**Introduccion**

Dado un árbol de directorios que obtenemos a partir de un directorio o archivo raíz, recorremos dicho árbol recolectando los paths obtenidos desde la raíz hasta cada una de las hojas. Una vez obtenido lo antes mencionado procedemos a revisar todos los caminos, que están formados por los nombres de los directorios que lo componen concatenado con el nombre del archivo que representa la hoja (si la opción fue habilitada), y encontramos, si existen, todos los palíndromos ubicados dentro de ellos.

El informe estará dividido en 5 secciones en las cuales se explicará cómo compilar y correr el programa, la estructura general de la solución, la estrategia utilizada para la creación de procesos hijos, cómo comunicamos y sincronizamos los procesos utilizados y cómo manejamos la cadena de caracteres que almacena los caminos del árbol.

**Sección 1: Cómo Compilar y Correr el Programa**

**Sección 2: Estructura General de la Solución**

Para el problema expuesto en la introducción creamos un grafo no dirigido implícito definido de la siguiente manera: un nodos es cada uno de los archivos o directorios que componen el árbol, y una relación es todo par (x,y) tal que x es un directorio que apunta a y, o viceversa. Una vez construido el grafo procedemos a implementar el algoritmo DFS sobre él, partiendo desde la raíz, con la variación de que no utilizamos el atributo de visitados de los nodos, ya que por la estructura de árbol dos caminos distintos no pueden intersectarse, y a medida que avanzamos en el grafo concatenamos el nombre del nodo actual en un arreglo de caracteres que contiene el camino que se ha recorrido para cada nodo.

Una vez alcanzada una hoja, o llegado al límite de altura introducido como flag, no se sigue extendiendo el camino actual y se procede a escribir en un archivo el contenido en la cadena de caracteres acumulado hasta el nodo en que nos encontramos, incluyendo este si no es un archivo o, si es un archivo, si la opción fue habilitada. Una vez se hayan visitado todas las hojas del árbol, o todos los nodos cuya altura es igual a la altura máxima, se procede a leer el archivo, línea por línea y, por cada una de ellas, se hayan todos los palíndromos que se encuentren en su interior.

**Sección 3: Creación de Procesos Hijo**

Para realizar el recorrido del árbol explicado en la sección anterior decidimos emplear una función de creación recursiva de procesos llamada “navegar\_directorio”, para la cual uno de los parámetros ingresados es la ruta de un directorio. Dicha función crea un proceso hijo del proceso que la llama, la cual abre el directorio y, para cada uno de los elementos encontrados en él, revisa si es un directorio o un archivo. Si el elemento que se encuentra en el directorio pasado como parámetro es a su vez un directorio procedemos llamar de nuevo a la función, esta vez con el directorio actual como parámetro., con lo cual seguiremos creando procesos que explorarán el directorio.

La función “navegar\_directorio” posee los siguientes parámetros:

* Name: Ruta para llegar al directorio.
* Namestring: Ruta para llegar al directorio eliminando los /.
* Maxlength: Altura límite pasada por parámetro.
* Height: Altura actual del directorio en el árbol.
* Includefiles: Entero que especifica si el nombre de archivos se concatena al arreglo de caracteres.

**Sección 4: Comunicación y Sincronización Entre Procesos**

La comunicación entre proceso se lleva a cabo, como mencionamos con anterioridad, a través de un archivo. Éste sirve como lugar común entre los procesos para que en él puedan escribir los caminos obtenidos, facilitando así la búsqueda de los palíndromos.

Para la sincronización de procesos nuestra estrategia fue aprovechar la estructura del DFS, la cual consiste en