File Divider:

```
import javax.swing.JFileChooser;
//import java.util.Scanner;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.io.IOException;

public class FileDivider {
	/**
	 * This program is designed to split a file into five pieces.
	 * To do this, it needs to choose a file, split it into five
	 * pieces, and then save each piece as a new file.
	 */

	JFileChooser chooser = new JFileChooser();
	File target;
	String filePath;
	private static final int TARGET_FILE_COUNT = 5;
	private static final int MAX_BYTES = 120000000;


	public File openFile() {
		/**
		 * This method uses a JFileChooser to let the user select
		 * a file to split.
		 */
		chooser.setDialogTitle("Pick a file to divide.");
		boolean foundGoodFile = false;
		while (!foundGoodFile) {
			int rVal = chooser.showOpenDialog(chooser);
			if (rVal == JFileChooser.APPROVE_OPTION) {
				target = chooser.getSelectedFile();
				if (target.canRead()) {
					foundGoodFile = true;
					filePath = chooser.getSelectedFile().getPath();
```

```
                                System.out.println(filePath);
                        }
                }
                if (rVal == JFileChooser.CANCEL_OPTION) {
                        System.out.println("Operation cancelled.");
                        System.exit(0);
                }
        }
        return target;

}

public String[] divideFile(File base) throws IOException {
        /**
         * This method divides a file into the target number
         * of strings, to later be printed again.
         */
        //String fullFile = "";
        char[] charFile = new char[MAX_BYTES];
        String[] sections = new String[TARGET_FILE_COUNT];
        //read the whole file
        //Scanner in = null;
        FileInputStream in2 = null;
        try {
                //in = new Scanner(base);
                in2 = new FileInputStream(base);
                /*
                if (in.hasNextLine()) {
                        fullFile = in.nextLine();

                        System.out.println("Read first line");
                }
                while (in.hasNextLine()) {
                        fullFile = fullFile + "\n" + in.nextLine();
                        System.out.println("Read a line");
                }
                //*/
                int r;
                System.out.println("Started reading.");
```

```
                for (int i = 0; (r = in2.read()) != -1; i++) {
                        charFile[i] = (char) r;
                }
        } finally {
                if (in2 != null) {
                        //in.close();
                        in2.close();
                        System.out.println("Done reading.");
                }
        }
        //do the math and divide the file
        String stringFile = String.valueOf(charFile).trim();
        int minStringSize = stringFile.length() / TARGET_FILE_COUNT;

        //int minSize = charFile.length / TARGET_FILE_COUNT;
        //int leftovers = fullFile.length() % TARGET_FILE_COUNT;
        for (int i = 0; i < TARGET_FILE_COUNT; i++) {
                if (i == TARGET_FILE_COUNT - 1) {
                        sections[i] = stringFile.substring(i * minStringSize);
                } else {
                        sections[i] = stringFile.substring(i * minStringSize, (i + 1) *
minStringSize);
                }
        }
        System.out.println("Done dividing.");
        return sections;
    }

    public void saveSections(String[] sections) throws IOException {
        String filetype = filePath.substring(filePath.lastIndexOf('.'));
        String filename = filePath.substring(0, filePath.lastIndexOf('.'));
        for (int i = 0; i < TARGET_FILE_COUNT; i++) {
                FileWriter out = null;
                try {
                        out = new FileWriter(String.format(filename + (i + 1) + filetype));
                        out.write(sections[i]);
                        System.out.println((i + 1) + " of " +TARGET_FILE_COUNT + "
files saved.");
```

```
                        } catch (FileNotFoundException e) {
                                System.out.println("File not found exception.");
                        } finally {
                                if (out != null) {
                                        out.close();
                                }
                        }
                }
        }
}
```

Driver:

```
import java.io.FileNotFoundException;
import java.io.IOException;

public class FileDriver {
        /**
         * This class initializes and manages the operation of the FileDivider class.
         */

        public static void main(String[] args) throws IOException, FileNotFoundException {
                FileDivider fd = new FileDivider();
                fd.saveSections( fd.divideFile( fd.openFile() ) );
        }
}
```

Files:

test.txt
Text Document

Date modified: 2016-09-24 10:21 AM
Size: 1.00 KB

toast.txt
Text Document

Date modified: 2016-09-23 5:42 PM
Size: 100 MB

test1.txt
Text Document

Date modified: 2016-09-24 1:29 PM
Size: 204 bytes

test2.txt
Text Document

Date modified: 2016-09-24 1:29 PM
Size: 204 bytes

test3.txt
Text Document

Date modified: 2016-09-24 1:29 PM
Size: 204 bytes

test4.txt
Text Document

Date modified: 2016-09-24 1:29 PM
Size: 204 bytes

test5.txt
Text Document

Date modified: 2016-09-24 1:29 PM
Size: 208 bytes

toast1.txt
Text Document

Date modified: 2016-09-24 1:50 PM
Size: 20.0 MB

toast2.txt
Text Document

Date modified: 2016-09-24 1:50 PM
Size: 20.0 MB

toast3.txt
Text Document

Date modified: 2016-09-24 1:50 PM
Size: 20.0 MB

toast4.txt
Text Document

Date modified: 2016-09-24 1:50 PM
Size: 20.0 MB

toast5.txt
Text Document

Date modified: 2016-09-24 1:50 PM
Size: 20.0 MB

Our assignment was to split a file ranging from 1KB to 100MB into five pieces. This program needed to be able to open a file, divide its contents into the desired number of parts, and then save each part into its own file.

Opening the file would be easy enough; I used a JFileChooser to open a window to let the user select a file without a hassle. The divideFile() method would be need to establish the given file's length and use it divide the file into a desired number of smaller Strings. Originally, I thought to use a scanner to build a string representation of the entire file, which I could then use to find the length needed to write each of the new files. It worked well enough with the small file, but giving the program the 100MB "toast.txt" made my poor laptop delay for an extremely long time and swelled the memory usage to ridiculous levels. Instead, since a size cap of 100MB was specified, I decided it would be easier to use a character array of a fixed maximum size to hold the data from the file. That contained the memory usage issue and reduced the wait for the 100MB file to be stored. From there, I stored the character array into a String, used the String's trim() method to cut off the extra empty spaces at the end of the character array, and made the desired number of substrings based on the String's length / TARGET_FILE_COUNT. The remainder from this division fits within the accepted difference in file sizes, so it was added to the last of the substrings. Finally, the saveSections() method took each new substring returned by divideFile() and saved it into its own file, preserving the type of the original file.