**Tarea 1 - Análisis de Algoritmos y Estructura de Datos**

Jovanni Schneider Pizarro

Universidad de Santiago de Chile

1-2022

**I. INTRODUCCIÓN**

Se tiene el siguiente problema de ordenamiento: Una pizzería requiere ordenar sus masas de pizza de distintos diámetros de mayor a menor (desde la base) realizando volteretas, las cuales consisten en meter la espátula debajo de una masa y voltear todas las que estén sobre esta, invirtiendo el orden de la posición desde la que se hizo la voltereta hasta la cima de la torre.

Por ejemplo si tenemos la función voltereta definida como V(x) donde x es la posición desde la cual se realiza la voltereta, y se tiene una torre formada por los diámetros (3, 5, 2 ,6 ,10) y realizamos V(2), nuestra torre de salida sería

(3, 10, 6, 2, 5).

La pizzería entrega un archivo en el cual en su primera línea contiene un

entero que corresponde a la cantidad de masas presentes en la torre y en la segunda línea contiene los diámetros de las pizzas separados por espacios, teniendo este archivo se debe generar un archivo de salida el cual contenga la secuencia de volteretas representadas por los índices desde donde se hacen las volteretas.

Por ejemplo, si tenemos la siguiente secuencia:

**V (3)− > V (1)− > V (5)− > V (1)**

El contenido del archivo de salida debiese ser una sola línea con lo

siguiente: 3 1 5 1.

Para la solución de este problema

se usa el lenguaje de programación C

**II. SOLUCION PROPUESTA**

Se tiene la lista de diámetros genérica:

(d1 , d2 ­, di , … , dn-1 , dn)

Dentro de este grupo de diámetros se considera una división en 2 subgrupos, de manera que se tiene un subgrupo ordenado y un subgrupo desconocido (comúnmente desordenado). A su vez se considera un elemento separador, el cual pertenece al sub conjunto desordenado y es el primer elemento de este. De modo que se tiene lo siguiente:



Se considerará inicialmente desordenado todo el conjunto de diámetros, de manera que se iniciará con un estado como el siguiente:





De esta manera se comenzará a crear el conjunto ordenado encontrando el mayor numero del conjunto desordenado y llevarlo a la posición del índice separador, de esta manera este elemento ahora estará ordenado y el índice debería avanzar a la siguiente posición, quedando así:





Donde d1 es el mayor número del conjunto completo y en la posición d2 debería llegar luego el segundo elemento mayor del conjunto completo.

Para encontrar el mayor número dentro de un conjunto se tiene el algoritmo [encontrarMayor](#encontrarMayor) de O(n) (ver anexo), y luego de encontrarlo se debe llevar a la posición del índice separador, sin embargo, la voltereta no nos permite llevarlo directamente, lo que hay que hacer es llevarlo a la posición más alta de la torre (n) y luego llevarlo a la posición del separador, para realizar la voltereta se tiene el algoritmo [swap](#swap) de O (n) (ver anexo).

Teniendo estas ideas y definiciones claras nos surgen 3 casos posibles que de no considerarlos podría dar errores en la creación de un algoritmo final:

1. El mayor número se encuentra en la posición del separador
2. El mayor número ya se encuentra en la cima de la torre
3. El mayor numero se encuentra entre la última posición y la del separador.

Para el caso 1 no hay nada que ordenar, por lo que solo se avanzaría el separador.

Para el caso 2 no hay necesidad alguna de llevar el mayor numero a la cima porque este ya se encuentra ahí, por lo que solo se debe llevar a la posición del separador y avanzar este.

Para el caso 3 se debe realizar la secuencia que se planteó anteriormente, llevar el numero mayor a la cima de la torre y luego llevarla a la posición del separador.

Para poder obtener la torre completamente ordenado habría que llevar el separador hasta la posición n, de manera que se tenga el siguiente estado:





El algoritmo [ordenar](#ordenar) O(n2)(ver anexo) es el encargado de realizar estas tareas. El cual va guardando la secuencia de swaps en un arreglo de enteros, el cual se guardará luego en el archivo de salida (las funciones utilizadas para leer el archivo o escribir en el son omitidas ya que dependen del lenguaje de programacioón al igual que su complejidad, por lo que solo se utilizan como escribirArchivo y leerArchivo)

Teniendo todas estas funciones y análisis nuestro algoritmo final que cumpla con los requerimientos de la pizzería ha de ser como el siguiente

Texto

Descripción generada automáticamente

Para su implementación en C este algoritmo cambia un poco para eliminar errores de lectura y por el trabajo con arreglos dinámicos, además de reemplazar los arreglos como tal con struct los cuales contienen los arreglos, se utilizan los siguientes structs para guardar su tamaño y hacer más cómodo su trabajo con el lenguaje:

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

**III. RESULTADOS Y ANALISIS**

El algoritmo planteado apunta a resolver el problema en un O(n2),

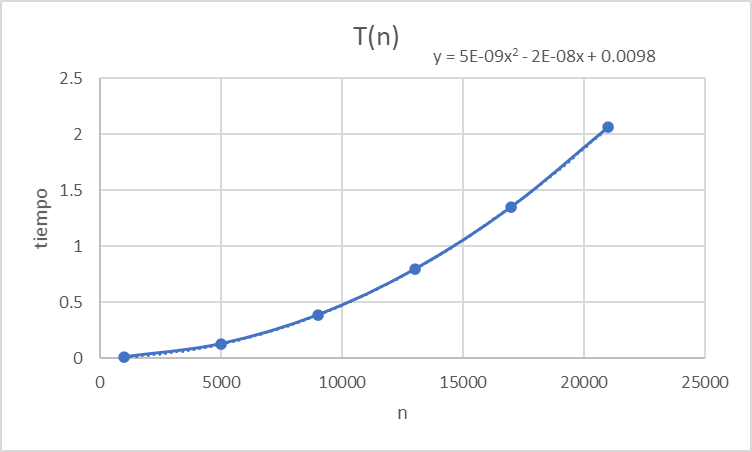
Dado que su función de ordenar recorre toda la lista con un for que da un orden n y dentro aplica funciones de orden n. Se escribe el código en C de manera que nos muestre su tiempo de ejecución de la siguiente manera:



Para medir sus resultados se hizo 4 mediciones con distintos números aleatorios provistos por la página web [pinetools](https://pinetools.com/es/generador-numeros-aleatorios) en distinto orden y con el mismo N, para luego sacar su promedio y este tomarlo como t(n) como se muestra en la siguiente tabla(revisar [anexo](#tabla) para ver tabla completa de tiempos)



Con la tabla anterior podemos realizar su grafico y ver como se comporta con los distintos tamaños de n, teniendo lo siguiente:



Acá se puede observar claramente que la forma de la línea de línea de tendencia es cuadrática, de hecho podemos obtener su ecuación provista por el software Microsoft Excel la cual es

*y = 5\*10-9x2 – 2\*10-8 x + 0.0098*

De esta manera queda comprobado que el tiempo de ejecución del programa es de n2.

**IV. CONCLUSIONES**

El algoritmo planteado resuelve bien la solicitud de la pizzería y entrega la secuencia de masas, sin embargo es bastante costosa la implementación de este en términos de orden, ya que tenemos un orden n2, sin embargo la naturaleza del problema apuntaba a eso, ya que estaba restringida la forma de realizar las volteretas para ordenar.

Quizás la forma de abordar el problema tampoco fue la más adecuada, podría haberse resuelto utilizando técnicas como Búsqueda en Espacios de Estados, sin embargo, se cumplió de manera satisfactoria y funciona bastante bien para la realidad de su uso que sería la pizzería, ya que entrega tiempos insignificantes para cantidades de pizzas absurdas.

**ANEXOS Y MANUAL DE USUARIO**

Para la correcta ejecución del programa es necesario contener en el mismo directorio los archivos “main.c” y “funciones.h”, además de el archivo de entrada el cual su contenido debe seguir el formato:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

De esta manera un directorio valido para la ejecución del programa sería el siguiente:

Interfaz de usuario gráfica, Aplicación, Tabla

Descripción generada automáticamente

Se debe compilar el archivo main.c y ejecutar su resultante dentro del mismo directorio, ingresando el nombre de nuestro archivo de entrada y nuestro futuro archivo de salida, de la siguiente manera:

Una captura de pantalla de un celular

Descripción generada automáticamente con confianza media

Si no ingresamos un archivo de entrada o este no existe el programa nos avisará de aquello, si no ingresamos ningún archivo de salida este simplemente no se creará.

Al ejecutar los comandos anteriores se tiene el siguiente resultado como directorio:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Y el contenido de nuestro archivo de salida vueltas.out es el siguiente:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Referencias:**

Función encontrarMayor()

Texto

Descripción generada automáticamente



Función swap

Texto

Descripción generada automáticamente



Función ordenar

Texto

Descripción generada automáticamente

Tabla

