



**UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA**

Estructura de Datos y Análisis de Algoritmos

**Laboratorio N°2
“Potencia de Matrices Dispersas”**

Preparado por: Jorge Sandoval M

Santiago - Chile
1-2019

TABLA DE CONTENIDOS

Tabla de Contenidos	2
Índice de Figuras	3
Índice de Tablas	3
CAPÍTULO 1. Introducción	4
CAPÍTULO 2. Contenidos del Informe	5
2.1 Descripción de la solución	5
2.1.1 Marco teórico	5
VARIABLES	5
PUNTEROS	5
MEMORIA DINÁMICA	5
ARREGLOS	5
ESTRUCTURAS	6
TAMAÑO DE UNA VARIABLE (BYTES)	6
MATRIZ DISPERSA	6
2.1.2 Herramientas y técnicas	6
1. Librerías	6
2. Listas Enlazadas Simples	7
2.1.3 Algoritmos y estructuras de datos	7
Lista Enlazada:	7
II. Algoritmos utilizados	7
2.2 Análisis de los resultados	8
2.3 Conclusión	9
2.4.1 Referenciar páginas Web	9
2.4.2 Referenciar Documentos en línea	9

CAPÍTULO 3. Manual de usuario	10
3.1 Introducción	10
3.2 Cómo compilar y ejecutar	10
3.3 Funcionalidades del programa	11
3.4 Posibles Errores	12

ÍNDICE DE FIGURAS

Figura 1: Tabla de tamaño de datos.	6
Figura 2: Entrada archivo .in	11
Figura 3: Cmd entorno Windows.	11
Figura 4: Consola entorno Linux.	12
Figura 5: Resultado.	13
Figura 6: Formato entrada.	14

ÍNDICE DE TABLAS

Tabla 1: $T(n)$ y Orden	8
-------------------------	---

CAPÍTULO 1. INTRODUCCIÓN

El curso de Análisis de Algoritmos y Estructuras de Datos de la Universidad de Santiago de Chile comienza a entregar herramientas para los alumnos que desean crear y optimizar algoritmos, una de estas herramientas son los arreglos y estructuras que permiten organizar datos estableciendo un orden para facilitar el trabajo con la información almacenada y generar soluciones a distintos tipos de problemas que se puedan presentar.

Para abordar el problema propuesto llamado “Potencia de Matrices Dispersas” se utilizará el lenguaje de programación C cuya principal característica es su velocidad de procesamiento y el control que puede llegar a otorgar al programador sobre la memoria que utiliza el programa. Además, se trabajará bajo el estándar ANSI C lo que permitirá que el programa sea compatible a casi todos los compiladores y entornos.

El objetivo principal de este proyecto consiste en desarrollar un programa que calcule la potencia n -ésima de una matriz cuadrada dispersa de MXM que será entregada en un fichero de texto plano y que luego muestre el resultado por pantalla. Como objetivo particular se requiere indagar e implementar los nuevos conocimientos en estructuras de datos para optimizar el uso de la memoria y los tiempos de ejecución en C.

También se espera que el programa pueda ser compilado y ejecutado desde cualquier distribución de Linux o Windows, utilizando el conjunto de compiladores GCC y como término del laboratorio, lograr calcular la complejidad y el orden de los algoritmos más relevantes.

El presente informe documentará los pasos seguidos para conseguir la solución, los conocimientos previos antes de la implementación, la descripción de la solución y todo el material teórico que fundamenta el trabajo realizado. Finalmente se detallará un manual de usuario para interactuar con el programa de forma correcta.

CAPÍTULO 2. CONTENIDOS DEL INFORME

A continuación, se explicarán las herramientas, técnicas y algoritmos empleados para la elaboración de una solución además del material investigado el cual sirvió como base para resolver el problema.

2.1 DESCRIPCIÓN DE LA SOLUCIÓN

En esta sección se describe el trabajo realizado y su fundamento teórico, señalando las herramientas, técnicas y supuestos incorporados, haciendo énfasis en los algoritmos y estructuras de datos utilizadas.

2.1.1 Marco teórico

I. VARIABLES

En programación, las variables son espacios reservados en la memoria que, como su nombre indica, pueden cambiar de contenido a lo largo de la ejecución de un programa. Una variable corresponde a un área reservada en la memoria principal del ordenador.

II. PUNTEROS

Un puntero es un dato que contiene una dirección de memoria.

Cuando se declara un puntero se reserva memoria para albergar una dirección de memoria, pero no para almacenar el dato al que apunta el puntero. El espacio de memoria reservado para almacenar un puntero es el mismo independientemente del tipo de dato al que apunte.

III. MEMORIA DINÁMICA

Es un espacio lógico para guardar memoria durante la ejecución, es decir, la asignación de memoria puede hacerse durante la implementación y podrá modificarse según el programa necesite

IV. ARREGLOS

Los arreglos son estructuras de datos consistentes en un conjunto de datos del mismo tipo. Los arreglos tienen un tamaño que es la cantidad de objetos del mismo tipo que pueden almacenar. Los arreglos son entidades estáticas debido a que se declaran de un cierto tamaño y conservan este todo a lo largo de la ejecución del programa en el cual fue declarado.

V. ESTRUCTURAS

Las estructuras son colecciones de variables relacionadas bajo un nombre. Las estructuras pueden contener variables de muchos tipos diferentes de datos (a diferencia de los arreglos que contienen únicamente elementos de un mismo tipo de datos).

VI. TAMAÑO DE UNA VARIABLE (BYTES)

Unidad de información almacenada según tipo de dato:

Tipo	Tamaño (bytes)	Rango
int (entero)	2	-32,768 a 32,767
float (flotante)	4	3.4 E-38 a 3.4 E+38
double (flotante de doble precisión)	8	1.7 E-308 a 1.7 E+308
char (carácter)	1	-128 a 127
void	0	sin valor

Figura 1: Tabla de tamaño de datos.

VII. MATRIZ DISPERSA

Es aquella que está compuesta por muchos elementos de valor = 0 de tal forma que los que son distintos de 0 se encuentran muy dispersos en la matriz y sin relación entre sí.

2.1.2 Herramientas y técnicas

Se decide utilizar, como técnicas para la resolución del problema, el recurso de punteros y estructuras que ofrece C. Las razones de su uso se detallan a continuación junto con aquellas librerías empleadas distintas a la de entrada-salida (stdio.h) manejo de memoria (stdlib.h) y operaciones matemáticas especiales (math.h).

1. Librerías

Dentro del programa es fundamental el uso de las librerías, utilizando las esenciales stdio.h encargada de las macro definiciones, constantes, y las declaraciones de funciones y tipos usados para la entrada de varios estándar y operaciones de salida, como también stdlib.h que contiene los prototipos de funciones de C para la gestión de memoria dinámica, generación de números pseudo-aleatorios, control de procesos y demás.

Otras librerías que fueron manejadas son `math.h` siendo fundamental para realizar cálculos matemáticos como las potencias.

2. Listas Enlazadas Simples

El uso de listas enlazadas facilitará el trabajo debido a que sin importar el tamaño de datos que recolecta el programa, esta podrá almacenarlos sin limitación a medida que avanza en ejecución, por lo que esta herramienta resulta fundamental para que la solución se ajuste a cualquier entrada y utilice la memoria precisa para solucionar el problema.

Teóricamente, Es una lista enlazada de nodos, donde cada nodo tiene un único campo de enlace. Una variable de referencia contiene una referencia al primer nodo, cada nodo (excepto el último) enlaza con el nodo siguiente, y el enlace del último nodo contiene NULL para indicar el final de la lista.

2.1.3 Algoritmos y estructuras de datos

En esta subsección o subcapítulo, se consideran y explican brevemente qué algoritmos y estructuras de datos fueron utilizados para el desarrollo del laboratorio y su funcionamiento, desglosando las funciones más importantes, detallando sus entradas, salidas y operación.

I. Lista Enlazada:

Se crea una sola lista enlazada que irá “capturando” nuevos datos al recorrer el archivo de entrada, guardando su posición fila/columna(entero) y evidentemente el dato en formato double que contiene la matriz.

II. Algoritmos utilizados

II.I Captura de datos

Al iniciar el programa, este comprobará que el archivo de entrada sea válido, con el formato requerido, siendo este un archivo de texto plano que contiene en primer lugar el tamaño N de la matriz cuadrada y la potencia a la que se elevará, luego los datos de la matriz. El programa recorrerá el archivo y guardará todos aquellos datos que sean distintos de cero, obteniendo también su posición fila/columna en la matriz original.

II.II Cálculo de la potencia

El algoritmo empleado para el cálculo de la potencia de la matriz simplemente recorre la lista enlazada y opera cada dato con la función POW (math.h) modificando el número obtenido en la lista enlazada y continuando con el siguiente.

II.III Muestra de matriz resultante

El algoritmo que imprime la matriz resultante utiliza los datos previos que se tienen de la matriz original, entonces el programa verificará por filas y columnas si existe algún dato almacenado en la lista enlazada en el caso de que exista imprimirá el dato y avanzará al siguiente nodo para verificar, en caso de que no encuentre una coincidencia imprimirá un cero, se mantendrá así hasta construir la matriz NxN.

2.2 ANÁLISIS DE LOS RESULTADOS

Funciones	Tiempo de Ejecución	Orden de complejidad
Añadir()	$T(n) = C + C + C = 3$	1
LeerMatriz()	$T(n) = 2Cn^2 + Cn + C$	n^2
powMatriz()	$T(n) = C + nC = Cn$	n
MostrarMatriz()	$T(n) = C + n^2C = Cn^2$	n^2
MostrarListaEnlazada()	$T(n) = nC + C = Cn$	n

Tabla 1. T(n) y orden.

El programa trabaja los datos de forma “lineal”, sólo utiliza el recurso for para recorrer la matriz inicial e imprimir la resultante, por lo que resulta un algoritmo muy eficiente para trabajar con matrices de grandes dimensiones, debido a que es selectivo y no gasta recursos en datos donde el resultado es conocido (0).

2.3 CONCLUSIÓN

Los objetivos esperados se cumplen de manera satisfactoria, el programa logra resolver potencia de matrices de grandes dimensiones con la condicional de que estas deben ser dispersas. El algoritmo implementado utiliza una estructura de datos organizada y eficiente que permite reducir el tiempo de ejecución y optimiza el uso de memoria del programa. Adicionalmente a los desafíos iniciales se podría implementar el caso de los 1 en las matrices, ya que al elevarlos a cualquier exponente, seguirán siendo 1 y podrían optimizar aún más el uso de la memoria del programa.

2.4 REFERENCIAS

2.4.1 Referenciar páginas Web

Matrices dispersas y métodos de resolución para programadores, Algebra Lineal: (https://www.fing.edu.uy/inco/cursos/comp1/teorico/2011/clase_23_2011_matDispersas.pdf)

Stdlib.h (2018,18) de junio. Wikipedia, la enciclopedia libre. Fecha consulta: abril 12, 2019 from <https://es.wikipedia.org/wiki/Stdlib.h>

Stdio.h (2010,25) de agosto. Blog Rikrdoavella. Fecha consulta: abril 12, 2019 from <http://rikrdoavella.over-blog.es/article-stdio-h-y-sus-funciones-55958648.html>

2.4.2 Referenciar Documentos en línea

Berzal, Fernando. (2004) ACM Senior Member & IEEE Computer Society Member (14/04/2019). <https://elvex.ugr.es/decsai/c/apuntes/punteros.pdf>.

CAPÍTULO 3. MANUAL DE USUARIO

3.1 INTRODUCCIÓN

El cálculo de una potencia de matriz puede resultar muy engorroso cuando se trata de matrices de grandes dimensiones, es por esto que se ha solicitado implementar un algoritmo para facilitar los cálculos y optimizar el aprendizaje.

El programa funciona de forma muy sencilla, recibiendo un archivo .in con la información necesaria para obtener la potencia de una matriz dispersa cuadrada.

3.2 CÓMO COMPILAR Y EJECUTAR

Antes de compilar el usuario debe corroborar que el archivo de entrada se encuentre en la misma carpeta donde compiló, y en el código, que ha ingresado el nombre correcto del archivo de entrada, para eso se solicita ir al código y en la línea 150 poner el nombre del archivo “.in” válido como se muestra a continuación:

```
149 //*****
150 fichero = fopen("MatrizPrueba.in","r");
151 //*****
```

Figura 2: Entrada archivo .in .

Verificado esto se procede según Sistema Operativo:

-Para Windows

Abra la consola de Windows escribiendo en el buscador “cmd” y diríjase a la carpeta donde se encuentra el programa y los archivos solicitados.

Una vez abierta la consola, para poder compilar el programa tiene que escribir la siguiente línea:

```
C:\> Símbolo del sistema
Microsoft Windows [Versión 10.0.17134.590]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Jorge Sandoval>cd Desktop

C:\Users\Jorge Sandoval\Desktop>gcc Proyecto2.c -o salida

C:\Users\Jorge Sandoval\Desktop>salida
```

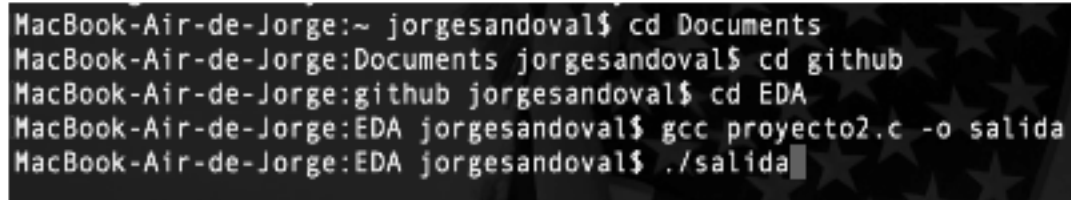
Figura 3: Cmd entorno Windows.

Para ejecutar escriba “salida” y presione enter.

-Para Linux

Abra el terminal de Linux escribiendo en el buscador Terminal y diríjase a la carpeta donde se encuentra el programa y los archivos solicitados.

Una vez abierta la consola, para poder compilar el programa tiene que escribir ‘gcc proyecto2.c -o salida’ como se muestra a continuación:

A screenshot of a Linux terminal window with a dark background and light-colored text. The terminal shows a series of commands and their outputs. The prompt is 'MacBook-Air-de-Jorge:~ jorgesandoval\$'. The first command is 'cd Documents', the second is 'cd github', the third is 'cd EDA', the fourth is 'gcc proyecto2.c -o salida', and the fifth is './salida'. The cursor is at the end of the last command.

```
MacBook-Air-de-Jorge:~ jorgesandoval$ cd Documents
MacBook-Air-de-Jorge:Documents jorgesandoval$ cd github
MacBook-Air-de-Jorge:github jorgesandoval$ cd EDA
MacBook-Air-de-Jorge:EDA jorgesandoval$ gcc proyecto2.c -o salida
MacBook-Air-de-Jorge:EDA jorgesandoval$ ./salida
```

Figura 4: Consola entorno Linux.

Para ejecutar escriba “./salida” y presione enter.

Resultado tras la ejecución esperado en ambos:

Tras la ejecución del programa y si todo ha funcionado correctamente, el usuario debe obtener una interfaz de este tipo en su consola:

```
MacBook-Air-de-Jorge:EDA jorgesandoval$ ./salida
La matriz dispersa otorgada es de [20,20].
El exponente otorgado ha sido: >3<.
 0  0  0 -8  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  8
 0  0  0 -8  0  0  0  0  0  0  8  0  0  0  8  0  0  0  0  0
 0  0  0 -1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0 -1  0  0  0  0  0  0  0  0  0  0  0  8  0  0  0  0
 0  0  0 -8  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0 -1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 1  0  0 -8  0  8  0  0  0  0  0  0  0  0  0  1  0  0  0  0
 0  0  0 -1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0 -1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0 -1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  8
 0  0  0 -8  0  1  0  0  0  0  0  8  0  0  0  0  0  0  0  0
 0  0  0 -8  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 8  0  0 -1  0  0  0  0  0  0  0  0  8  0  0  0  0  0  0  0
 0  0  0 -1  0  8  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0 -1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  8 -1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0 -8  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 1  0  0 -8  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0 -8  0  0  0  8  0  0  0  0  0  8  0  0  0  0  0  0
 0  0  8 -8  0  0  0  0  0  8  0  0  0  0  0  0  0  0  0  1
El tamaño de la matriz dispersa es: 2080 bytes.
El tamaño de la matriz completa habría sido: 6400 bytes.
```

Figura 5: Resultado

3.3 FUNCIONALIDADES DEL PROGRAMA

El presente programa puede resolver cualquier matriz dispersa que se entregue en formato .in con la siguiente estructura: primer elemento tamaño N de la matriz, segundo elemento la potencia que se desea aplicar y luego la matriz.

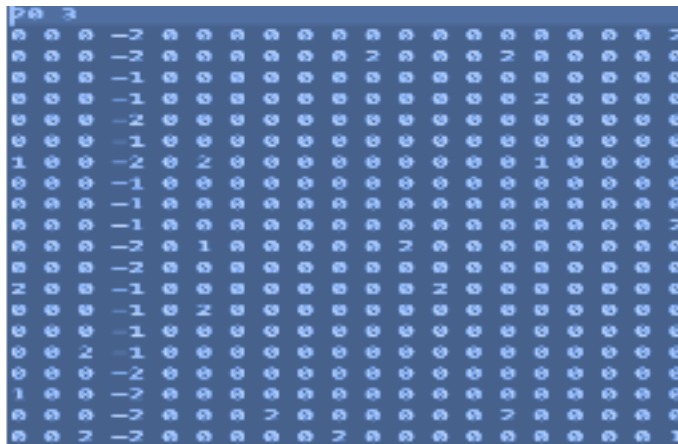


Figura 6: Formato entrada.

Y retornará el resultado de la potencia de la matriz por consola además de los datos de memoria utilizados.

3.4 POSIBLES ERRORES

En el caso de que entregue una matriz demasiado grande el programa podrá mostrar dificultades para calcular el tamaño de los BYTES que utiliza la matriz dispersa y los que utilizaría en el caso de que no lo fuera, esto ocurre ya que el cálculo es una variable “Largo Entero” que permite almacenar números en el rango [-2147483648 2147483647].