

**ESTRUCTURA DE DATOS Y ANÁLISIS DE ALGORITMOS
INFORME Y MANUAL**

Cristian Eduardo Espinoza Silva

Profesor: Jacqueline Köhler
Alejandro Cisterna
Ayudante: Luis Migryk
Fecha de Entrega: 8 de junio de 2017

Santiago de Chile

1 - 2017

TABLA DE CONTENIDO

CAPÍTULO 1. Introducción	5
1.1 Descripción del problema.....	5
1.2 Objetivos del software.....	7
CAPÍTULO 2. Desarrollo.....	8
2.1 Marco teórico	8
2.2 Aspectos de implementación y alcances	9
2.3 Metodología implementada.....	9
2.4 Estructura implementada.....	10
2.5 Descripción del algoritmo	11
2.6 Análisis de resultados.....	13
2.7 Grafica de rendiimiento	14
CAPÍTULO 3. Conclusión	16
CAPÍTULO 4. Referencias.....	17
CAPÍTULO 5. Manual de usuario.....	18
5.1 Introducción	18
5.2 Instrucciones.....	19
5.2.1 Descarga de MinGW	19
5.2.2 Compilación	19
5.2.3 Ejecución	26
5.3 Funcionalidades.....	28
5.4 Exitos y Posibles errores	34

ÍNDICE DE FIGURAS

Figura 1: Entrada del software.	6
Figura 2: Salida entregada por el software.	6
Figura 3: Arbol AVL equilibrado.....	8
Figura 4: Estructuras del software.....	10
Figura 5: Grafico de rendimiento del software.	15
Figura 6: Ventana ejecutar en sistema operativo Windows.	19
Figura 7: Consola de Windows.	20
Figura 8: Crear carpeta y almacenar software dentro.....	20
Figura 9: Buscando ruta de la carpeta creada.....	21
Figura 10: Compilando software.....	21
Figura 11: Creado el ejecutable del software.	22
Figura 12: Abrir consola de Linux o Terminal.	23
Figura 13: Ventana de la consola de Linux o Terminal.	23
Figura 14: Carpeta con el software dentro.	24
Figura 15: Consola de Linux con la ruta de la carpeta que contiene el software.	24
Figura 16: Compilación de software.	25
Figura 17: Carpeta con el archivo ejecutable.	25
Figura 18: Software ejecutado.....	26
Figura 19: Software ejecutado.....	27
Figura 20: Menú principal del software.	28
Figura 21: Cargando el archivo al software.	29
Figura 22: Mostrando el laberinto por pantalla.	29
Figura 23: Buscando camino mínimo hacia la llave.	30
Figura 24: Mostrando los nombres a realizarle la búsqueda.	30
Figura 25: Realizar la búsqueda de los números telefónicos, además del tiempo de ejecución de software.	31
Figura 26: Generando el archivo de salida: “TelefonosEncontrados.out”	31
Figura 27: Cerrando el software.....	32
Figura 28: Carpeta con el archivo de “TelefonosEncontrados.out” del software	32
Figura 29: Salida final del software.	33

ÍNDICE DE TABLAS

Tabla 1: “Tiempos de ejecución y orden de ejecución”	13
Tabla 2: “Tiempo y orden de ejecución del bloque principal”	13
Tabla 3: Tabla de tiempos de ejecución.	14

CAPÍTULO 1. INTRODUCCIÓN

En el presente informe se ha pedido a los alumnos de la universidad de Santiago de Chile de la asignatura Estructura de datos y análisis de algoritmos que lleven a cabo un software orientado a la búsqueda de números telefónicos, dicha problemática fue planteada por la organización ADEsetnaduyA, dicha resolución de la problemática planteada se tiene que llevar a cabo mediante un árbol de búsqueda AVL.

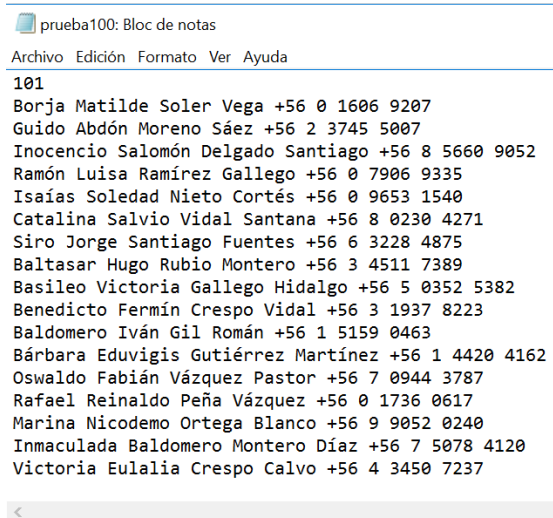
Respecto a la solución de la problemática anterior recae en encontrar los números telefónicos de las personas solicitadas, lo que conlleva a los alumnos de la asignatura crear un software que sea capaz de poder cumplir con dicho objetivo planteado anteriormente, con un paradigma de programación imperativo, además de ocupar un lenguaje de programación C.

En el transcurso del informe se dará a conocer el desarrollo y la solución implementada para dicha problemática, es decir, se describirá paso a paso el desarrollo de la solución que logro acabar con cada una de las problemáticas planteadas, para proceder a analizar los resultados y finalizar con una reflexión acerca de software entregado.

1.1 DESCRIPCIÓN DEL PROBLEMA

El problema consiste en diseñar un software que sea capaz de resolver un problema el cual consiste en encontrar un número telefónico a partir del nombre completo de la persona.

Se leen los nombres de cada persona y su respectivo número de contacto, estos datos son obtenidos mediante la lectura de un archivo.in, el cual se puede apreciar en la figura 1:



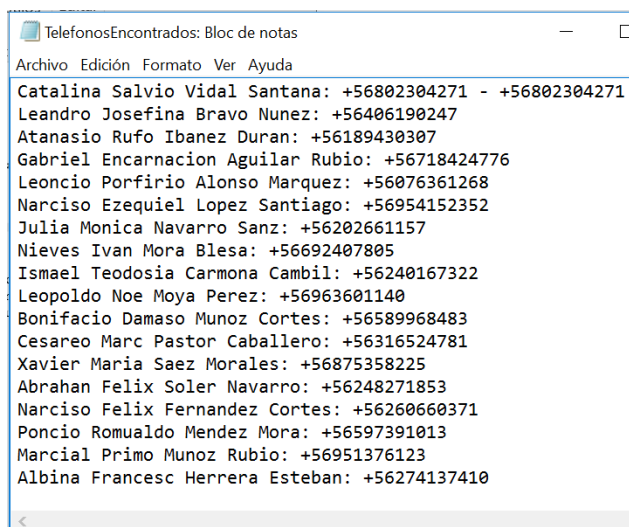
```

prueba100: Bloc de notas
Archivo Edición Formato Ver Ayuda
101
Borja Matilde Soler Vega +56 0 1606 9207
Guido Abdón Moreno Sáez +56 2 3745 5007
Inocencio Salomón Delgado Santiago +56 8 5660 9052
Ramón Luisa Ramírez Gallego +56 0 7906 9335
Isaías Soledad Nieto Cortés +56 0 9653 1540
Catalina Salvio Vidal Santana +56 8 0230 4271
Siro Jorge Santiago Fuentes +56 6 3228 4875
Baltasar Hugo Rubio Montero +56 3 4511 7389
Basileo Victoria Gallego Hidalgo +56 5 0352 5382
Benedicto Fermín Crespo Vidal +56 3 1937 8223
Baldomero Iván Gil Román +56 1 5159 0463
Bárbara Eduvigis Gutiérrez Martínez +56 1 4420 4162
Oswaldo Fabián Vázquez Pastor +56 7 0944 3787
Rafael Reinaldo Peña Vázquez +56 0 1736 0617
Marina Nicodemo Ortega Blanco +56 9 9052 0240
Inmaculada Baldomero Montero Díaz +56 7 5078 4120
Victoria Eulalia Crespo Calvo +56 4 3450 7237

```

Figura 1: Entrada del software.

La salida que entrega el software es el nombre completo de las personas con su respectivo número telefónico. Lo mencionado anteriormente se entrega en un `TelefonosEncontrados.out`, teniendo una apreciación como se muestra en la figura 2:



```

TelefonosEncontrados: Bloc de notas
Archivo Edición Formato Ver Ayuda
Catalina Salvio Vidal Santana: +56802304271 - +56802304271
Leandro Josefina Bravo Nunez: +56406190247
Atanasio Rufo Ibanez Duran: +56189430307
Gabriel Encarnacion Aguilar Rubio: +56718424776
Leoncio Porfirio Alonso Marquez: +56076361268
Narciso Ezequiel Lopez Santiago: +56954152352
Julia Monica Navarro Sanz: +56202661157
Nieves Ivan Mora Blesa: +56692407805
Ismael Teodosia Carmona Cambil: +56240167322
Leopoldo Noe Moya Perez: +56963601140
Bonifacio Damaso Munoz Cortes: +56589968483
Cesareo Marc Pastor Caballero: +56316524781
Xavier Maria Saez Morales: +56875358225
Abrahan Felix Soler Navarro: +56248271853
Narciso Felix Fernandez Cortes: +56260660371
Poncio Romualdo Mendez Mora: +56597391013
Marcial Primo Munoz Rubio: +56951376123
Albina Francesc Herrera Esteban: +56274137410

```

Figura 2: Salida entregada por el software.

1.2 OBJETIVOS DEL SOFTWARE

El objetivo general es realizar un software capaz de solucionar la problemática que presenta la organización ADEsetnaduyA, desarrollado en lenguaje de programación C, ocupando el paradigma de programación imperativo.

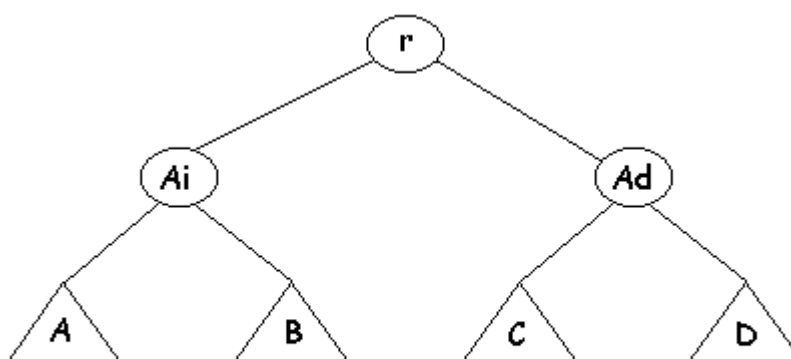
Los objetivos específicos del software son implementar lo siguiente:

- Cargar los nombres de las personas con sus respectivos números telefónicos.
- Encontrar los números telefónicos de las personas que se solicitan.
- Utilizar árbol AVL para la resolución de dicha problemática.
- Entregar un archivo de texto con los nombres completos de las personas, además de su número de contacto o contactos en caso de tener más de un número.
- Mostrar por consola el nodo actual, su respectiva altura, su hijo derecho y por ultimo su hijo izquierdo.

CAPÍTULO 2. DESARROLLO

2.1 MARCO TEÓRICO

- Árbol AVL: Es un tipo de árbol binario ordenado especial, ya que tiene la particularidad de mantenerse en equilibrio, lo que lo hace mucho eficiente. El criterio para poder aplicar alguna rotación es siempre tener una diferencia menor o igual a 1, entre las alturas de sus nodos.



Esquema general de árbol AVL.

Figura 3: Arbol AVL equilibrado.

- Estructuras: Representan colecciones de variables que son diferenciadas cada una de ellas por el tipo de variable y su respectivo nombre, estas pueden contener variables de distintos tipos. Ocupan la palabra reservada “Typedef struct”, en la figura 4 se puede apreciar los struct declarados.

2.2 ASPECTOS DE IMPLEMENTACIÓN Y ALCANCES

El principal alcance que se presenta es el cumplimiento del desarrollo del software.

Se lleva a cabo el software en el lenguaje de programación C, desarrollado en el editor de texto “Sublime Text 3”, es un editor de texto encarecido, es decir, funciona como una ayuda con la sintaxis propia del lenguaje destacando estructuras, funciones, declaraciones, etc.

La estructura del software se organiza en tres archivos, los cuales son: El archivo ejecutable que contiene el “main”, un archivo “.c” que contiene las funciones que se ocupan y por último el “.h” que contiene las estructuras que se utilizan en el transcurso del software.

Se importan distintos tipos de bibliotecas, con el fin de poder controlar e solucionar problemas que se fueron descubriendo en el transcurso del software.

2.3 METODOLOGÍA IMPLEMENTADA

La metodología implementada fue división en sub-problemas y recursión. El algoritmo implementado dentro del software se basa en la búsqueda “Inorden”, de modo que nos entrega una lista ordena alfabéticamente de los nombres cargados.

La secuencia de pasos que sigue en algoritmo implementado son las siguientes:

- El algoritmo comienza leyendo la cantidad de nombres que contendrá el árbol AVL, además de leer cada uno de sus teléfonos de contacto, con el fin de guardar en memoria cada una de ellas. Tener en cuenta que al momento de ir leyendo cada uno de sus nombres con su número telefónico, se va creando el árbol AVL ordenado y equilibrado de inmediato.
- Tener en cuenta que mediante se van agregando nuevos nodos al arbol AVL, se va actualizando las alturas de cada uno de los nodos y luego enseguida se procede a equilibrar el arbol, con el fin de mantener el arbol ordenado en todo momento, tener presente que si se producen rotaciones dentro del arbol AVL, se vuelven actualizar las alturas de cada uno de los nodos que contiene el arbol AVL.
- Luego se produce a realizar la búsqueda de los nombres solicitados para encontrar su número de contactos, el cual realiza la búsqueda con ayuda de la función de Inorden, la cual se basa en una recursión doble que va

llamando en primera instancia a su hijo derecho y luego a su hijo izquierdo. La cual se encarga de ir accediendo a cada uno de las hojas del árbol e ir comprobando si el nombre en el que se encuentra es respectivamente el que andamos buscando.

- Destacar que este proceso se realiza dentro de un ciclo for, el cual se repite tantas veces como nombres se desea buscar. Tener en cuenta que, si el nombre actual es el que se busca, se recupera el número de la persona, con el fin de obtener el número de contacto que se anda buscando.
- Finalmente, se debe recalcar que se realiza la creación del archivo final de nombre “TelefonosEncontrados.out”, con cada uno de los nombres solicitados junto con su número telefónico.

2.4 ESTRUCTURA IMPLEMENTADA

Las estructuras que se utilizan para implementar el software, son las siguientes:

```
typedef struct Informacion
{
    char primerNombre[100];
    char segundoNombre[100];
    char primerApellido[100];
    char segundoApellido[100];
    char nombreCompleto[1000];
    char numero[100];
    int bandera;
    int altura;
}Informacion;

typedef struct Nodo
{
    Informacion* info;
    struct Nodo* hijoDerecho;
    struct Nodo* hijoIzquierdo;
    int largo;
}Nodo;

typedef struct Lista
{
    int largo;
    Nodo* nodo;
}Lista;
```

Figura 4: Estructuras del software.

2.5 DESCRIPCIÓN DEL ALGORITMO

A continuación, se describe cada una de las funciones empleadas dentro del software y su procedimiento.

- **cargar(char nombre[])**: Función que carga los nombres y números telefónicos de las personas.
- **crearLista ()**: Función que se encarga de crear una lista y, inicializarla en datos Null.
- **CrearInformación (char primerNombre [], char segundoNombre [], char primerApellido [], char segundoApellido [], char numero)**: Función que se encarga de crear la variable Información con cada una de las características almacenadas en los parámetros.
- **Int mayor (Informacion* aux, Informacion* aux1)**: Función que entrega la variable que es mayor alfabéticamente.
- **AgregarNodo (Nodo* árbol, Informacion* dato)**: Función que se encarga de agregar un nodo al árbol binario ordenado.
- **CrearNodo ()**: Función que se encarga de inicializar un nodo y toda su característica almacenada en su estructura.
- **Int largoMenor (Informacion* palabra1, Informacion* palabra2)**: Función que se encarga de entregar el largo menor entre las dos palabras que se reciben como parámetro.
- **Inorden(Nodo* arbol)**: Función que se encarga de mostrar por pantalla mediante el criterio de ordenamiento de Inorden.
- **CargarNombresBuscados (char nombre [])**: Función que se encarga de cargar a la memoria los nombres que se solicitan buscar sus números telefónicos.
- **PersonasBuscadas (Nodo* listaPersonas, Lista* listaPersonasABuscar, int contador, Lista* auxLista)**: Función que se encarga de encontrar los números telefónicos de la persona que se recibe como parámetro.
- **AgregarNodoLista (Lista* lista, Nodo* aux)**: Función que se encarga de agregar un nodo a la lista.
- **MostrarPersonasBuscadas (Lista* lista)**: Función que se encarga de mostrar por pantalla las personas que se solicitaron buscar su número telefónico.

- **ArreglarLista (Lista* lista, Lista* listaPersonas, int largoLista):** Función que se encarga de verificar si una persona tiene más de un numero de contacto, en su caso, junta sus números telefónicos.
- **crearArchivo (Lista* listaPersonas, int largo):** Función que se encarga de crear el archivo de salida, con los nombres completos de las personas a buscar, además de sus números de contactos.
- **DiferenciaAltura(Nodo* nodo):** Se encarga de entregar la diferencia de alturas de los hijos derecho y izquierdo del nodo entregado.
- **Altura(Nodo* nodo):** Entrega la altura del nodo que se entrega como parámetro.
- **RotarDerecha(Nodo* arbol):** Se encarga de realizar un rotación hacia la derecha en el arbol AVL.
- **RotacionIzquierda(Nodo* arbol):** Se encarga de realizar un rotación hacia la izquierda en el arbol AVL.
- **Equilibrar(Nodo* arbol):** Función que se encarga de verificar si el arbol esta o no equilibrado, en caso contrario comprueba que tipo de rotación necesita para poder quedar en equilibrio, dentro de esas rotaciones tenemos las rotaciones simples hacia la derecho o izquierda, además de las rotaciones dobles hacia la izquierda o derecha.
- **ActualizarAltura(Nodo* aux):** Se encarga de actualizar la altura del nodo que se entrega como parámetro.
- **MaxAltura(Nodo* aux, Nodo* aux1):** Se encarga de entregar la altura máxima entre los dos nodos que se entregan como parámetro.
- **Altura(Nodo* arbol):** Se encarga de entregar la altura correspondiente al nodo que se entrega como parámetro.
- **InordenPedido(Nodo* arbol):** Se encarga de entregar cada uno de los nodos actuales, mostrando todas sus características, además de mostrar la información de cuál es su hijo derecho e izquierda.

A continuación, se deja una tabla con cada una de las funciones y sus tiempos de ejecución y ordenes de cada una de ellas:

Funciones	$T(n)$	$O(n)$
cargar(char nombre[])	$n^3 + 20n + 25$	n^3
CrearInformacion()	18	1
mayor(Informacion*aux, Informacion* aux1)	$3n+9$	n
agregarNodo(Nodo* arbol, Informacion* dato)	$n^2 + n + 6$	n^2
crearNodo()	7	1
largoMenor(Informacion* palabra1, Informacion* palabra2)	$n^2 + 2n + 2$	n^2
inorden(Nodo* arbol)	$n + 8$	n
cargarNombresBuscados(char nombres[])	$27n + 4$	n
personasBuscadas()	$2n + 16$	n
crearLista()	4	1
agregarNodoLista(Lista* lista, Nodo* aux)	7	1
mostrarPersonasBuscadas(Lista* lista)	$13n + 1$	n
arreglarLista(Lista* lista, Lista* listaPersonas)	$8n^2 + 4$	n^2
crearArchivo(Lista* listaPersonas, int largo)	$3n + 4$	n
diferenciaAltura(Nodo* aux)	$n + 1$	n
altura(Nodo* arbol)	$2n + 4$	n
rotarIzquierda(Nodo* arbol)	$2n + 10$	n
rotarDerecha(Nodo* arbol)	$2n + 10$	n
actualizarAltura(Nodo* aux)	$2n + 1$	n
equilibrar(Nodo* arbol)	$3n + 3$	n

Tabla 1: “Tiempos de ejecución y orden de ejecución”

2.6 ANÁLISIS DE RESULTADOS

Si realizamos un análisis a la función principal, el bloque principal hace el llamado a todas las funciones descritas anteriormente, quedando su tiempo de ejecución de la siguiente manera:

Funciones	$T(n)$	$O(n)$
Bloque principal	$n^4 + 2n^3 + n^2 + 5n + 11$	n^4

Tabla 2: “Tiempo y orden de ejecución del bloque principal”

Respecto a los anterior, tenemos que es un algoritmo poco eficiente, es decir, que su tiempo y orden de ejecución es muy alto.

Una de las falencias del software recae en lo mencionado anteriormente, sin embargo, esta falencia se puede solucionar implementado un algoritmo con un orden menor que cumpla las mismas expectativas.

Una posible mejora a la falencia detectada anteriormente, recae en disminuir los siglos que se utilizan dentro de la función principal del software la cual es: *cargar(char nombre[])*, la cual además es la que contiene le orden mayor dentro del algoritmo. Dicha función tiene respectivo orden ($O()$), debido a que, al momento de cargar los datos, se procede de inmediato a realizar el árbol AVL.

2.7 GRAFICA DE RENDIMIENTO

A continuación, les dejaremos los tiempos que se toma el software en resolver los problemas, se debe tener en consideración que el tiempo es tomado desde el comienzo del programa, pasando por las siguientes instrucciones:

- Primero leemos el archivo.
- Luego mostramos los nombres de las personas leídas del paso anterior.
- Luego procedemos a ingresar el nombre del archivo que contiene los nombres de las personas a buscar su número telefónico.
- A partir de los anterior procedemos a mostrar cada una de los nombres de las personas leídas en el paso anterior.
- Luego realizamos la búsqueda de los números de cada una de las personas solicitadas.
- Finalmente procedemos a crear el archivo con la solución al problema planteado.

Datos	Tiempo
0	0
10	4
50	5
100	6
1000	9
2000	9
10000	17
20000	15
100000	35
200000	49
400000	98

Tabla 3: Tabla de tiempos de ejecución.

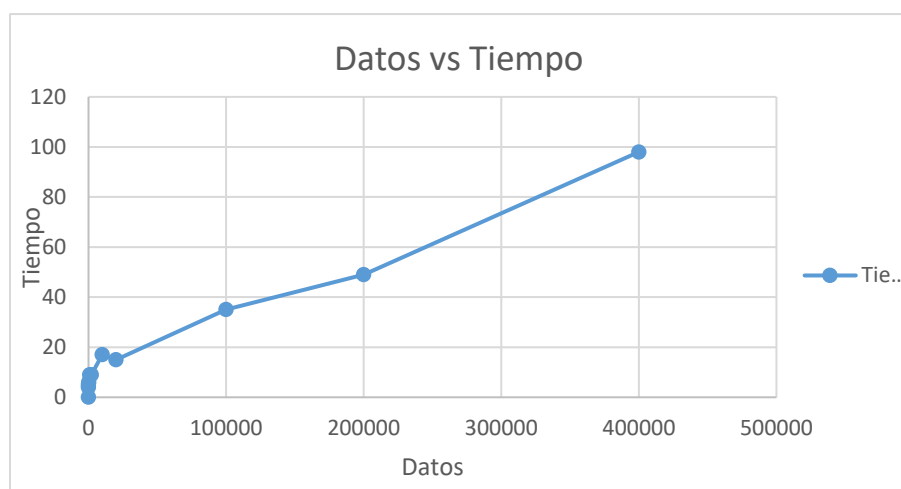


Figura 5: Grafico de rendimiento del software.

Teniendo una comparación con el laboratorio n°4, se ve una clara mejora en los tiempos que se pueden observar en la tabla de datos, esto se produce mediante que el arbol ya no es un arbol binario solamente, sino que además tenemos el arbol ordenado y a la vez en equilibrio en todo momento, esto hace que al momento de ir haciendo consultas en el algoritmo de búsqueda tenemos que descartamos mayor cantidad de arbol donde puede encontrarse nuestro objetivo.

CAPÍTULO 3. CONCLUSIÓN

En este presente “Laboratorio” se pudo llevar a ejercer la teoría que se aprendió en la catedra, pudiendo tener una visión del cómo llevar a la práctica cada uno de los elementos vistos y estudiados.

Como se ha mencionado anteriormente, se ha podido lograr la implementación en cada uno de los objetivos tanto como generales y específicos. Destacar que se llegó a la solución de cada uno de ellos mediante la implementación de un árbol AVL y la búsqueda Inorden. En esta ocasión no quedan objetivos sin cumplir, ya que el software cumple con los requisitos solicitados.

La mayor dificultad dentro del desarrollo del software fue la organización del código para obtener la solución requerida.

Para llegar a cada una de las soluciones finales se crearon varios algoritmos que no cumplían con lo que se buscaba obtener, estos han sido descartados, pero dejando una gran idea para poder llegar al algoritmo correcto, que cumplía con todas las necesidades que se buscaba.

Además, se debe destacar que los resultados del software fueron exitosos en las pruebas realizadas.

Finalmente, se implementó un algoritmo de $O(n^4)$, sin embargo, se debe dejar abierta la posibilidad de posibles mejoras al algoritmo. Una de las mejores que se pueden implementar es disminuir el uso de iteraciones en las funciones principales del algoritmo, además de poder optimizar cada una de las funciones que fueron implementadas.

CAPÍTULO 4. REFERENCIAS

- [1] Anomino. (Julio de 2007). *C con clase*.
- [2] Borjas, V. (2013). *Estructuras en C*.
- [3] Jacqueline, K. C. (s.f.). *Apuntes de la asignatura: Análisis de algoritmos y estructura de*. Chile, Universidad de Santiago de Chile, Depto. De Ingeniería en Informática. Obtenido de http://www.udesantiagovirtual.cl/moodle2/pluginfile.php?file=%2F203862%2Fmod_resource%2Fcontent%2F1%2FApuntes%20completos%20-%201-2017.pdf

CAPÍTULO 5. MANUAL DE USUARIO

5.1 INTRODUCCIÓN

En el siguiente documento se presenta una forma adecuada, explicativa y simple de cómo utilizar el software hecho.

El software principalmente consiste en la resolución del problema entregado en el laboratorio 5, que el objetivo principal es resolver la problemática que plantea la organización ADEsetnaduyA, la cual recae en encontrar el número de contacto de cada una de las personas que se solicitan.

Dentro del transcurso del manual se procede a explicar desde el primer paso que se debe hacer para poder utilizar el software de una manera correcta.

Se recomienda tener poseer un lenguaje técnico mayor al promedio para poder aprovechar las ventajas que entrega el software.

Por ultimo debemos destacar que el software esta implementado en un paradigma imperativo, que principalmente consta de declaraciones que se ejecutan secuencialmente.

5.2 INSTRUCCIONES

A continuación, se procede a explicar cada uno de los pasos a seguir para poder compilar el software que soluciona la problemática planteada por la organización ADEsetnaduyA.

5.2.1 Descarga de MinGW

Para poder compilar el software credo se necesita descargar lo siguiente, con el fin que los comandos ingresados sean conocidos por su equipo.

Se puede obtener del siguiente link:

- <https://www.fdi.ucm.es/profesor/luis/fp/devtools/MinGW.html>

5.2.2 Compilación

5.2.2.1 Windows:

1. Debemos abrir la consola de Windows, existen dos opciones las cuales son:
 - a. Apretar Windows + R
 - b. Abrir la ventana de ejecutar y escribir “cmd”.

Una de las opciones se puede apreciar en la Figura 6, que viene a continuación:

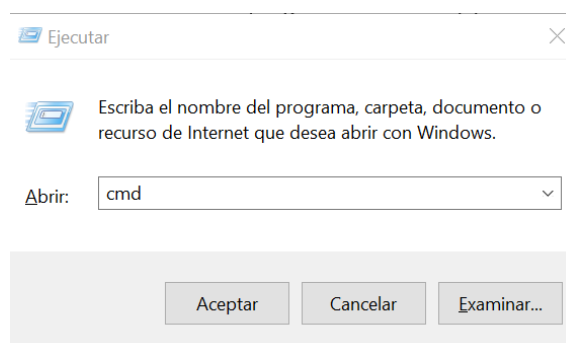


Figura 6: Ventana ejecutar en sistema operativo Windows.

2. Luego de realizar el anterior paso, se abrirá la consola como mostrará en la Figura 7:

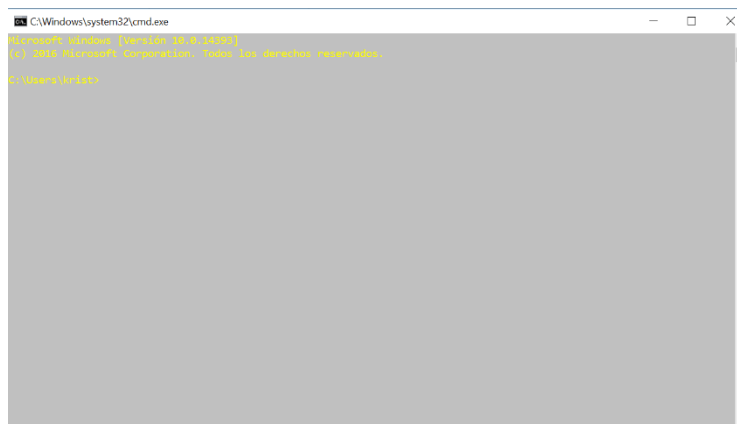


Figura 7: Consola de Windows.

3. Almacenar el software dentro de una carpeta para poder tener todo lo necesario dentro de ella, lo recomendado es crear dicha carpeta en el escritorio de su máquina, como se muestra en la Figura 8:

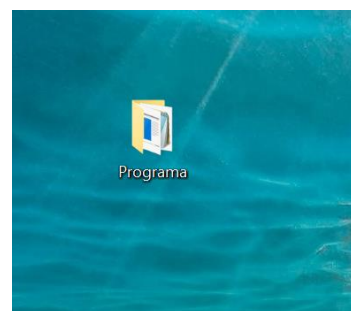
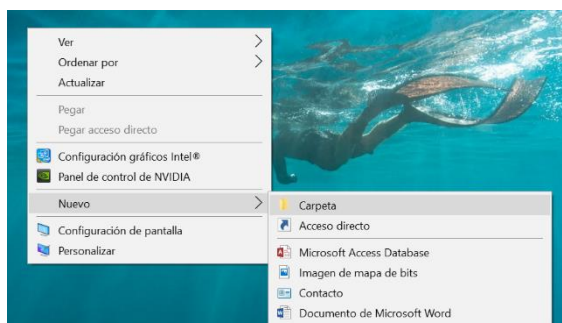
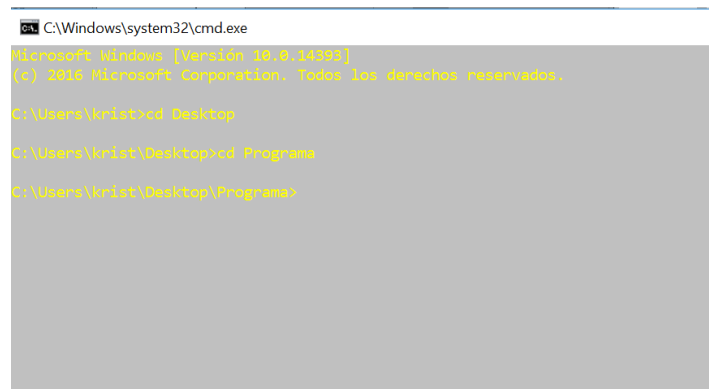


Figura 8: Crear carpeta y almacenar software dentro.

4. Luego volvemos a la consola de Windows, para poder ejecutar el software y seguir los siguientes pasos:
 - a. Primer paso, dejamos la consola de Windows en la ruta donde tenemos almacenado nuestro programa, de la manera que muestra en la Figura 9:



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 10.0.14393]
(c) 2016 Microsoft Corporation. Todos los derechos reservados.

C:\Users\krist>cd Desktop

C:\Users\krist\Desktop>cd Programa

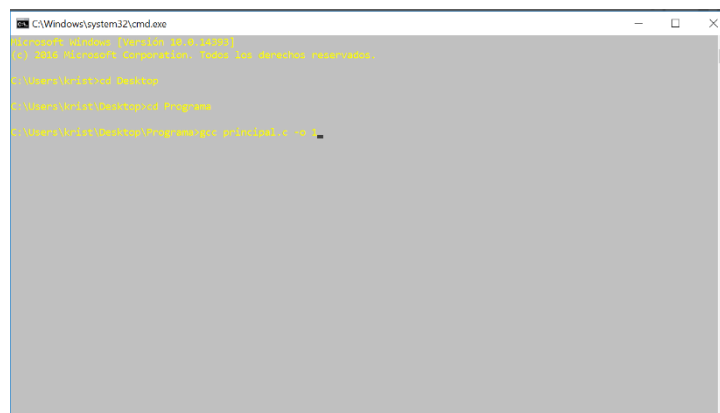
C:\Users\krist\Desktop\Programa>
```

Figura 9: Buscando ruta de la carpeta creada.

b. Luego comenzamos la compilación, escribiendo lo siguiente en la consola de Windows:

- gcc (Nombre del archivo).c -o (Nombre del ejecutable)
- (Nombre del ejecutable).exe

La Figura 10 entrega las instrucciones con mayor claridad:



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 10.0.14393]
(c) 2016 Microsoft Corporation. Todos los derechos reservados.

C:\Users\krist>cd Desktop

C:\Users\krist\Desktop>cd Programa

C:\Users\krist\Desktop\Programa>gcc principal.c -o 1
```

Figura 10: Compilando software

- c. Luego se crea el archivo ejecutable en la carpeta donde tenemos almacenado el software, como muestra la Figura 11:

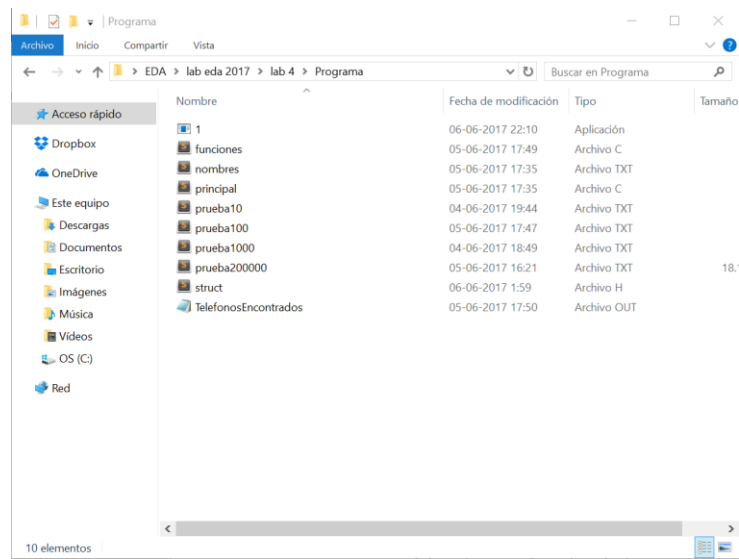


Figura 11: Creado el ejecutable del software.

5.2.2.2 Linux

1. Debemos abrir la consola de Linux o también llamada terminal, existen dos opciones las cuales son:
 - a. Apretar clic derecho y colocar la opción abrir terminal.
 - b. Buscar en los programas de Linux y colocar “Terminal”

Una de las opciones se puede apreciar en la Figura 12 que se observa abajo:

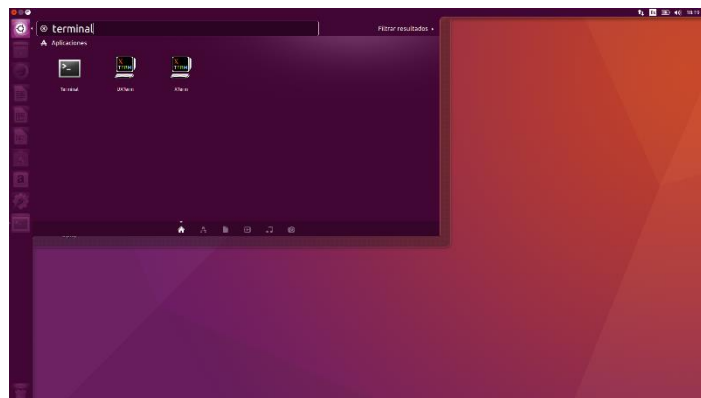


Figura 12: Abrir consola de Linux o Terminal.

2. Luego de realizar el anterior paso, se abrirá la consola como mostrará la Figura 13:

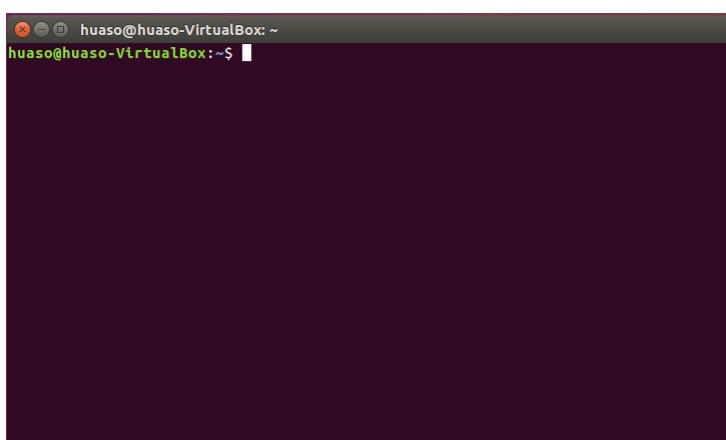


Figura 13: Ventana de la consola de Linux o Terminal.

3. Almacenar el software dentro de una carpeta para poder tener todo lo necesario dentro de ella, lo recomendado es crear dicha carpeta en el escritorio de su máquina, como se muestra en la Figura 14:

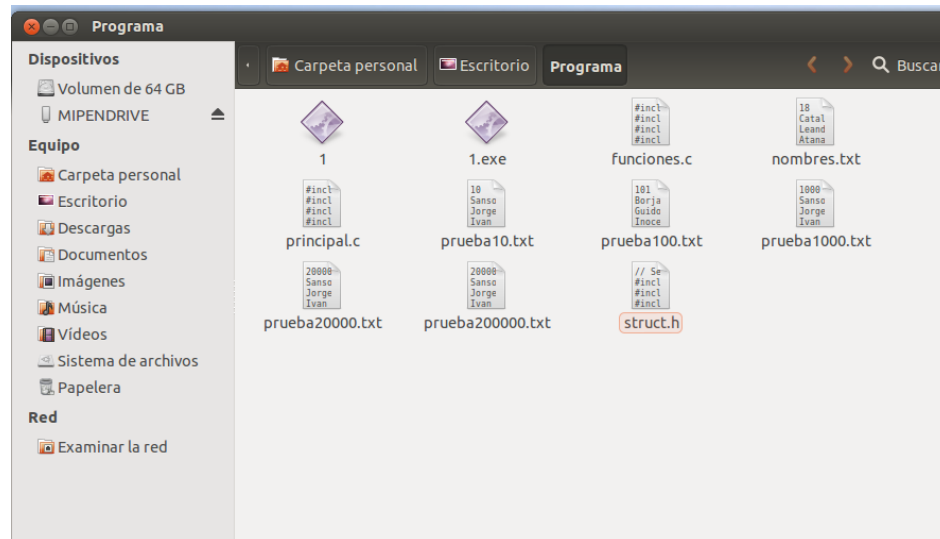


Figura 14: Carpeta con el software dentro.

4. Luego volvemos a la consola de Linux, para poder ejecutar el software como muestra en la Figura 15:
 - a. Primer paso, dejamos la consola de Linux en la ruta donde tenemos almacenado nuestro programa, de la manera que muestra la siguiente imagen:

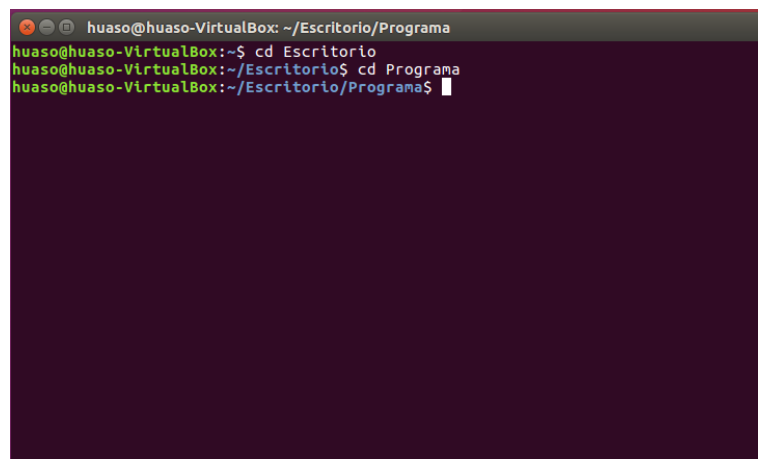
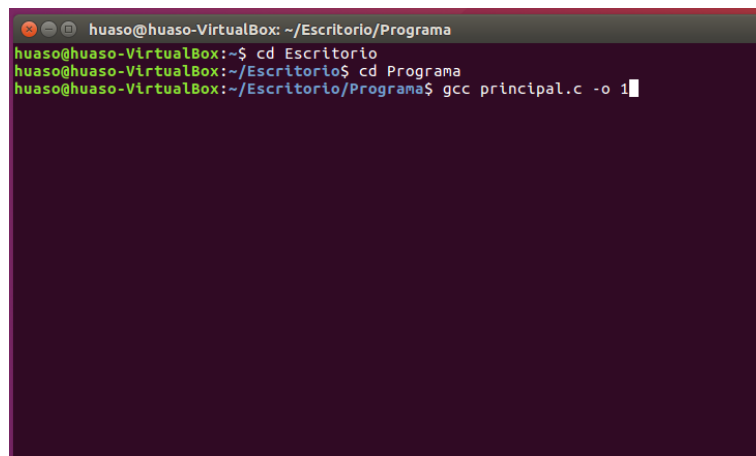


Figura 15: Consola de Linux con la ruta de la carpeta que contiene el software.

b. Luego comenzamos la compilación, escribiendo lo siguiente en la consola de Linux:

- gcc (Nombre del archivo).c -o (Nombre del ejecutable)
- ./(Nombre del ejecutable)

La Figura 16 entrega las instrucciones con mayor claridad:



```

huaso@huaso-VirtualBox: ~/Escritorio/Programa
huaso@huaso-VirtualBox:~$ cd Escritorio
huaso@huaso-VirtualBox:~/Escritorio$ cd Programa
huaso@huaso-VirtualBox:~/Escritorio/Programa$ gcc principal.c -o 1

```

Figura 16: Compilación de software.

c. Luego se crea el archivo ejecutable en la carpeta donde tenemos almacenado el software, como muestra Figura 17:

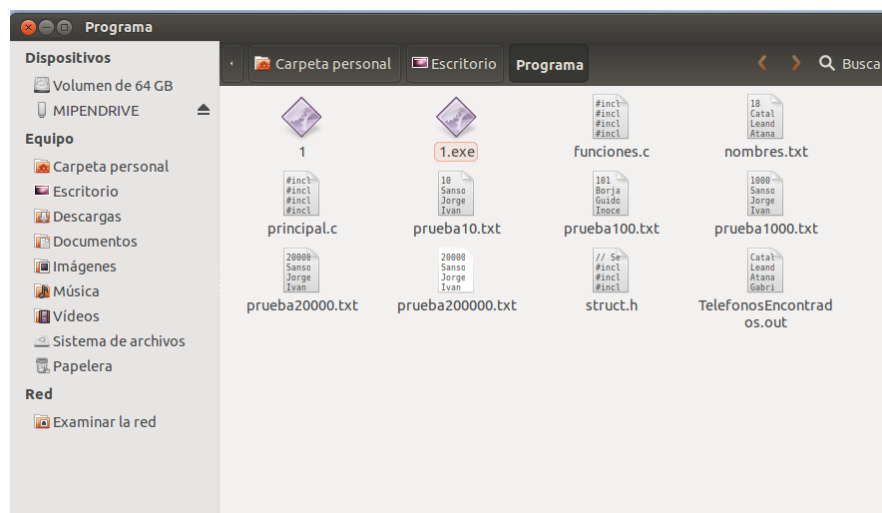


Figura 17: Carpeta con el archivo ejecutable.

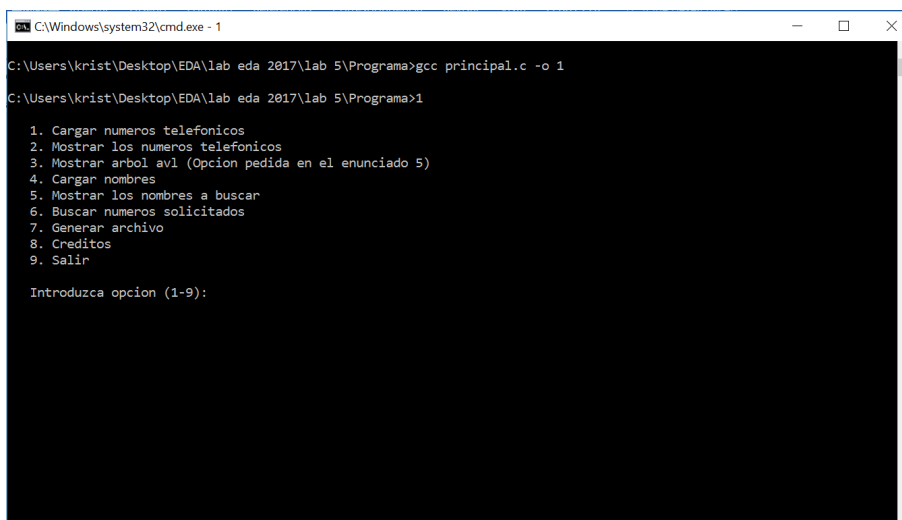
5.2.3 Ejecución

Teniendo el archivo “.exe” o “.out” en Windows y Linux respectivamente se procede a ejecutarlo cada uno en su ambiente.

Antes de llevar a cabo dicha acción se debe verificar que los archivos que contiene los nombres de las personas con sus respectivos números telefónicos estén en la misma carpeta donde se tiene almacenado dicho software, además de corroborar que los archivos donde se tiene almacenado cada uno de los caminos estén con la extensión “.txt”.

5.2.3.1 Windows

1. Se vuelve a la consola de comandos de Windows, colocando la siguiente línea para efectuar la ejecución, en la Figura 18 se puede apreciar de mejor forma:



```

C:\Windows\system32\cmd.exe - 1
C:\Users\krist\Desktop\EDA\lab_eda_2017\lab_5\Programa>gcc principal.c -o 1
C:\Users\krist\Desktop\EDA\lab_eda_2017\lab_5\Programa>1
1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir
Introduzca opcion (1-9):

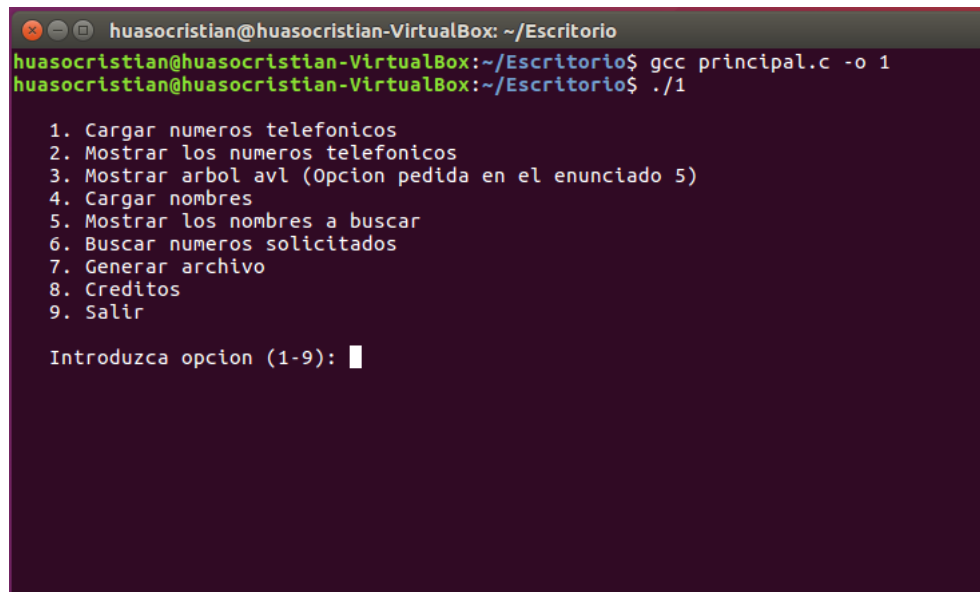
```

Figura 18: Software ejecutado.

- Al terminar el proceso anterior, se puede observar en la imagen anterior que el software está ejecutado y listo para ser manipulado.

5.2.3.2 Linux

1. Se vuelve a la consola de comandos de Linux, colocando la siguiente línea para efectuar la ejecución, en la Figura 19 se puede apreciar de mejor forma:



```
huasocristian@huasocristian-VirtualBox: ~/Escritorio
huasocristian@huasocristian-VirtualBox:~/Escritorio$ gcc principal.c -o 1
huasocristian@huasocristian-VirtualBox:~/Escritorio$ ./1

1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9): █
```

Figura 19: Software ejecutado.

- Al terminar el proceso anterior, se puede observar en la imagen anterior que el software está ejecutado y listo para ser manipulado.

5.3 FUNCIONALIDADES

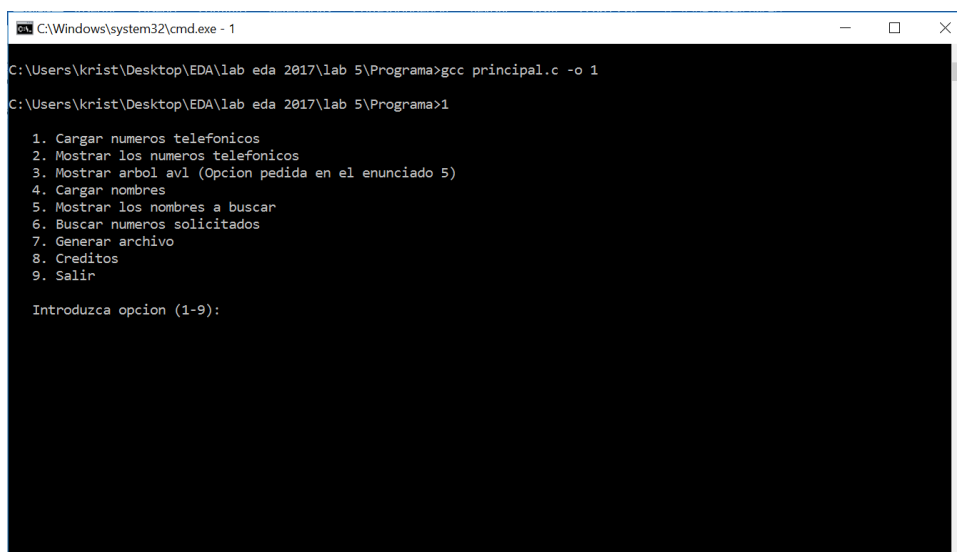
El software tiene la funcionalidad de encontrar el número telefónico de las personas que se solicitan.

El usuario tiene interacción con el software que recae en ingresar el nombre del archivo que contiene los nombres de las personas y sus respectivos números telefónicos que se desea abrir, tener en cuenta que este archivo tendrá una entrada “.txt”.

El resultado entregado por el software se verá expuesto en un archivo de salida con extensión “.out”.

A continuación, se realizará un procedimiento de los pasos a seguir para hacer uso del software de una buena manera:

- Primero debemos volver a la consola y ejecutar el software, entregando una ventana como muestra la Figura 20:



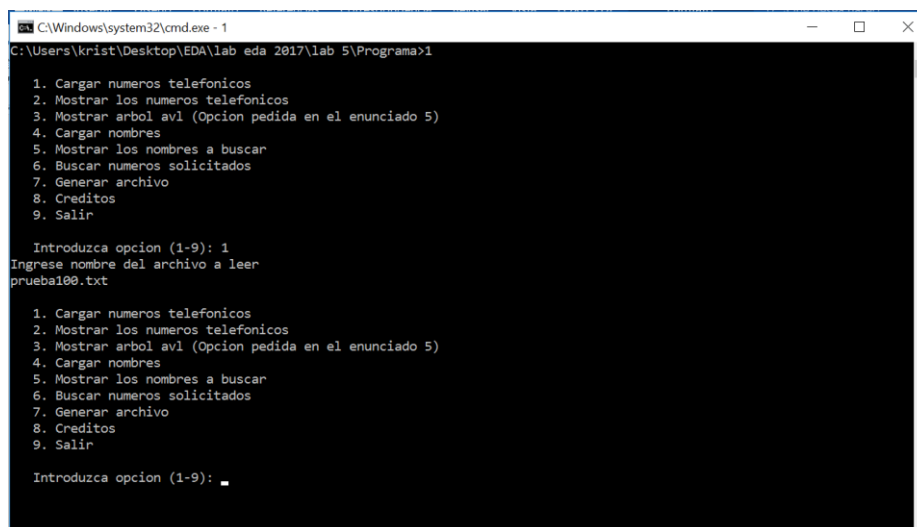
```
C:\Windows\system32\cmd.exe - 1
C:\Users\krist\Desktop\EDA\lab_eda_2017\lab_5\Programa>gcc principal.c -o 1
C:\Users\krist\Desktop\EDA\lab_eda_2017\lab_5\Programa>1

1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9):
```

Figura 20: Menú principal del software.

- Luego de eso debemos seleccionar la opción de ingresar el nombre del tablero, como se muestra en la Figura 21:



```

C:\Windows\system32\cmd.exe - 1
C:\Users\krist\Desktop\EDA\lab_eda_2017\lab_5\Programa>1

1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9): 1
Ingrese nombre del archivo a leer
prueba100.txt

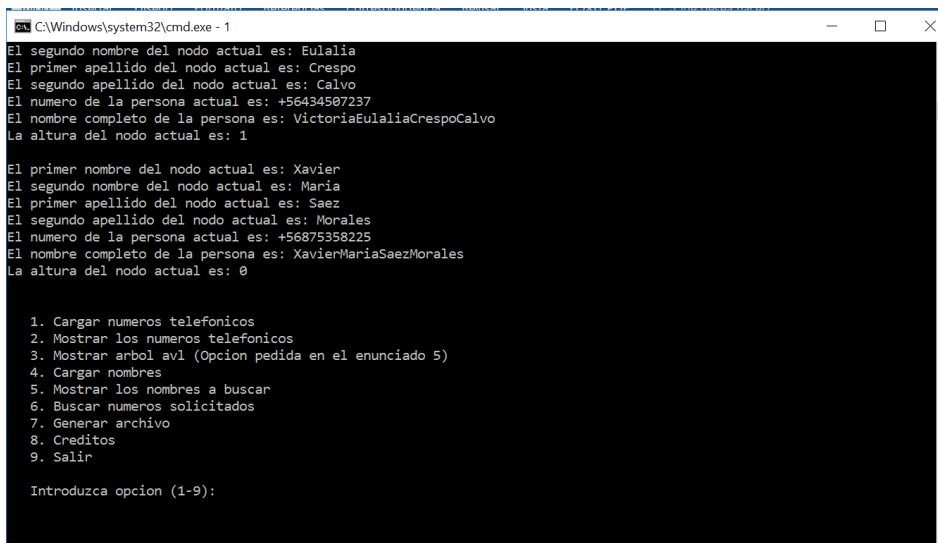
1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9):

```

Figura 21: Cargando el archivo al software.

- Luego de eso debemos volver a ingresar una opción, en este caso ocuparemos la opción número 2 que nos muestra los nombres leídos, se debe hacer como se muestra en la Figura 22:



```

C:\Windows\system32\cmd.exe - 1
El segundo nombre del nodo actual es: Eulalia
El primer apellido del nodo actual es: Crespo
El segundo apellido del nodo actual es: Calvo
El numero de la persona actual es: +56434507237
El nombre completo de la persona es: VictoriaEulaliaCrespoCalvo
La altura del nodo actual es: 1

El primer nombre del nodo actual es: Xavier
El segundo nombre del nodo actual es: Maria
El primer apellido del nodo actual es: Saez
El segundo apellido del nodo actual es: Morales
El numero de la persona actual es: +56875358225
El nombre completo de la persona es: XavierMariaSaezMorales
La altura del nodo actual es: 0

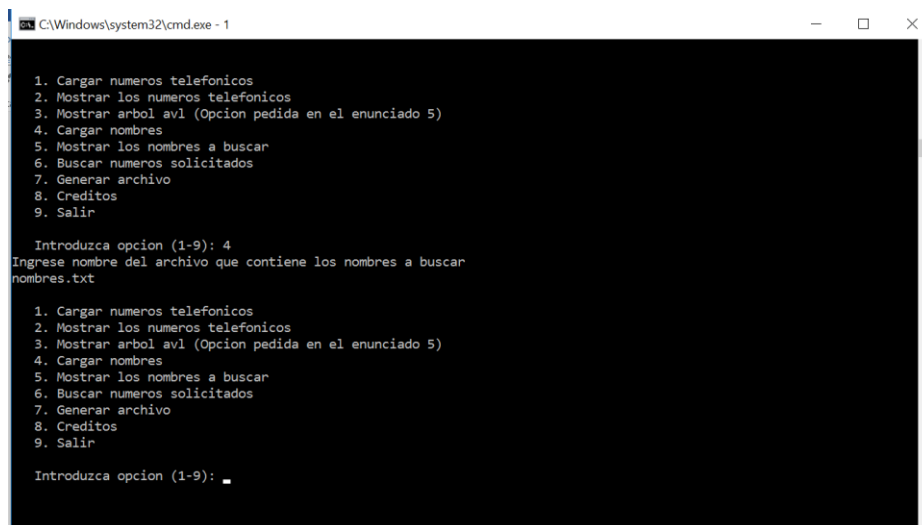
1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9):

```

Figura 22: Mostrando el laberinto por pantalla.

- Luego debemos realizar los pasos 4 para cargar los nombres a los cuales se les desea realizar la búsqueda de su número telefónico, que se puede apreciar en la imagen 23:



```

C:\Windows\system32\cmd.exe - 1

1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9): 4
Ingrese nombre del archivo que contiene los nombres a buscar
nombres.txt

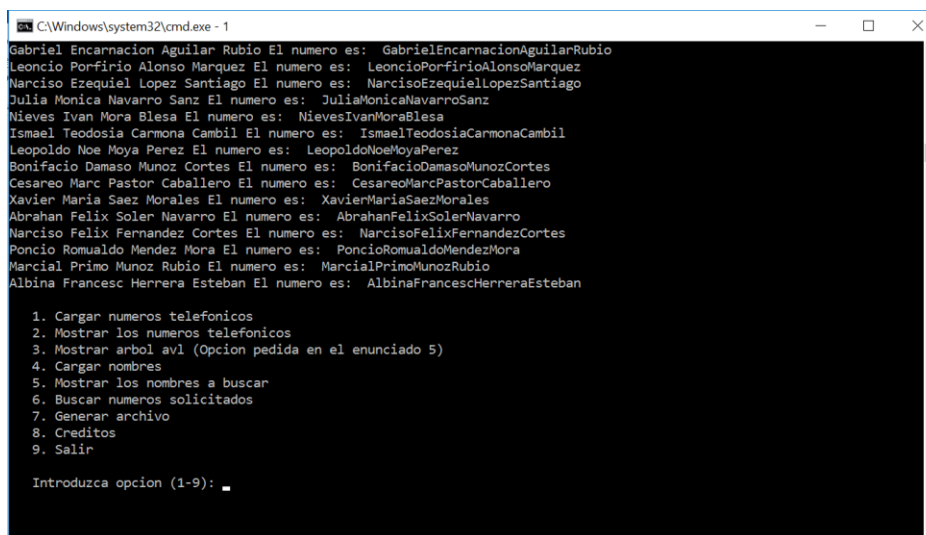
1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9): _

```

Figura 23: *Buscando camino mínimo hacia la llave.*

- Luego de eso debemos volver a ingresar una opción, en este caso ocuparemos la opción número 5 que nos muestra los nombres leídos a los cuales se desea buscar su número telefónico, se debe hacer como se muestra en la Figura 24:



```

C:\Windows\system32\cmd.exe - 1

Gabriel Encarnacion Aguilar Rubio El numero es: GabrielEncarnacionAguilarRubio
Leoncio Porfirio Alonso Marquez El numero es: LeoncioPorfirioAlonsoMarquez
Narciso Ezequiel Lopez Santiago El numero es: NarcisoEzequielLopezSantiago
Julia Monica Navarro Sanz El numero es: JuliaMonicaNavarroSanz
Nieves Ivan Mora Blesa El numero es: NievesIvanMoraBlesa
Ismael Teodosia Carmona Cambil El numero es: IsmaelTeodosiaCarmonaCambil
Leopoldo Noe Moya Perez El numero es: LeopoldoNoeMoyaPerez
Bonifacio Damaso Munoz Cortes El numero es: BonifacioDamasoMunozCortes
Cesareo Marc Pastor Caballero El numero es: CesareoMarcPastorCaballero
Xavier Maria Saez Morales El numero es: XavierMariaSaezMoraes
Abraham Felix Soler Navarro El numero es: AbrahamFelixSolerNavarro
Narciso Felix Fernandez Cortes El numero es: NarcisoFelixFernandezCortes
Poncio Romualdo Mendez Mora El numero es: PoncioRomualdoMendezMora
Marcial Primo Munoz Rubio El numero es: MarcialPrimoMunozRubio
Albina Francesc Herrera Esteban El numero es: AlbinaFrancescHerreraEsteban

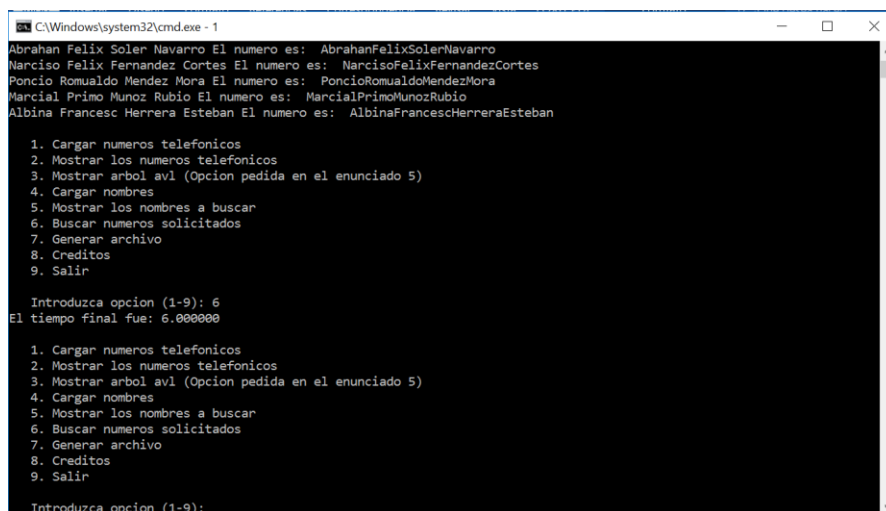
1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9): _

```

Figura 24: *Mostrando los nombres a realizarle la búsqueda.*

- Luego debemos realizar la opción número 6, la cual nos hace la búsqueda de cada uno de los números de contactos de los nombres ingresados para realizar dicha búsqueda, como se puede apreciar en la imagen 25:



```

C:\Windows\system32\cmd.exe - 1
Abrahan Felix Soler Navarro El numero es: AbrahanFelixSolerNavarro
Narciso Felix Fernandez Cortes El numero es: NarcisoFelixFernandezCortes
Poncio Romualdo Mendez Mora El numero es: PoncioRomualdoMendezMora
Marcial Primo Munoz Rubio El numero es: MarcialPrimoMunozRubio
Albina Francesc Herrera Esteban El numero es: AlbinaFrancescHerreraEsteban

1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9): 6
El tiempo final fue: 6.000000

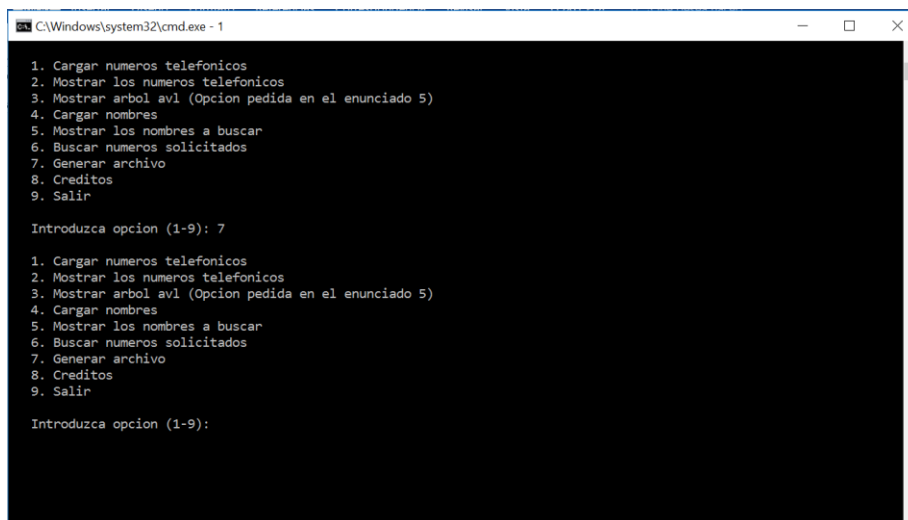
1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9):

```

Figura 25: Realizar la búsqueda de los números telefónicos, además del tiempo de ejecución de software.

- Luego debemos presionar la opción número 7, la cual se encarga de generar el archivo con cada uno de los nombres y sus respectivos números telefónicos encontrados, como se aprecia en la imagen 26:



```

C:\Windows\system32\cmd.exe - 1

1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9): 7

1. Cargar numeros telefonicos
2. Mostrar los numeros telefonicos
3. Mostrar arbol avl (Opcion pedida en el enunciado 5)
4. Cargar nombres
5. Mostrar los nombres a buscar
6. Buscar numeros solicitados
7. Generar archivo
8. Creditos
9. Salir

Introduzca opcion (1-9):

```

Figura 26: Generando el archivo de salida: “TelefonosEncontrados.out”

- Finalmente, para salir del software se selecciona la opción número 9, como muestra la Figura 27:

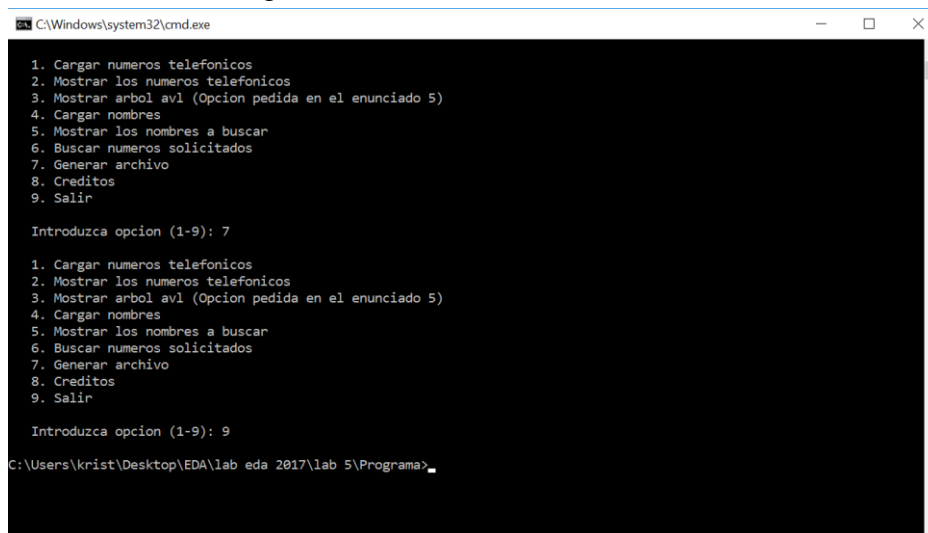


Figura 27: Cerrando el software.

- Luego el software se cierra y si nos dirigimos a la carpeta donde tenemos almacenado el programa no aparece el archivo con la salida, como se puede observar en la Figura 28:

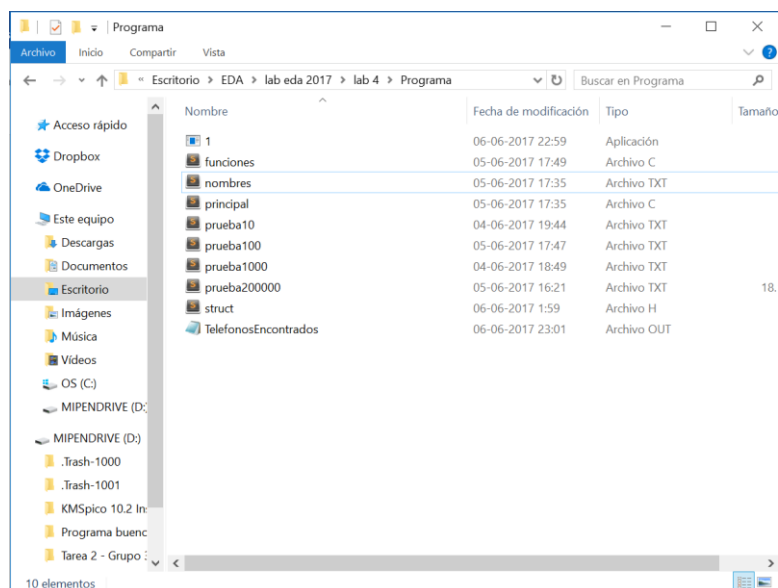


Figura 28: Carpeta con el archivo de “TelefonosEncontrados.out” del software

- Luego abrimos el archivo para verificar la solución final del “Laberinto” que entrega el software, como muestra la Figura 29:

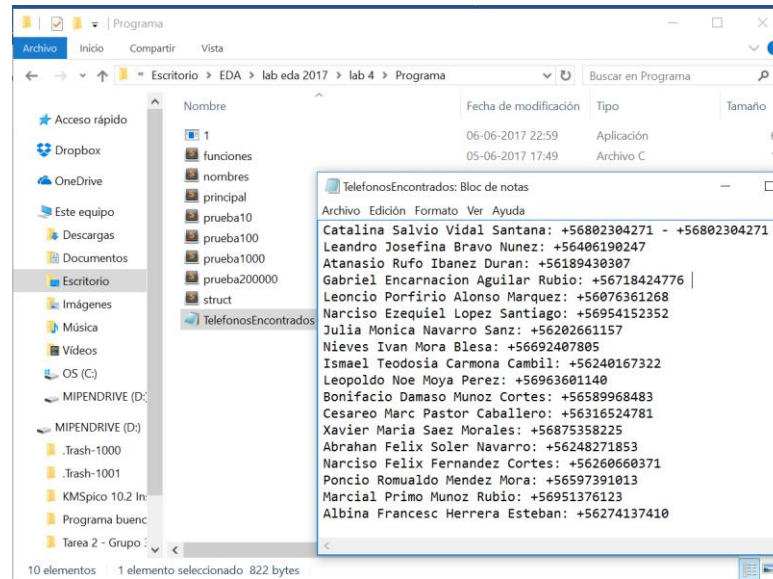


Figura 29: Salida final del software.

5.4 EXITOS Y POSIBLES ERRORES

En ejecución la aplicación paso todas las pruebas que se le realizaron, tener en cuenta que fue sometida a distintas pruebas, con diferente cantidad de personas en el archivo de lectura y siempre teniendo un resultado exitoso.

Algunos de los errores posibles que puede hacer que la aplicación son los siguientes:

- Cuando el archivo de entrada contiene una escritura distinta a la que se entrega en el enunciado del laboratorio 5.
- Cuando los nombres de las personas que están en el archivo, contienen tildes o nombres que contenga la letra “Ñ”, produce que el software no cumpla con todos los objetivos requeridos por la organización ADEsetnaduyA.
- Cuando el usuario modifica alguna línea del código planteado.
- Cuando las entradas no son las que el software espera.
- Cuando el disco se queda sin memoria.