# Exercise: Streams, Files and Directories

## Even Lines

Write a program that reads a **text** file and prints on the console its **even** **lines**. Line numbers start from 0. Use **StreamReader**. Before you print the result replace {"-", ",", ".", "!", "?"} with "@" and reverse the order of the words.

### Examples

|  |  |
| --- | --- |
| **text.txt** | **output** |
| -I was quick to judge him, but it wasn't his fault.  -Is this some kind of joke?! Is it?  -Quick, hide here. It is safer. | fault@ his wasn't it but him@ judge to quick was @I  safer@ is It here@ hide @Quick@ |

## Line Numbers

Write a program that **reads** a **text** **file** and inserts **line** **numbers** in front of **each** of its **lines and count all the letters and** **punctuation marks**. The result should be **written** to **another** text file. Use the static class **File**.

### Examples

|  |  |
| --- | --- |
| **text.txt** | **output.txt** |
| -I was quick to judge him, but it wasn't his fault.  -Is this some kind of joke?! Is it?  -Quick, hide here. It is safer. | Line 1: -I was quick to judge him, but it wasn't his fault. (37)(4)  Line 2: -Is this some kind of joke?! Is it? (24)(4)  Line 3: -Quick, hide here. It is safer. (22)(4) |

## Word Count

Write a program that reads a **list** of **words** from the file **words.txt** and finds **how** **many** **times** each of the words is **contained** in another file **text.txt**. Matching should be **case-insensitive**. Write the results in file **actualResults.txt**. **Sort** the words by **frequency** in **descending** order and then compare the result with the file **expectedResult.txt**. Use the **File** class.

### Examples

|  |  |  |  |
| --- | --- | --- | --- |
| **words.txt** | **text.txt** | **actualResult.txt** | **expectedResult.txt** |
| quick  is  fault | -I was quick to judge him, but it wasn't his fault.  -Is this some kind of joke?! Is it?  -Quick, hide here. It is safer. | quick - 2  is - 3  fault - 1 | is - 3  quick - 2  fault - 1 |

## Copy Binary File

Write a program that copies the contents of a binary file (e.g. image, video, etc.) to another using **FileStream**. You are **not allowed** to use the **File** class or similar helper classes.

## Directory Traversal

Write a program that traverses a given **directory** for **all** **files** with the given **extension**. Search through the **first** **level** of the **directory** **only** and write information about each **found** file in **report.txt**. The files should be **grouped** by their **extension**. **Extensions** should be **ordered** by the **count** of their files **descending**, then by **name alphabetically**. **Files** under an extension should be **ordered** by their **size**. **report.txt** should be saved on the **Desktop**. Ensure the desktop path is always valid, regardless of the user.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Directory View** | **report.txt** |
| . | directory | .cs  --Mecanismo.cs - 0.994kb  --Program.cs - 1.108kb  --Nashmat.cs - 3.967kb  --Wedding.cs - 23.787kb  --Program - Copy.cs - 35.679kb  --Salimur.cs - 588.657kb  .txt  --backup.txt - 0.028kb  --log.txt - 6.72kb  .asm  --script.asm - 0.028kb  .config  --App.config - 0.187kb  .csproj  --01. Writing-To-Files.csproj - 2.57kb  .js  --controller.js - 1635.143kb  .php  --model.php - 0kb |

## Zip and Extract

Write a program that **creates** a **zip** file in a given **directory** and **extracts** it in **another** one. Use the **copyMe.png** file from your resources and zip it in a directory of your choice. **Extract** the zip file in another directory, again, by your choice.

### Hint

Use the **ZipFile** class.