Jesus Sanchez-Rivera

CEN4025

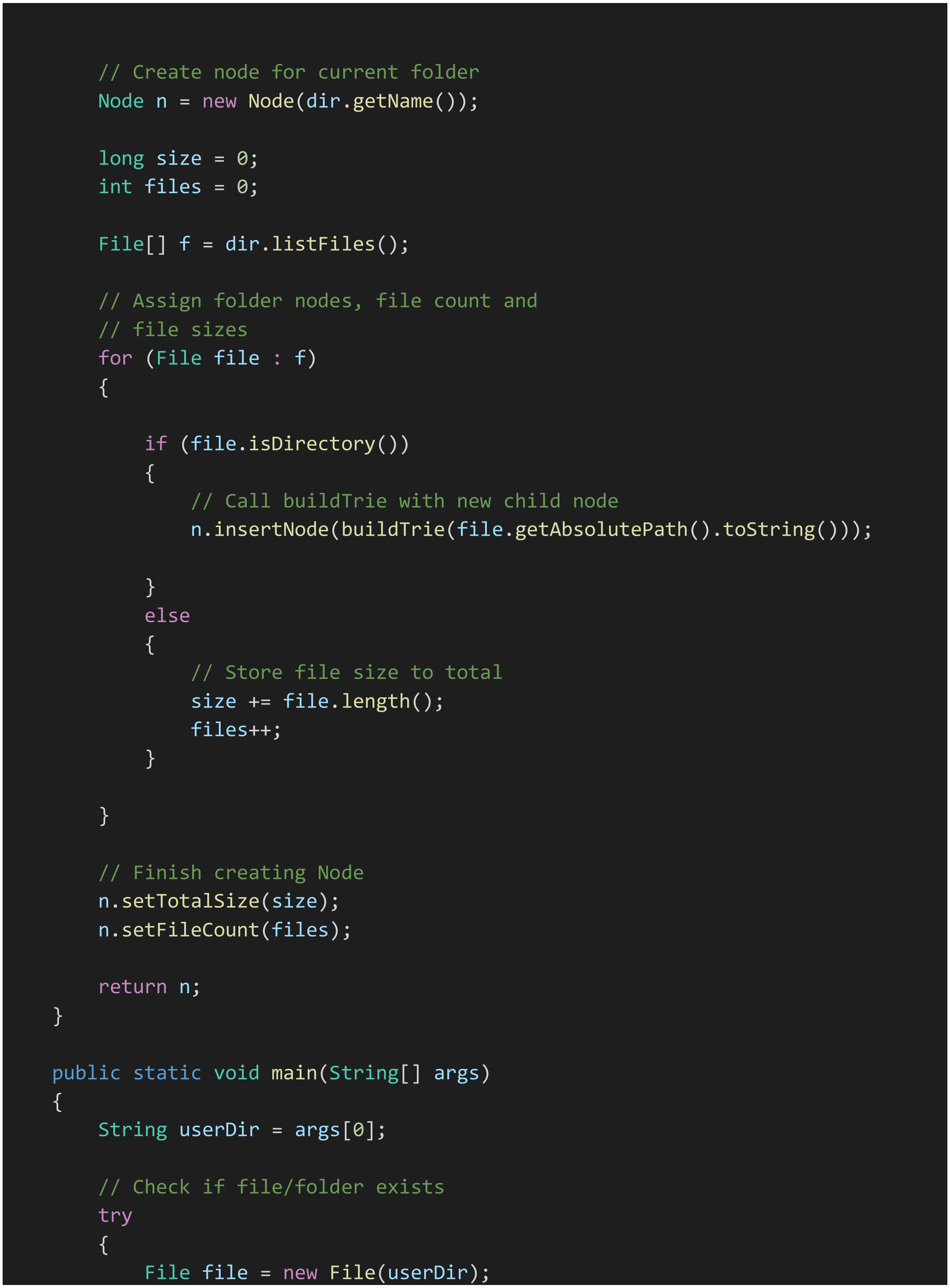
Prof Walauskis

January 16, 2023

Source code:

DirectoryTraversal.java

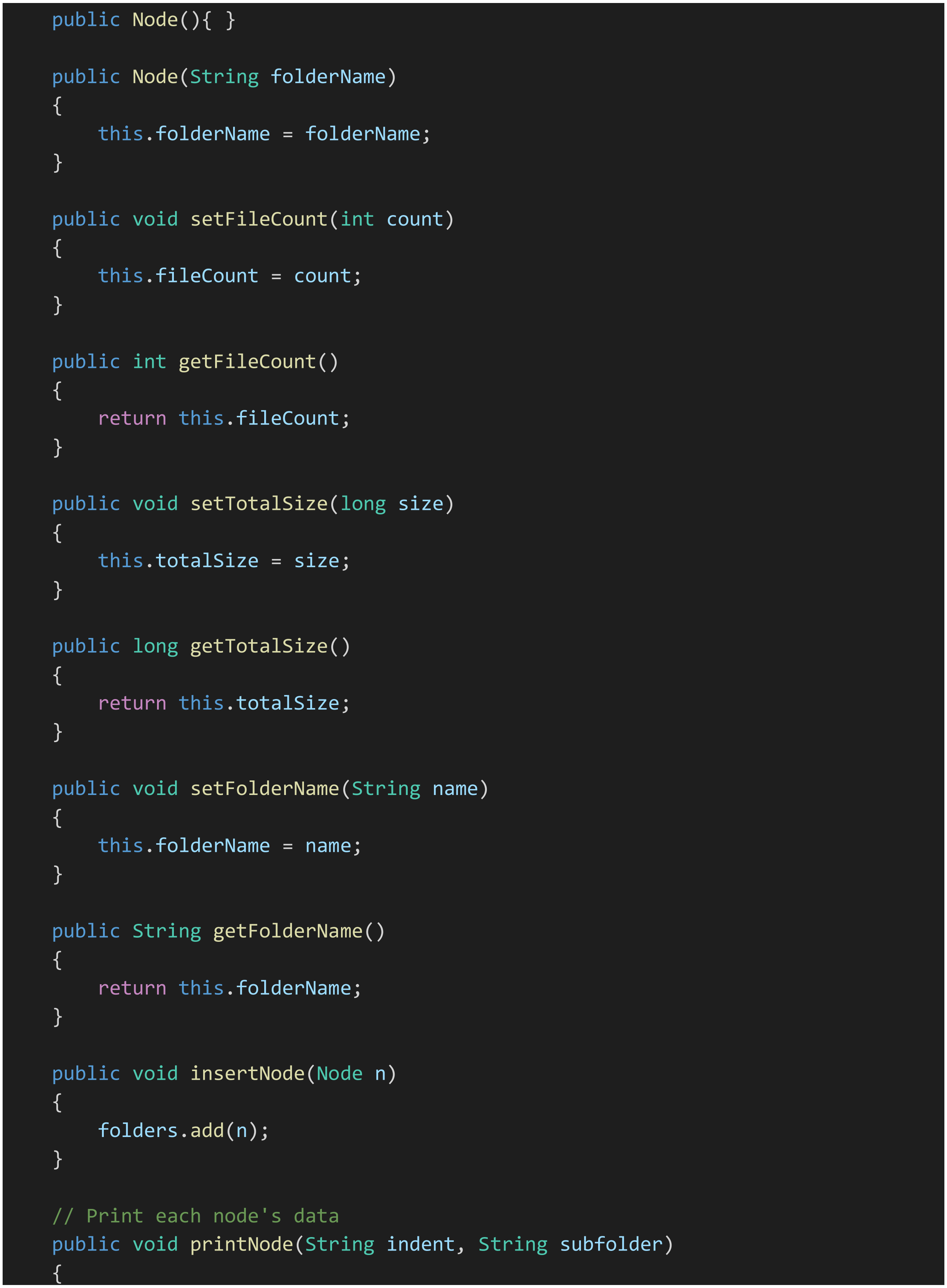
```java
// Jesus Sanchez-Rivera
// CEN 4025
// Prof Walauskis
// January 16, 2023

import java.io.File;
import java.io.IOException;

public class DirectoryTraversal
{
    // Print the data structure recursively
    public static void printTrie(Node head, StringBuilder indent, StringBuilder 
subfolder)
    {
        // Call each node's print function
        head.printNode(indent.toString(), subfolder.toString());

        // Next level below
        for (Node n : head.folders)
        {
            // Adjust indent level each time printTrie
            // is called
            printTrie(n, indent.append("|_"), subfolder.append(" |"));
            indent.setLength(indent.length() - 2);
            subfolder.setLength(subfolder.length() - 2);
        }
    }

    public static Node buildTrie(String filepath) throws IOException
    {

        // Current directory content
        File dir = new File(filepath);
```



```java
        // Create node for current folder
        Node n = new Node(dir.getName());

        long size = 0;
        int files = 0;

        File[] f = dir.listFiles();

        // Assign folder nodes, file count and
        // file sizes
        for (File file : f)
        {

            if (file.isDirectory())
            {
                // Call buildTrie with new child node
                n.insertNode(buildTrie(file.getAbsolutePath().toString()));

            }
            else
            {
                // Store file size to total
                size += file.length();
                files++;
            }

        }

        // Finish creating Node
        n.setTotalSize(size);
        n.setFileCount(files);

        return n;
    }

    public static void main(String[] args)
    {
        String userDir = args[0];

        // Check if file/folder exists
        try
        {
            File file = new File(userDir);
            if (!(file.exists()))
            {
                System.out.println("Directory/File does not exist.");
            }
            else
            {
                // Build first node as the head of the
                // data structure
                Node head = buildTrie(userDir);

                // Print the data structure with characters
                // to help indent the output
                StringBuilder indent = new StringBuilder("");
                StringBuilder subfolder = new StringBuilder("|");
                printTrie(head, indent, subfolder);
            }
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

Node.java

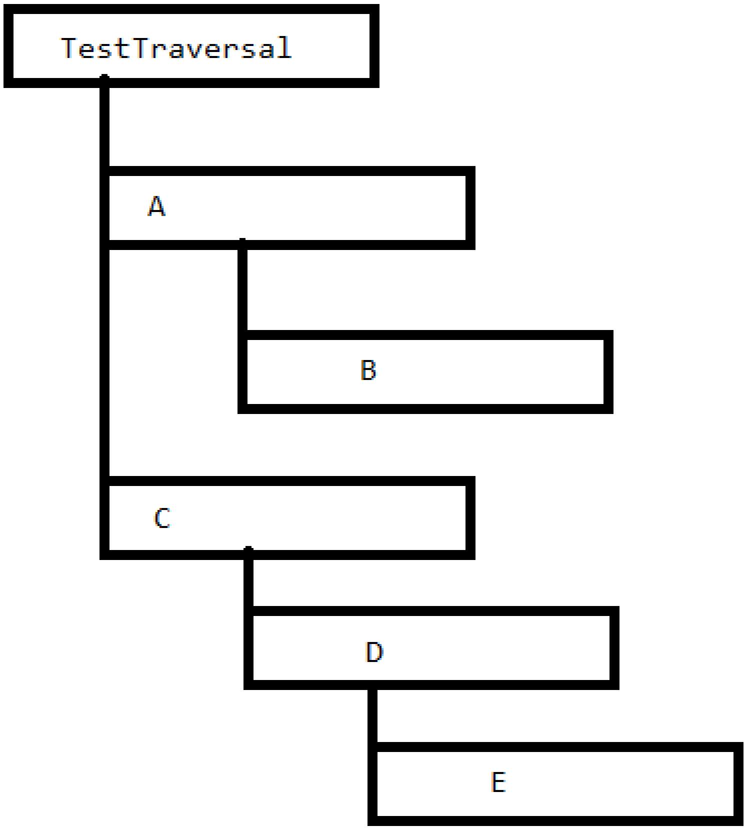
```
// Jesus Sanchez-Rivera
// CEN 4025
// Prof Walauskis
// January 16, 2023

import java.util.ArrayList;

class Node
{
    int fileCount;
    long totalSize;
    String folderName;
    ArrayList<Node> folders = new ArrayList<>();

    // Default constructor
```

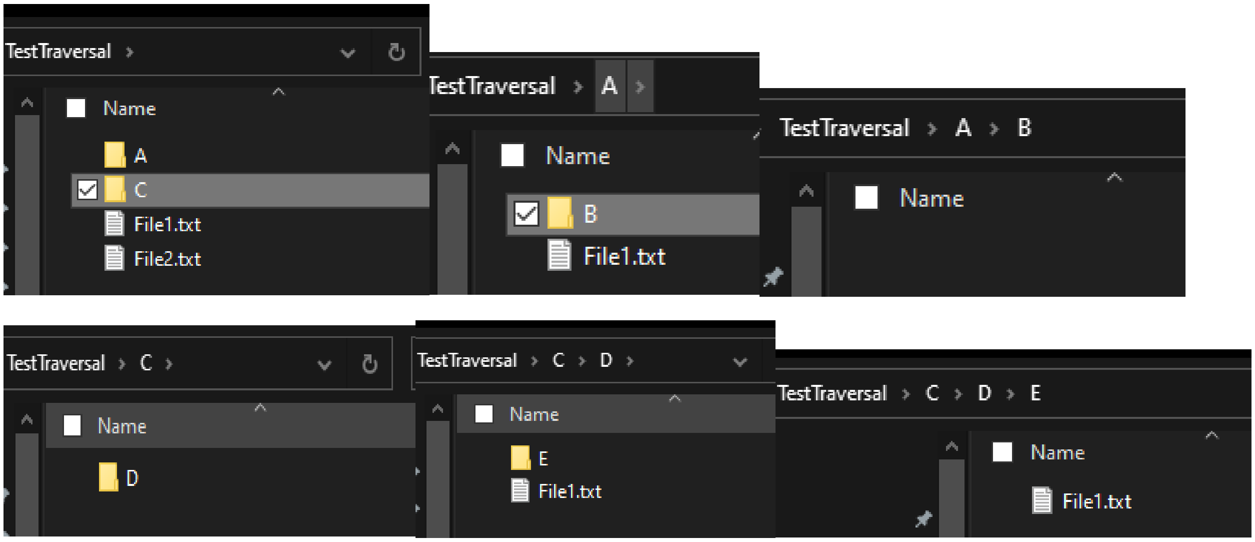


```java
    public Node(){ }

    public Node(String folderName)
    {
        this.folderName = folderName;
    }

    public void setFileCount(int count)
    {
        this.fileCount = count;
    }

    public int getFileCount()
    {
        return this.fileCount;
    }

    public void setTotalSize(long size)
    {
        this.totalSize = size;
    }

    public long getTotalSize()
    {
        return this.totalSize;
    }

    public void setFolderName(String name)
    {
        this.folderName = name;
    }

    public String getFolderName()
    {
        return this.folderName;
    }

    public void insertNode(Node n)
    {
        folders.add(n);
    }

    // Print each node's data
    public void printNode(String indent, String subfolder)
    {
        System.out.print(indent + "*Folder*: " + this.folderName +
                        "\n" + indent + "File count: " + this.fileCount +
                        "\n" + indent + "Combined file size: " + this.totalSize +
" kb");

        // Formatting
        if (folders.size() == 0)
        {
            System.out.println();
        }
        else
        {
            System.out.println("\n" + subfolder);
        }
    }
}
```

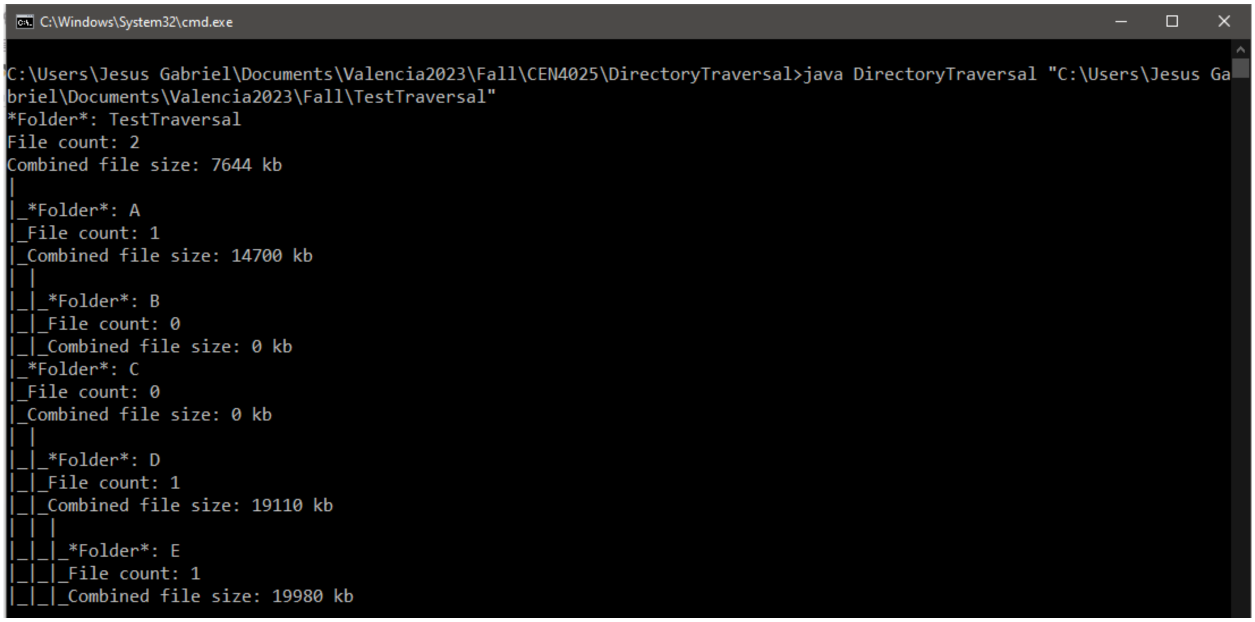
Output:

Given this sample folder structure



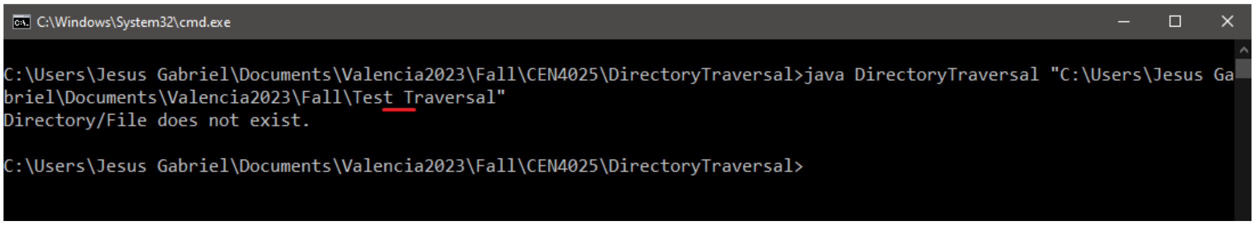


The program output is:



```
C:\Users\Jesus Gabriel\Documents\Valencia2023\Fall\CEN4025\DirectoryTraversal>java DirectoryTraversal "C:\Users\Jesus Ga
briel\Documents\Valencia2023\Fall\TestTraversal"
*Folder*: TestTraversal
File count: 2
Combined file size: 7644 kb
|
|_*Folder*: A
|_File count: 1
|_Combined file size: 14700 kb
| |
|_|_*Folder*: B
|_|_File count: 0
|_|_Combined file size: 0 kb
|_*Folder*: C
|_File count: 0
|_Combined file size: 0 kb
| |
|_|_*Folder*: D
|_|_File count: 1
|_|_Combined file size: 19110 kb
| | |
|_|_|_*Folder*: E
|_|_|_File count: 1
|_|_|_Combined file size: 19980 kb
```

The program also checks if the directory exists:



```
C:\Users\Jesus Gabriel\Documents\Valencia2023\Fall\CEN4025\DirectoryTraversal>java DirectoryTraversal "C:\Users\Jesus Ga
briel\Documents\Valencia2023\Fall\Test Traversal"
Directory/File does not exist.

C:\Users\Jesus Gabriel\Documents\Valencia2023\Fall\CEN4025\DirectoryTraversal>
```

Note:

Thanks to the previous Software Dev I class, I got used to uploading my program to a github repository. I initially was using string concatenation to increase the indent and formatting for the output of folders. From previous experience, using concatenation (example: “Hello” + “ world”) is more time and memory consuming than creating a Stringbuilder and using the built in function .append() to add a new string together. I created a new branch to test the addition of Stringbuilder without affecting the original source code.