Christopher Bussen

CPS 150 02 – Algorithms and Programming 1

Lab 13 – Files

11/5/20

**Problem 1 Algorithm**

1. Start the program
2. Open a file with the name input.txt
3. Import the scanner
4. Prompt the user to enter the message they want stored
5. Declare a String variable for the user’s message
6. Store the user’s message
7. Close the file
8. Open the same file again
9. Read the message into a new String variable
10. Print the message
11. End the program

**Problem 1 Running Screenshot**

**Text

Description automatically generated**

**Problem 1 Code**

/\*

Christopher Bussen

CPS 150 02

Lab 13

WriteFile: string ; string

program takes in a message from the user and writes a text file with the message

example: user inputs "Hello world" - program outputs Hello world and stores in a text file

\*/

import java.io.File;

import java.io.FileNotFoundException;

import java.io.PrintWriter;

import java.util.Scanner;

public class WriteFile {

public static void main(String [] args) throws FileNotFoundException {

//open a file with the name input.txt

PrintWriter out = new PrintWriter("input.txt");

//import scanner

Scanner input = new Scanner(System.in);

//prompt the user to enter the message they want stored and declare a String variable for its value

System.out.print("Please enter the message you want to be stored: ");

String userMessage = input.nextLine();

//store the user's message

out.println(userMessage);

//close the file

out.close();

//open the same file again

Scanner in = new Scanner(new File("input.txt"));

//read the message into a String variable

String message = in.nextLine();

//print the message

System.out.println(message);

}

}

**Problem 2 Algorithm**

1. Start the program
2. Give the full path and name of the file that you want to be deleted
3. Use the file that was created in the WriteFile program before this problem – make sure you run the WriteFile program first so that a file is present and can be deleted
4. Use a boolean variable and the delete method to delete the file
5. If the file deletion was successful, print a message telling the user the file was deleted
6. If the file deletion was not successful, print an error message telling the user the file was not deleted
7. End the program

**Problem 2 Running Screenshot**

**Text

Description automatically generated**

**Problem 2 Code**

/\*

Christopher Bussen

CPS 150 02

Lab 13

DeleteFile: no inputs ; String

program checks a certain path and file name on the computer then deletes

the file if it is present

ex1: program will print File was deleted if the path/name are correct and the file is successfully deleted

ex2: program will output File was unable to be deleted if path/name are incorrect or file is not present

\*/

import java.io.File;

import java.io.FileNotFoundException;

public class DeleteFile {

public static void main(String [] args) throws FileNotFoundException {

//give full file name and path of file to be deleted

//use file created in WriteFile program

File createdFile = new File ("/Users/ChristopherBussen/Desktop/UD F2020/CPS 150/Labs/Lab 13 - Files/input.txt");

//use a boolean variable and the delete method to delete the file

boolean delete = createdFile.delete();

if(delete){

//let user know file was successfully deleted

System.out.println("File was successfully deleted");

}

else{

//print error message

System.out.println("File was unable to be deleted");

}

}

}

**Problem 3 Algorithm**

1. Start the program
2. Give the full path and name of the file that you want to be renamed
3. Use the file that was created in the WriteFile program before this problem – make sure that the WriteFile program has been run before running this program so that the file you want renamed is present
4. Create a file object with the new file name
5. Use a boolean variable and the renameTo method to rename the file
6. If the file was successfully renamed, print a message telling the user the file was renamed
7. If the file was not successfully renamed, print an error message telling the user the file was not renamed
8. End the program

**Problem 3 Running Screenshot**

**Text

Description automatically generated**

**Problem 3 Code**

/\*

Christopher Bussen

CPS 150 02

Lab 13

RenameFile: no inputs ; string

program specifies a file name and path and then renames the file

ex1: program will print File was successfully renamed if the path/name are correct and the file is successfully renamed

ex2: program will output File was not renamed if path/name are incorrect or file is not present

\*/

import java.io.File;

import java.io.FileNotFoundException;

public class RenameFile {

public static void main(String [] args) throws FileNotFoundException{

//give full path and name of file that will be renamed

//use file created in WriteFile program

File createdFile = new File("/Users/ChristopherBussen/Desktop/UD F2020/CPS 150/Labs/Lab 13 - Files/input.txt");

//create a File object with the new file name

File renamedFile = new File("RenamedFile.txt");

//use a boolean variable and the renameTo method to rename the file

boolean rename = createdFile.renameTo(renamedFile);

//if the file is renamed, tell the user - otherwise print an error message

if(rename){

System.out.println("File was successfully renamed");

}

else{

System.out.println("File was not renamed");

}

}

}

**Problem 4 Algorithm**

1. Start the program
2. Specify the given directory of the files you want to be listed
3. For this problem, I will use a folder with the syllabi for each of my classes
4. Create an array of the files in the given directory using the list method
5. Print the list of files in the given directory using a for loop – start i at zero and end the for loop when i is not less than the length of the array
6. End the program

**Problem 4 Running Screenshot**

**Graphical user interface, text

Description automatically generated**

**Problem 4 Code**

/\*

Christopher Bussen

CPS 150 02

Lab 13

ListFilesInDirectory: no inputs ; string

program lists all of the files in a given directory

\*/

import java.io.File;

public class ListFilesInDirectory {

public static void main(String [] args){

//specify the given directory of the files you want listed

//for this problem, I will use a folder of the syllabi of my classes

File directory = new File("/Users/ChristopherBussen/Desktop/UD F2020/Syllabi");

//create an array of the files using the list method

String [] listOfFiles = directory.list();

//print the list of files using a for loop

for(int i = 0; i < listOfFiles.length; i++){

System.out.println(listOfFiles[i]);

}

}

}

**Problem 5 Algorithm**

1. Start the program
2. Create an array for contents to be copied into
3. Specify the name of the file that you want to be copied
4. Specify the name of the file that you want contents to be copied into
5. Use the read method to read the contents of the original file
6. Use the write method to write data into the new file
7. Print a message telling the user the file was successfully copied
8. Close both the original file and the copied file
9. Use a catch exceptions clause to prevent errors
10. End the program

**Problem 5 Running Screenshot**

**Text

Description automatically generated**

**Problem 5 Code**

/\*

Christopher Bussen

CPS 150 02

Lab 13

CopyFileToNewFile: no inputs ; string

program specifies a file name and path and then copies that file's

contents to a new file

ex1: program will print File was successfully copied if the path/name are correct and the file is successfully copied

\*/

import java.io.FileInputStream;

import java.io.FileOutputStream;

public class CopyFileToNewFile {

public static void main(String [] args){

//create an array for contents to be copied in

byte [] array = new byte[50];

try {

//specify the file name of the file you want to be copied and the name of the file you want contents to be copied to

FileInputStream originalFile = new FileInputStream("input.txt");

FileOutputStream copiedFile = new FileOutputStream("copiedFile.txt");

//use read method to read data from original file

originalFile.read(array);

//use write method to write data into new file

copiedFile.write(array);

//print message telling user the file was successfully copied

System.out.println("File was successfully copied");

//close both files

originalFile.close();

copiedFile.close();

}

catch (Exception e){

e.getStackTrace();

}

}

}