Computer Science 2A

Practical Assignment 03

Assignment date: 2024-03-05

Deadline 2024-03-12 12h00

Marks: 95

This practical assignment must be uploaded to eve.uj.ac.za **before** 2024-03-12 12h00. Late[1] or incorrect submissions **will not be accepted**, and will therefore not be marked. You are **not allowed to collaborate** with any other student.

Good coding practices include a proper coding convention and a good use of documentation. Marks will be deducted if these are not present. Every submission **must** include a batch file unless stated otherwise.

The **reminder page** includes details for submission. Please ensure that **ALL** submissions follow the guidelines. The reminder page can be found on the last page of this practical.

This practical aims to introduce Text IO and Regular Expressions.

The Lead PyroTechnicians meticulously craft their firework displays by generating text files, which are then dispatched to the **Firework Management Bureau (FMB)** [2]. However, the intense sound from the fireworks can sometimes induce disruptions, leading to the transmission of corrupted or inaccurate text instructions. The FMB needs your help to examine and identify any messages that may be invalid or compromised, as sent by the Lead PyroTechnicians. You will be given access to a set of files for this task, and it will be your responsibility to utilize regular expressions for the validation of the data within these files. The transmission file is structured accordingly:

File information format

```
// Data is separated with TABs
// The first line is the Firework Display data
FD_ID   [FD_NAME]   "FD_THEME"
// Next is the Lead PyroTechnician
PT_FULL_NAME   PT_PHONE_NUMBER
// The rest of the file is filled with Fireworks
F_ID   F_NAME   F_FUSE_LENGTH   F_COLOUR
F_ID   F_NAME   F_FUSE_LENGTH   F_COLOUR
F_ID   F_NAME   F_FUSE_LENGTH   F_COLOUR
                ...
F_ID   F_NAME   F_FUSE_LENGTH   F_COLOUR
```

[1] Alternate arrangements for exceptional circumstances will been posted on eve.

[2] Disclaimer - This series of problem statements are a work of fiction. Names, characters, businesses, places, events and incidents are either the products of the author's imagination or used in a fictitious manner. Any resemblance to actual persons, living or dead, or actual events is purely coincidental.

The different attributes are defined as:

`FD_ID` - The Unique Identifier of the Firework Display eg. FD0076
`FD_NAME` - The Name to commemorate the Firework Display e.g. [Ode to Joy]
`FD_THEME` - The Theme the Firework Display aims to achieve e.g. ”New Years Eve”
`PT_FULL_NAME` - The Full Name of the Lead PyroTechnician e.g. ”Jane-Doe”
`PT_PHONE_NUMBER` - The Phone Number to call when things ... ummm ... BLOW UP e.g. ”555-010-9111”
`F_ID` - The Unique Identifier of the Firework, starts with FW, then 6 additional digits eg. FW007600
`F_Name` - The firework’s name on record e.g. ”Bouncing Betty”
`F_FUSE_LENGTH` - The eplosion delay (in seconds) e.g. 1.2
`F_COLOUR` - The `E_COLOUR` produced by the Firework

To limit problems in the transmissions the **FMB** specify the following conditions:

- `Firework Display ID`s must start with `FD` followed by 4 additional digits. Example: `FD1234`.
- `Firework Display Name`s can be any length, separated by spaces and placed within square brackets `[ ]`. Example: `[New Year's Eve Spectacular]`.
- `Firework Display Theme`s can be any length, separated by spaces and placed within quotation marks `" "`. Example: `"Winter Wonderland"`.
- `PyroTechnician Full Name`s are formatted as follows: `Yxxxx-Yxxxxx`, where `Y` is any capital letter and `x` is any number of lowercase letters. The hyphen `-` joins the First Name and the Last Name. Example: `Ajohn-Bsmiiiiiith`.
- `PyroTechnician Phone Number`s must start **and** end with a digit that is not ’0’, and formatted as `XXX-XXX-XXXX`, where `X` is a digit and hyphens `-` separate the digit groups. Example: `123-456-7891`.
- `Firework ID`s must start with `FW` followed by 6 additional digits. Example: `FW123456`.
- `Firework Name`s can be any length made using uppercase and lowercase letters in any position and can contain spaces between the words. Example: `Sky Blossom`.
- `Firework Fuse Length`s must be any value greater than 0, as well as any decimal value greater than 0. Example: `0.5` or `2.52`.
- Only the following legal `E_COLOUR`s will be considered :
  - RED
  - ORANGE
  - GREEN
  - BLUE
  - YELLOW
  - MAGENTA
  - WHITE
  - CYAN

You have been provided with a models.jar file to make use of and the following changes need to be made:

- **FMB** requires that all development is correctly and accurately modelled and therefore requires you to complete a UML class diagram for all classes in your solution.
- Create a `DisplayFileHandler` class in the `acsse.csc2a.fmb.file` package. The class must have a `readDisplay` method which:
  - Takes a filename as a parameter
  - Reads through each line of the file and tests for validity. If not valid, an error message is printed to the error stream.
  - Create a `FireworkDisplay` instance and add `Firework`s as described
  - Returns the `FireworkDisplay` instance found in the file

  - Handle any exceptions that arise during the File IO process
- In the `Main` class, read the text files provided using the `DisplayFileHandler` and display the valid `Display`s and respective `Firework`s.
- You must display all the valid data, and place it in a user friendly and easily readable format.

Hints

The following regex might be of assistance:

- The escape character `\`: Used to escape reserved regex symbols. For example, `.` is a special class that matches anything. If you want to match a decimal point, simply using `.` in your pattern will not work as expected. Instead, you need to escape it like so: `\.`.
- The optional quantifier `?`: Indicates that the preceding element is optional, meaning it might be present, but its absence is also acceptable.
- The one-or-more quantifier `+`: Indicates that the preceding element must appear at least once.
- Combining quantifiers `?` and `+`: These quantifiers can be combined to match a pattern of any length that may be separated by a specific symbol. For instance, to match text like ”a-b-c”, where any lowercase letter is optionally followed by a hyphen (`-`) of any length, you can use the following regex: `([a-z]-?)+`.

Marksheet

1. UML class diagrams for all classes in all packages[3] [10]
2. `DisplayFileHandler` class
   (a) Declare an instance of `Display` [02]
   (b) Open file [02]
   (c) Read file [02]
   (d) Regular Expressions (`FireworkDisplay`, `Firework`, and the lead `PyroTechnician`) [05]
   (e) Match check for `FireworkDisplay` [05]
   (f) Match check for `Firework` [05]
   (g) Match check for `PyroTechnician` [05]
   (h) Create `FireworkDisplay` instance [02]
   (i) Add `Firework` instances to `FireworkDisplay` [02]
   (j) Close file [02]
   (k) Return `FireworkDisplay` instance [02]
   (l) Exception handling [06]
3. `Main` class
   (a) Read file using `DisplayFileHandler`. [05]
   (b) Display `FireworkDisplay` information (including `Firework`s, and the lead `PyroTechnician`) [05]
4. Packages [05]
5. Coding convention (structure, layout, OO design) [05]
6. Commenting (normal and JavaDoc commenting). [05]
7. Correct execution (if it doesn't run from your batch file you get 0)
   (a) Can create and correctly display normal text [10]
   (b) Can print error for invalid text [10]

NB

Submissions which **do not compile** will be capped at 40%!

Practical marks are awarded subject to the student's ability to explain the concepts and decisions made in preparing the practical assignment solution. (Inability to explain code = inability to be given marks.)

Execution marks are awarded for a correctly functioning application and not for having related code.

[3] Failure to submit code will result in this section being awarded 0

Reminder

Your submission must follow the naming convention below.

`SURNAME_INITIALS_STUDENTNUMBER_SUBJECTCODE_YEAR_PRACTICALNUMBER`

Example

Surname	Berners-Lee	**Module Code**	CSC02A2
Initials	TJ	**Current Year**	2024
Student number	209912345	**Practical number**	P03

`Berners-Lee_TJ_209912345_ CSC02A2_2024_P03`

Your submission must include the following folders:

Folder	State	Purpose
`bin`	*Required*	Should be empty at submission but will contain runnable binaries when your submission is compiled.
`docs`	*Required*	Contains the batch file to compile your solution, UML diagrams, and any additional documentation files. All files must be in **PDF** format. Your details must be included at the top of any **PDF** files submitted. **Do not include generated JavaDoc.**
`src`	*Required*	Contains all relevant source code. Source code must be places in relevant sub-packages! Your details must be included at the top of the source code.
`data`	*Optional*	Contains all data files needed to run your solution.
`lib`	*Optional*	Contains all libraries needed to compile and run your solution.

NB

Every submission **must** include a batch file that contains commands which will:

- Compile your Java application source code.
- Compile the associated application JavaDoc.
- Run the application.

Do not include generated JavaDoc in your submission. All of the classes/methods which were created/updated need to have JavaDoc comments.

Multiple uploads

Note that only **one** submission is marked. If you already have submitted once and want to upload a newer version then submit a newer file with the same name as the uploaded file in order to overwrite it.