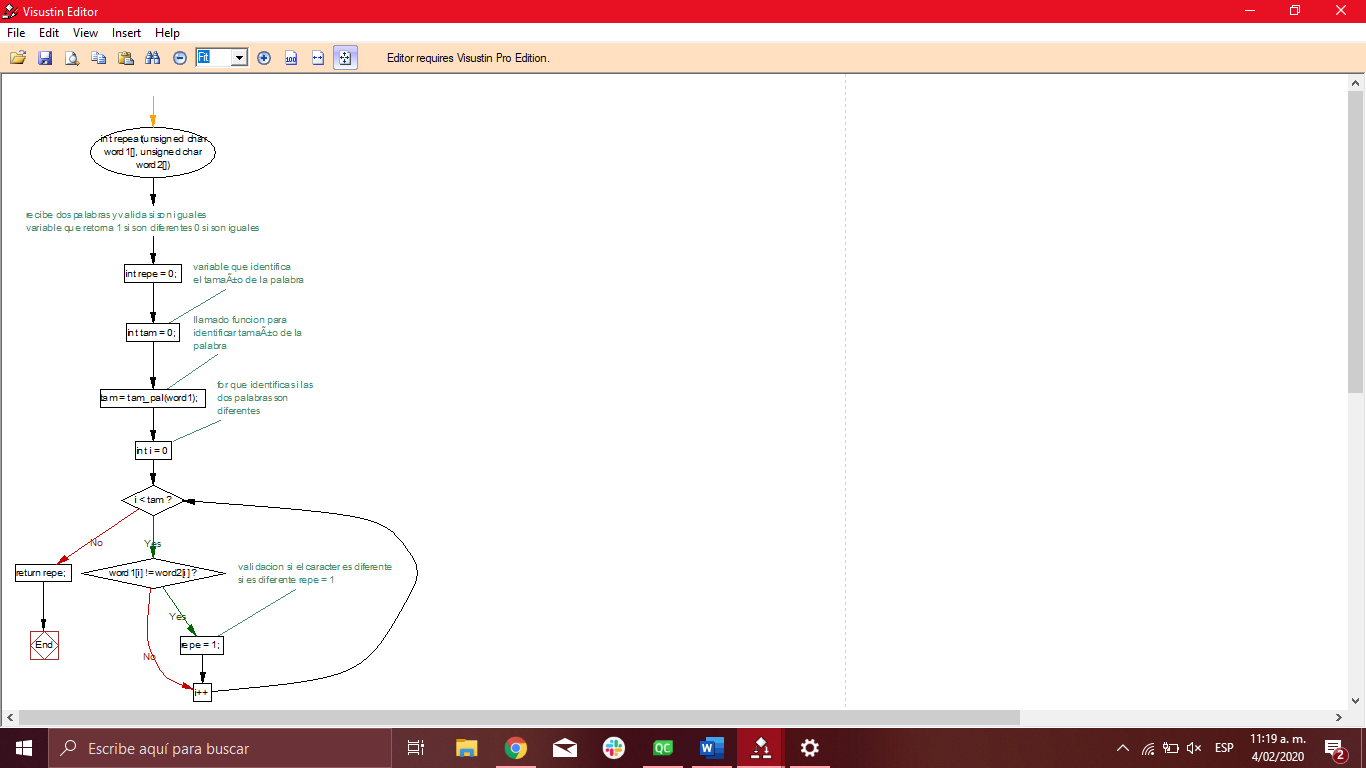
*Imagen 4: flujograma del código*



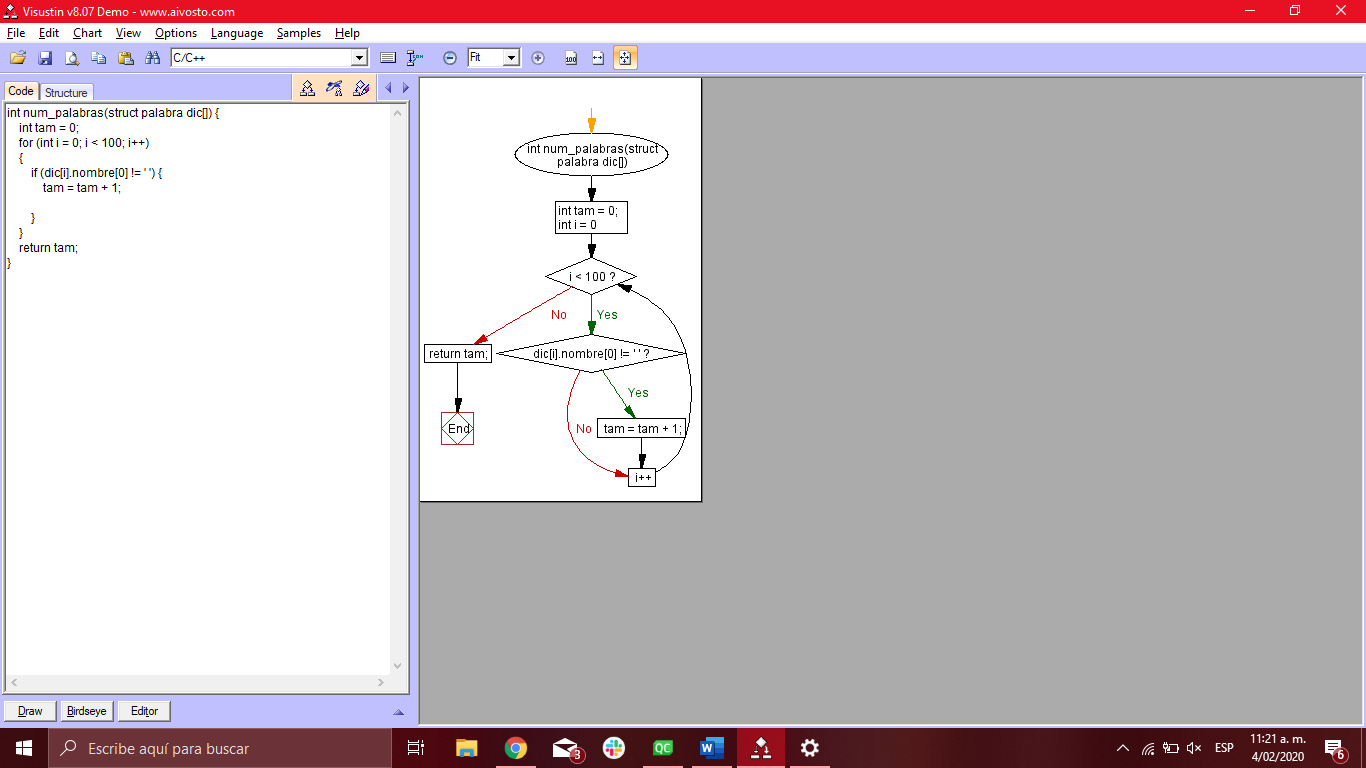
*Imagen 5: flujograma del código*



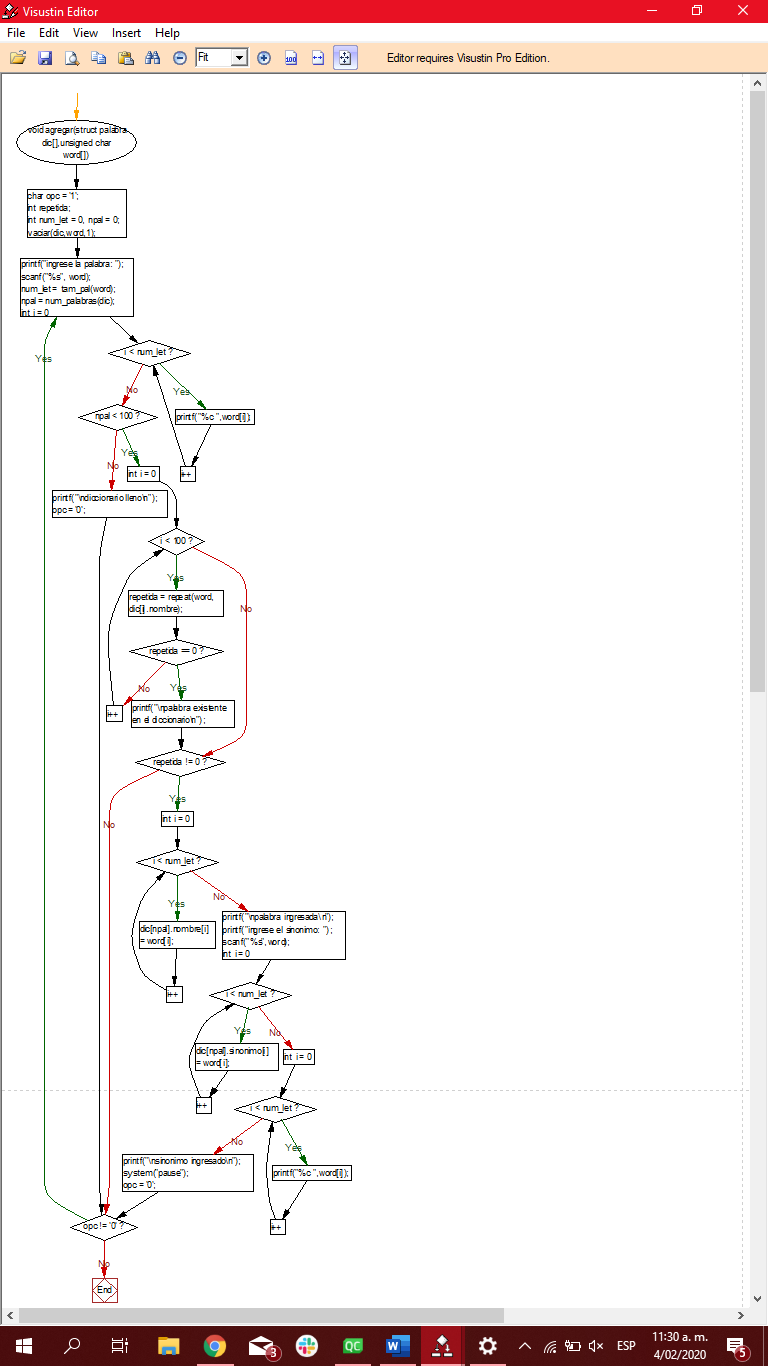
*Imagen 6: flujograma del código*



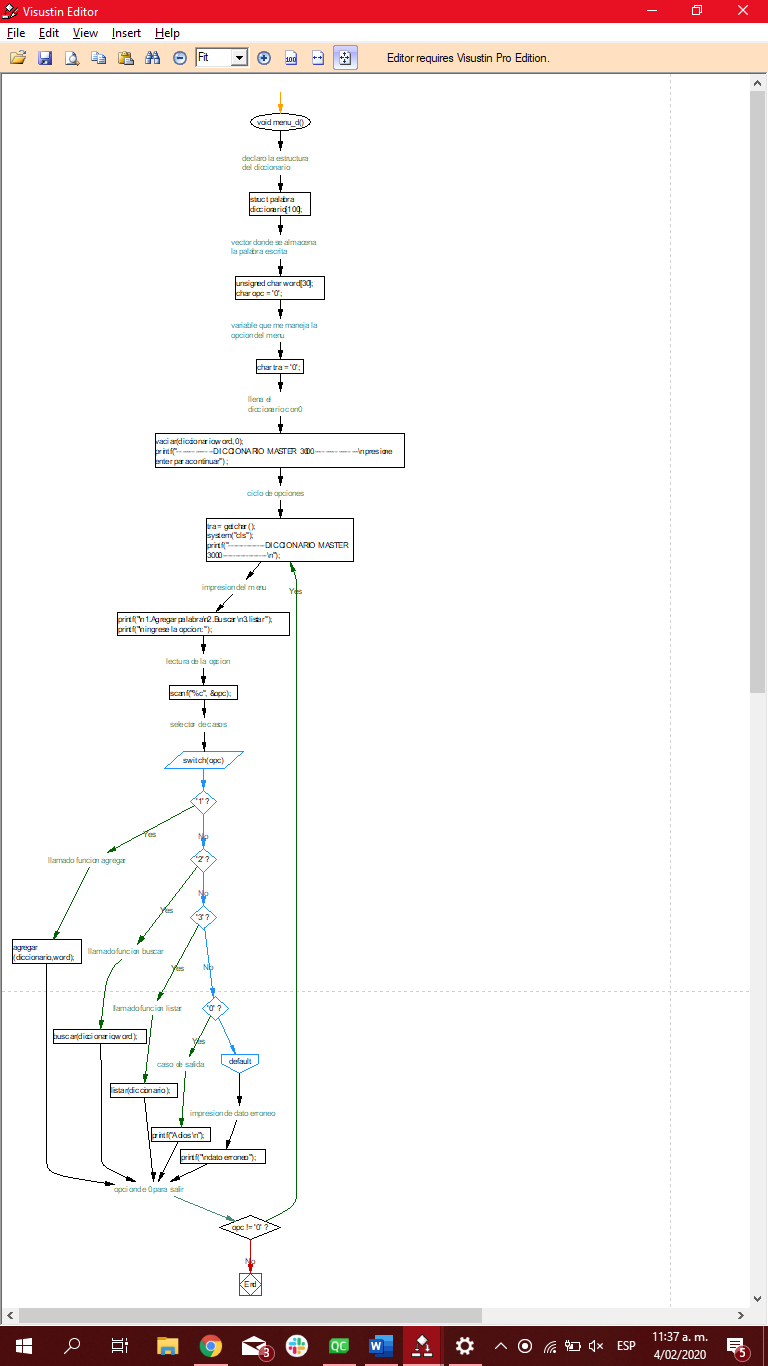
*Imagen 7: flujograma del código*



*Imagen 8: flujograma del código*

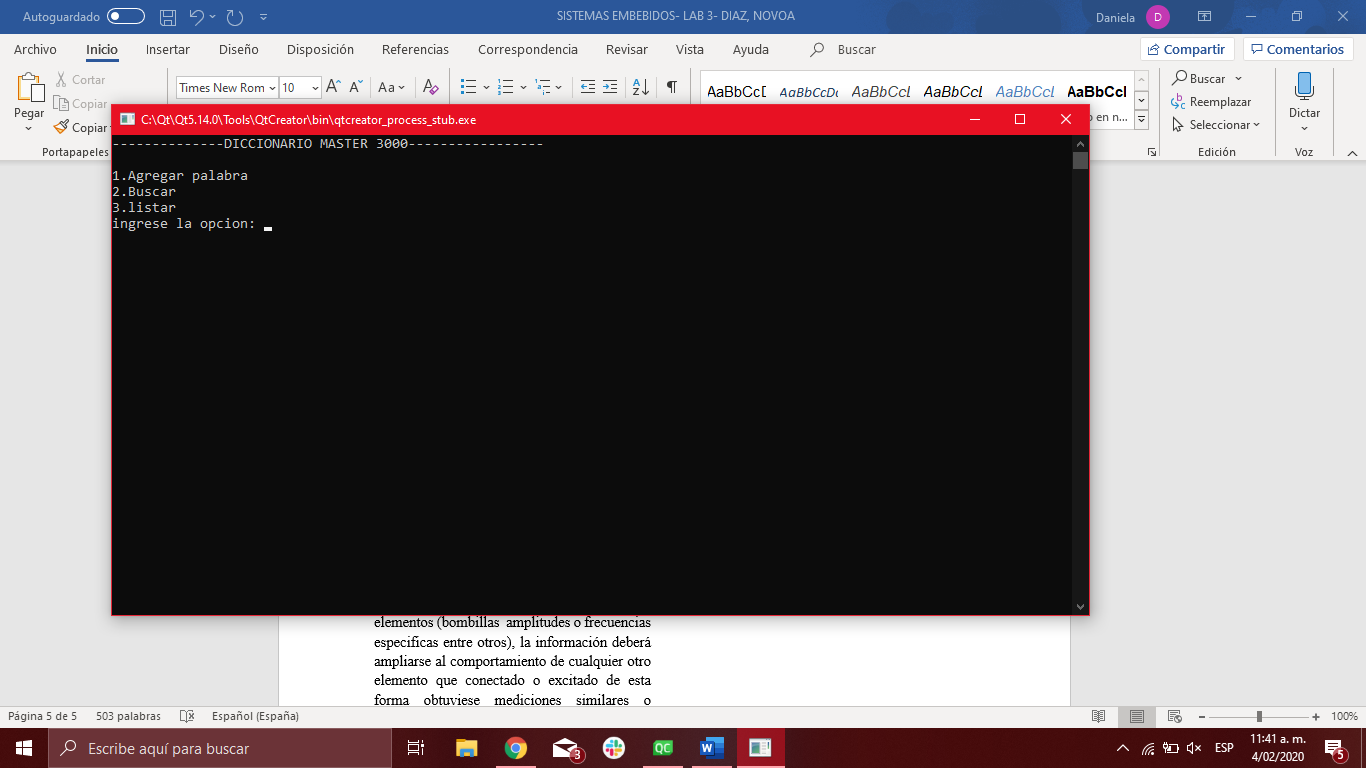


*Imagen 9: flujograma del código*

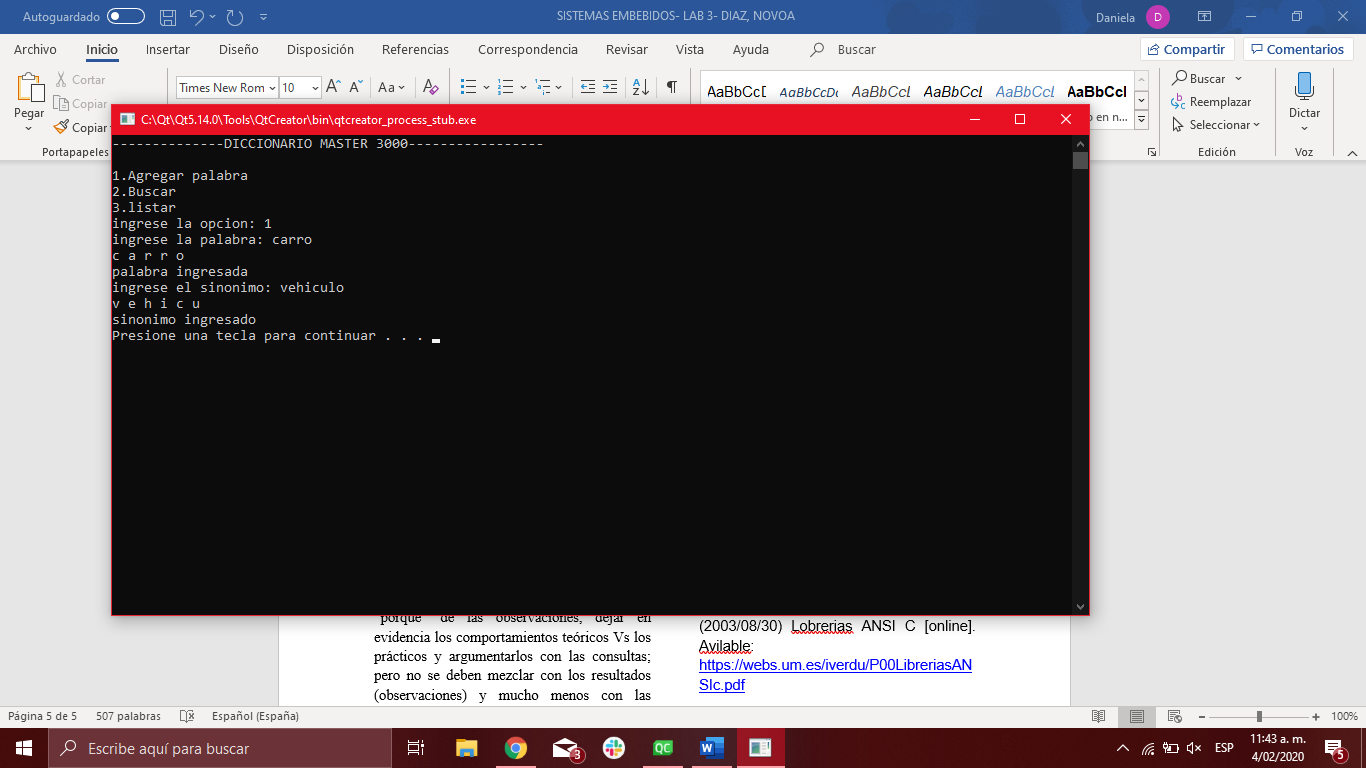


*Imagen 10: flujograma del código*

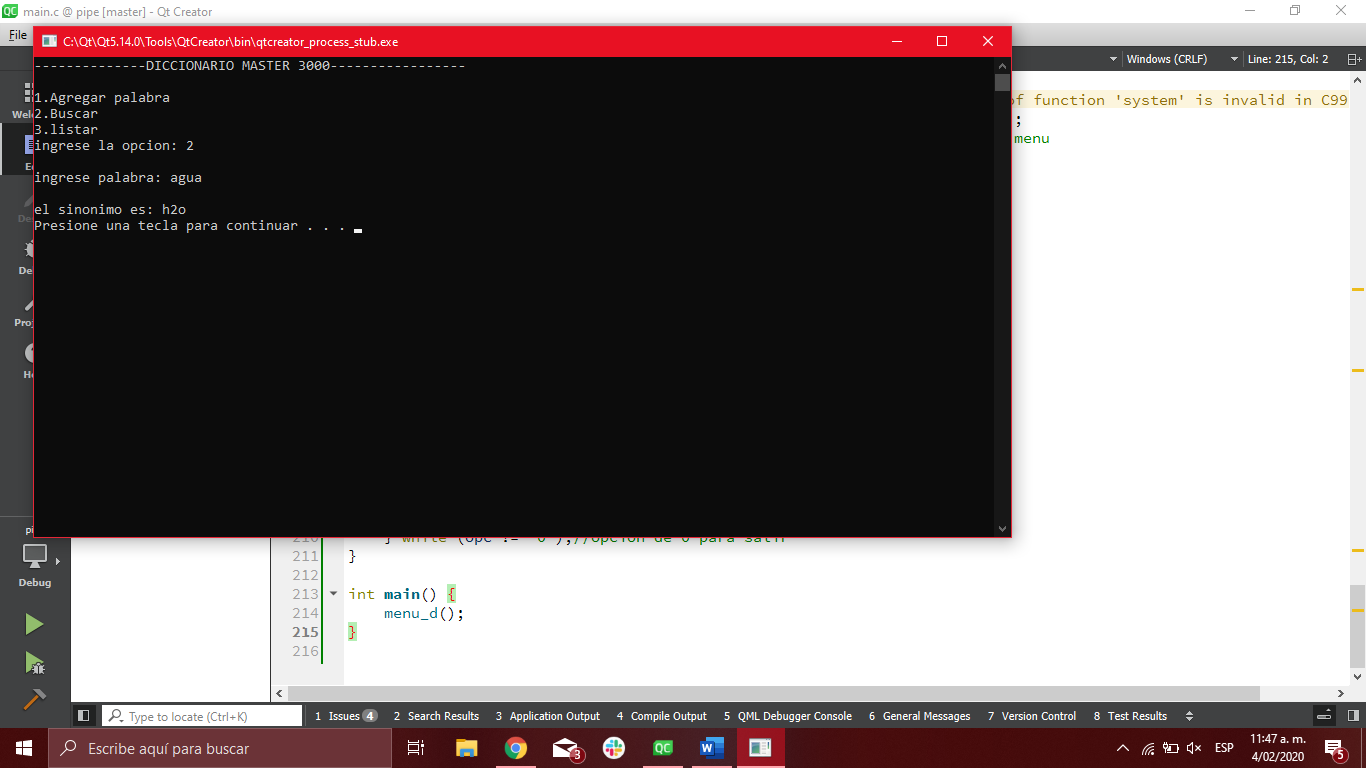
1. RESULTADOS



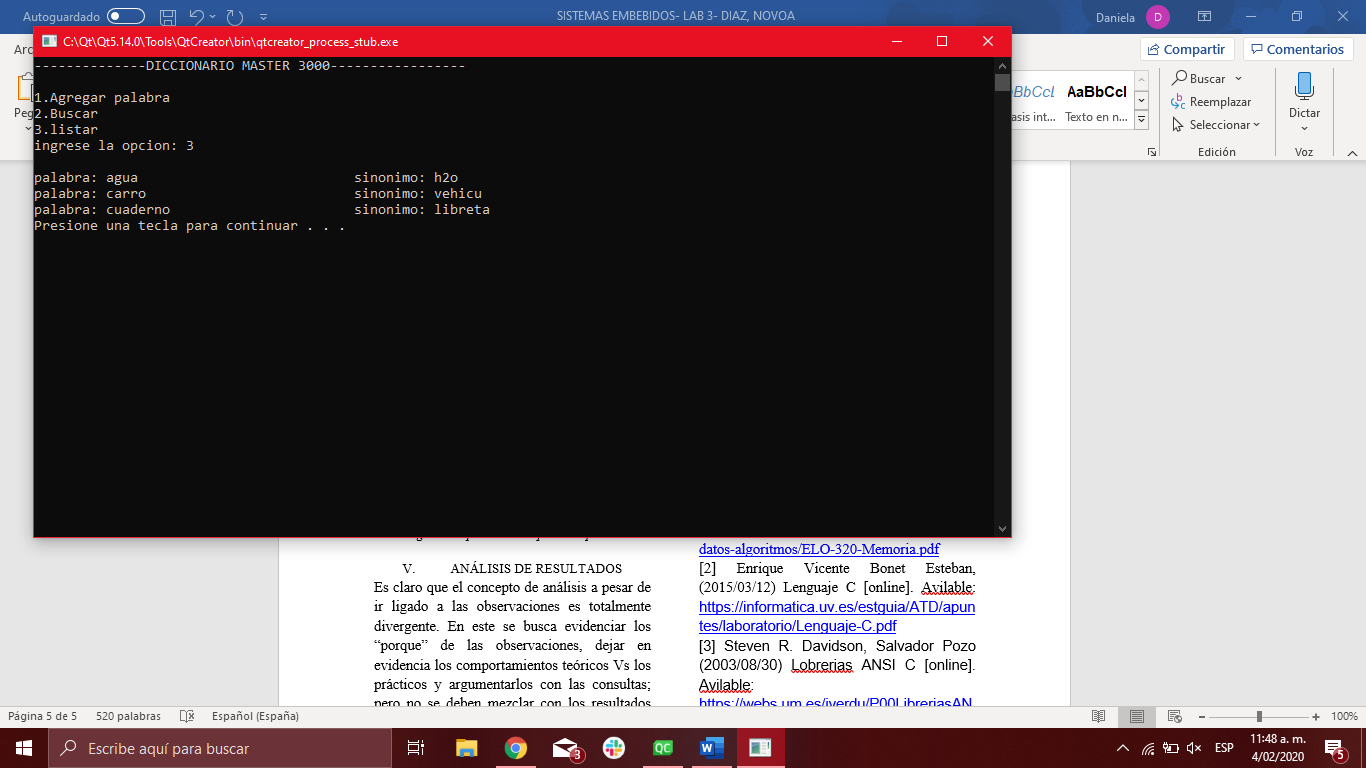
*Imagen 11: menú principal.*



*Imagen 12: opción de agregar palabra*



*Imagen 13: opción de búsqueda de palabra*



*Imagen 14: opción de listar las palabras*

1. ANÁLISIS DE RESULTADOS

Causalidades de error:

* según los resultados obtenidos y las variables analizadas; al momento de ingresar las opciones al menú principal, si se ingresa una palabra el código recibe la palabra, pero la interpreta carácter por carácter.
* Al momento de buscar, el código no reconoce una cadena por medio de subcadenas.

1. CONCLUSIONES

* A pesar de que se buscó documentación para las unciones planteadas en la guía de laboratorio, en el momento de implementarlas, no funcionaron de manera correcta; Esto puede ser debido a que la documentación encontrada manejaba otro tipo de compilador (C++), para solucionar este tipo de bugs en la versión actual del diccionario, es necesario encontrar documentación relacionada al compilador MINGW para C.
* Para el correcto funcionamiento de lo que significa un diccionario (programa que permita identificar palabra a partir Queries) es necesario hacer una actualización de versión y corregir los bugs de subcadenas, variables erróneas y tamaño del diccionario.

1. REFERENCIAS BIBLIOGRÁFIA

[1] Tomas Arredondo Vidal, (2010/04/08) Estructuras de datos y algoritmos [online]. Avilable: <http://profesores.elo.utfsm.cl/~tarredondo/info/datos-algoritmos/ELO-320-Memoria.pdf>

[2] Enrique Vicente Bonet Esteban, (2015/03/12) Lenguaje C [online]. Avilable: <https://informatica.uv.es/estguia/ATD/apuntes/laboratorio/Lenguaje-C.pdf>

[3] Steven R. Davidson, Salvador Pozo (2003/08/30) Lobrerias ANSI C [online]. Avilable: <https://webs.um.es/iverdu/P00LibreriasANSIc.pdf>

1. ANEXOS

#include<stdio.h>

#include<string.h>

#define vect\_tam = 100;

struct palabra

{

unsigned char nombre[30];

unsigned char sinonimo[30];

};

void vaciar(struct palabra dic[],unsigned char word[], int opc) {

//1 si es un vector , 0 si es la estructura

if(opc == 1){//validacion de la opcion a llenar

for (int i = 0; i < 30; i++)//ciclo que recorre el vector de la palabra 30 porque es el maximo numero de letras

{

word[i] = ' ';//llenado de 0

}

}else {

for(int i = 0;i < 100;i++){

for (int j =0;j<30;j++){

dic[i].nombre[j] = ' ';

dic[i].sinonimo[j] = ' ';

}

}

}

}

int tam\_pal(unsigned char word[]){

int num\_let = 0;

for (int i = 0; i < 30; i++)

{

if (word[i] != ' ') {

num\_let = num\_let + 1;

}

else

{

break;

}

}

return num\_let;

}

void imprimir\_d(struct palabra dic[]){

printf("\n");

//vaciar(dic);

for(int i = 0 ; i<100;i++){

if(dic[i].nombre[0] != ' '){

printf("palabra: ");

for(int j = 0; j < 30;j++){

printf("%c",dic[i].nombre[j]);

}

printf(" sinonimo: ");

for(int j = 0; j < 30;j++){

printf("%c",dic[i].sinonimo[j]);

}

printf("\n");

}

}

}

void listar(struct palabra dic[]) {

struct palabra temp ;

for(int i = 0 ; i<100;i++){

for(int j = 0; j < 99 ;j ++){

if(dic[j].nombre[0] > dic[j+1].nombre[0]){

temp = dic[j];

dic[j] = dic[j+1];

dic[j+1] = temp;

}

}

}

imprimir\_d(dic);

system("pause");

}

void buscar(struct palabra dic[],unsigned char word[]) {

int tam\_word = 0;

int repe = 0;

vaciar(dic,word,1);

printf("\ningrese palabra: ");

scanf("%s", word);

tam\_word = tam\_pal(word);

for(int i = 0; i < 100; i ++){

for(int j = 0; j < tam\_word; j ++){

if (dic[i].nombre[j]== word[j]){

repe = repe + 1;

}

}

if (repe == tam\_word){

printf("\nel sinonimo es: ");

tam\_word = tam\_pal(dic[i].sinonimo);

for(int k = 0;k< tam\_word;k++){

printf("%c",dic[i].sinonimo[k]);

}

break;

}

repe = 0;

}

if(repe == 0){

printf("\npalabra no existente\n");

}

printf("\n");

system("pause");

}

int repeat(unsigned char word1[], unsigned char word2[]) {//recibe dos palabras y valida si son iguales

int repe = 0;//variable que retorna 1 si son diferentes 0 si son iguales

int tam = 0;//variable que identifica el tamaÃ±o de la palabra

tam = tam\_pal(word1);//llamado funcion para identificar tamaÃ±o de la palabra

for (int i = 0; i < tam; i++)//for que identifica si las dos palabras son diferentes

{

if (word1[i] != word2[i]) {//validacion si el caracter es diferente

repe = 1;//si es diferente repe = 1

}

}

return repe;

}

int num\_palabras(struct palabra dic[]) {

int tam = 0;

for (int i = 0; i < 100; i++)

{

if (dic[i].nombre[0] != ' ') {

tam = tam + 1;

}

}

return tam;

}

void agregar(struct palabra dic[],unsigned char word[]) {

char opc = '1';//variable de salida de ciclo

int repetida;//variable que me indica si la palabra ya esta en el diccionario

int num\_let = 0, npal = 0;//tam = tamaÃ±o de la palabra y npal = numero de palabras en el diccionario

vaciar(dic,word,1);//llamado a la funcion llenar (llena con 0)

do//ciclo de funciones

{

printf("ingrese la palabra: ");

scanf("%s", word);//lectura de la palabra

num\_let = tam\_pal(word);//identificacion del tamaÃ±o de la palabra

npal = num\_palabras(dic);//busqueda de cuanta palabras hay en el diccionario

for (int i = 0 ; i < num\_let; i++){

printf("%c ",word[i]);

}

if (npal < 100) {//si el diccionario no esta lleno

for (int i = 0; i < 100; i++) {//recorre todo el diccionario en busqueda de una palabra repetida

repetida = repeat(word, dic[i].nombre);//funcion repeat valida si las dos palabras son iguales

if (repetida == 0) {

printf("\npalabra existente en el diccionario\n");

break;

}

}

if(repetida != 0){

for (int i = 0 ; i < num\_let; i++){

dic[npal].nombre[i] = word[i];

}

printf("\npalabra ingresada\n");

printf("ingrese el sinonimo: ");

scanf("%s",word);

for (int i = 0 ; i < num\_let; i++){

dic[npal].sinonimo[i] = word[i];

}

for (int i = 0 ; i < num\_let; i++){

printf("%c ",word[i]);

}

printf("\nsinonimo ingresado\n");

system("pause");

opc = '0';

}

}

else

{

printf("\ndiccionario lleno\n");

opc = '0';

}

} while (opc != '0');

}

void menu\_d() {

struct palabra diccionario[100];//declaro la estructura del diccionario

unsigned char word[30];//vector donde se almacena la palabra escrita

char opc = '0';

char tra = '0';//variable que me maneja la opcion del menu

vaciar(diccionario,word,0);//llena el diccionario con 0

printf("--------------DICCIONARIO MASTER 3000-----------------\npresione enter para continuar");

do//ciclo de opciones

{

tra = getchar();

system("cls");

printf("--------------DICCIONARIO MASTER 3000-----------------\n");

printf("\n1.Agregar palabra\n2.Buscar\n3.listar");//impresion del menu

printf("\ningrese la opcion: ");

scanf("%c", &opc);//lectura de la opcion

switch (opc)//selector de casos

{

case '1':

agregar(diccionario,word);//llamado funcion agregar

break;

case '2':

buscar(diccionario,word);//llamado funcion buscar

break;

case '3':

listar(diccionario);//llamado funcion listar

break;

case '0':

printf("Adios\n");//caso de salida

break;

default:

printf("\ndato erroneo");//impresion de dato erroneo

break;

}

} while (opc != '0');//opcion de 0 para salir

}

int main() {

menu\_d();

}