Computer Science 2A

Practical Assignment 02

Assignment date: 2023-02-28

Deadline 2023-03-07 12h00

Marks: 200

This practical assignment must be uploaded to eve.uj.ac.za **before** 2023-03-07 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 familiarise you with Problem Analysis and Model Classes.

The **Firework Management Bureau (FMB)**, recognising your expertise and track record, has specifically requested your assistance in developing a comprehensive system to manage their spectacular firework displays. [2] With numerous *Firework Displays* featuring a variety of *Fireworks*, the FMB seeks to enhance coordination for both efficiency and effectiveness in their endeavors.

Your task is to develop a robust system capable of not only organizing the *FireworkDisplays* but also meticulously managing the diverse array of *Fireworks* used, thereby ensuring seamless execution and maximizing the impact of each event. The **FMB** requires the ability to create multiple *Firework Displays*, each comprising various *Fireworks*. To ensure accountability in case of any mishaps, a lead *Pyrotechnician* is assigned to oversee each Firework Display.

The **FMB** keeps track of each `Pyrotechnician`. This data consists of:

- Full Name - the name and surname of the Technician, formatted as "<Name>-<Surname>" e.g. "Jane-Doe"
- Phone Number - to call them when things ... ummm ... BLOW UP

To conform with law, the **FMB** use an enumeration `E_COLOUR` to ensure only the following legal colours are used:

- RED, GREEN, BLUE, YELLOW, MAGENTA, WHITE, CYAN

[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 **FMB** keeps track of each `Firework`. This data consists of:

- Firework ID - a unique alphanumeric ID for the firework
- Firework Name - the firework's name on record
- Firework Fuse Length - the explosion delay (in seconds)
- Firework Colour - the `E_COLOUR` produced by the firework

Each `FireworkDisplay` in the **FMB** also has its information stored. This data consists of:

- Display ID - a unique alphanumeric ID for the display
- Display Name - the display's name on record (e.g. Ode to Joy)
- Display Theme - the theme targetted by the display (E.g. New Year's Day)
- Lead Technician - the `Pyrotechnician` in charge of the display
- Fireworks - the `Fireworks` used in the display

Using the information provided in the problem statement above, do the following:

- Create a `Pyrotechnician` class with the required attributes[3].
- Create an `E_COLOUR` enumeration with the required colours.
- Create a `Firework` class with the required attributes[3].
  - Make use of an ENUM TYPE for the Firework Colour.
- Create a `Firework Display` class with the required attributes[3].
  - A `Firework Display` must have a way to add new `Fireworks`
- Create a `Main` class.
  - Create two `Firework Display` instances.
  - For each `Firework Display` add two `Fireworks` and assign a `Pyrotechnician`.
  - Call the `printDisplay` method on each `Firework Display`.
  - Make sure the output is well formatted.

Place the relevant classes into the `acsse.csc2a` package[4].

Mark Sheet

1. UML class diagram [10]
2. `Pyrotechnician` class
   (a) Attributes (4 marks per attribute) [08]
3. `E_COLOUR` enumeration
   (a) Declaration [02]
4. `Firework` class
   (a) Attributes (4 marks per attribute) [12]
5. `FireworkDisplay` class
   (a) Attributes (4 marks per attribute) [20]

[3] Hint: remember that accessor and mutator methods for attributes are required.
[4] Hint: The `Main` class does not need to be in a package

(b) `printDisplay` [04]

(c) `addFirework` [04]

6. `Main` class [10]
7. Packages [05]
8. Coding convention (structure, layout, OO design) [05]
9. Commenting (normal and JavaDoc commenting) [05]
10. Correct execution [115]

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.

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**	P02

`Berners-Lee_TJ_209912345_ CSC02A2_2024_P02`

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.