Exercise Recording

Program

소프트웨어학부

20183401 박정훈

20185912 장민혁

20186091 이동건

20185016 박호동



index

title1

index2

*brief description or oop concept3*

UML modeling4

*implement issue7*

compile issue10

execution results11

Brief description or oop concept

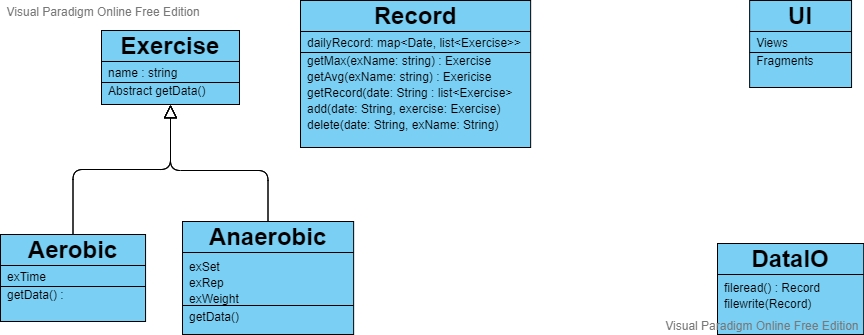
We created a program that allows people to record exercise. Our target user is people who do weight training or aerobic. User can record today’s exercise set, repetition, weight for anaerobic exercise, and time for aerobic exercise. Also, we implemented delete or modify, search function.

We made our service with Kotlin, which is object-oriented language. We segment exercise class to aerobic, anaerobic class. These classes inherit exercise class and override method. We use **inheritance** in here. So, we increase the reusability of code. If we add more specific exercise type or method, we add another class which inherits exercise class. We also use **encapsulation** in each exercise. We made all recorded exercise to instance of exercise class. So, we can make all the exercise to same attribution. We programed all the attributes of exercise class to **hide information**. Finally, we use **abstraction** on the exercise class to declare method without the content. We write contents on Aerobic and Anaerobic class.

Our program read & write data only in record class. We implement UI class for users to access the program. We segment data processing, algorithm of program, user’s interface. So, if when changing of UI or data processing, you can change that part regardless of other parts.

UML modeling

* class diagram



Our program consists of those classes.

**Record class** store user’s exercise records. Each method calculates the max, avg exercise record and get exercise record on the date users select in record class.

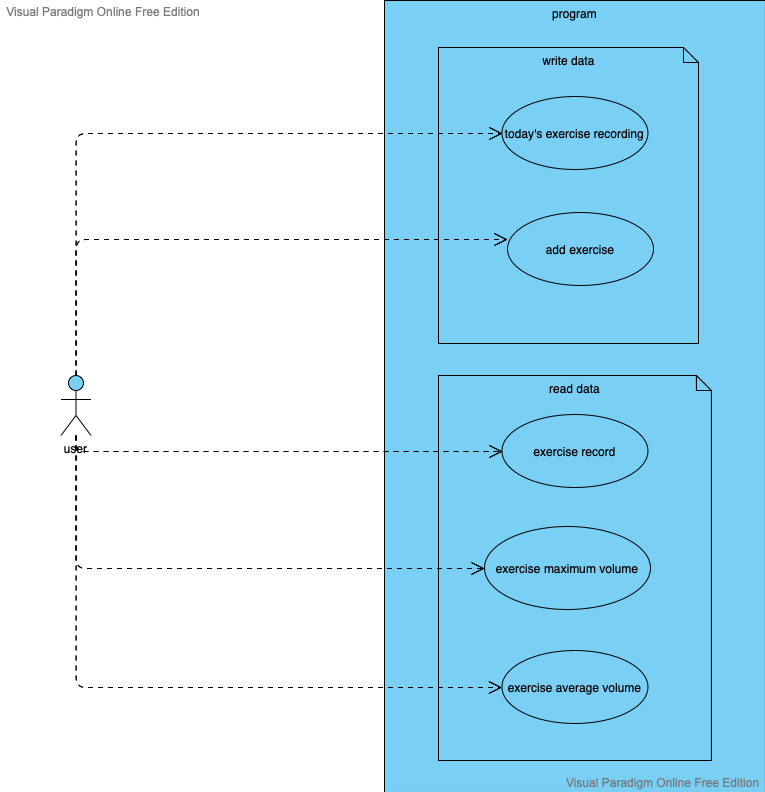
**Exercise class** has exercise name attribute and getData() method.

**Aerobic & Anaerobic class** inherit Exercise class. Aerobic class has time attribute. And Anaerobic class has set, rep, weight attribute.

**DataIO class** is just for read & write text file.

**UI class** is for UI which consists of View and Fragment. We made main UI is View, and others are Fragment.

* use-case diagram



Our program segment read data field & write data field.

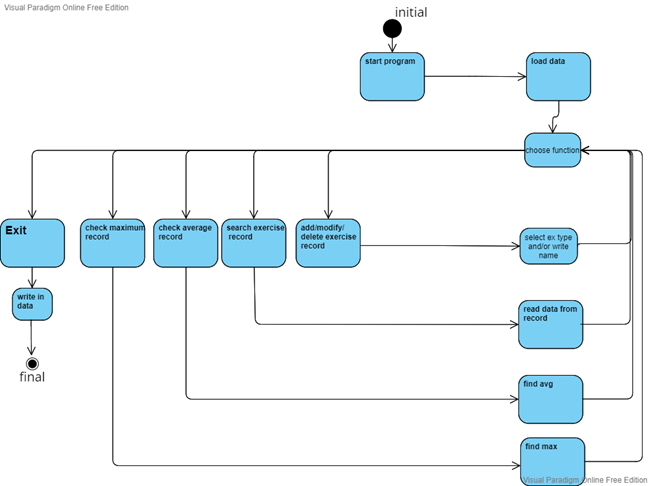
In write data field, mainly user record his today’s exercise.

User can add exercise in person.

In read data field, when writing exercise name, users can read the exercise record.

Also, user can read maximum or average exercise record.

* activity diagram



When user start program, it read the text file first.

When user choose any menu, then print that in UI.

After that, program works for selection. Print the UI for user to add, delete, modify, search his exercise record.

Reading menu print the UI for user to search exercise. searching UI has radio button to choose max or average or normal record.

When program exit, write the data in the text file.

Implement issue

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

In file i/o, to segment data, we append “|” between the data. we also parse “|” to read the data only.

텍스트이(가) 표시된 사진

자동 생성된 설명

We declare exercise class abstract class. Aerobic class inherit exercise class and override getData method. Also use private variable not to access the program.

텍스트이(가) 표시된 사진

자동 생성된 설명

Add weight with same name. Then divide sum with array size of same exercise. Likewise, in method getMax, compare the weight with same name. Finishing iteration, max is returned.

Compile issue

We had a hard time running the program because of build environment. So, we notice our build environment.

We use **intellij.**

**jdk** version is **17**.

**gradle** version is **7.3.**

Execution results

* **main UI**

**텍스트, 스크린샷, 모니터, 실내이(가) 표시된 사진

자동 생성된 설명**

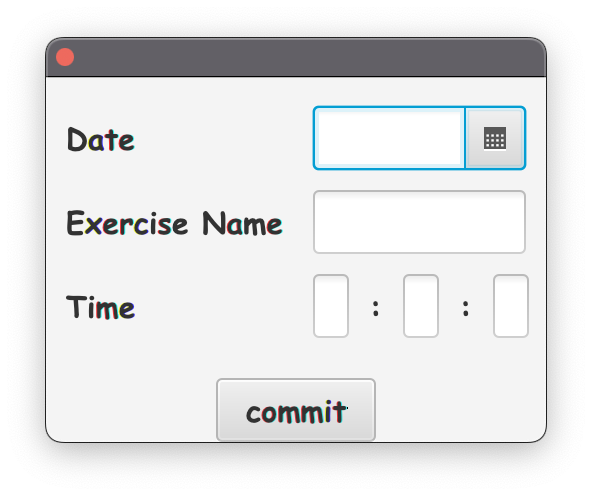
This is main UI. You can choose add, modify, delete and search function. If you click the date or exercise button, it sort the date or exercise in ascending or descending way.

* **add UI**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

User choose which exercise user want to add.

**텍스트이(가) 표시된 사진

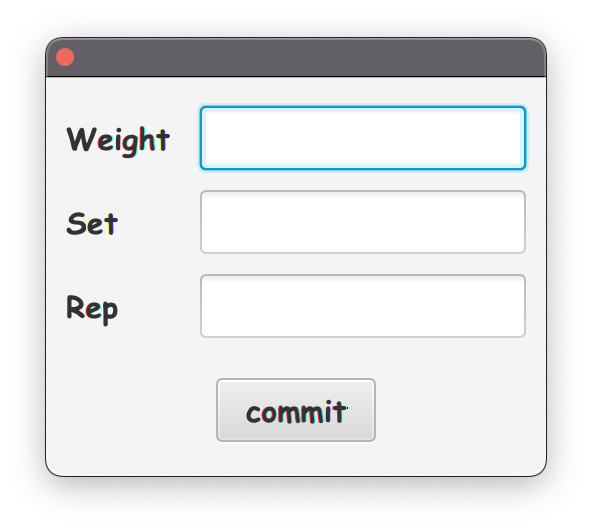
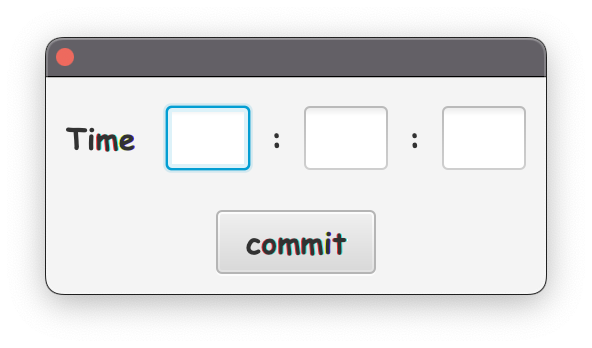
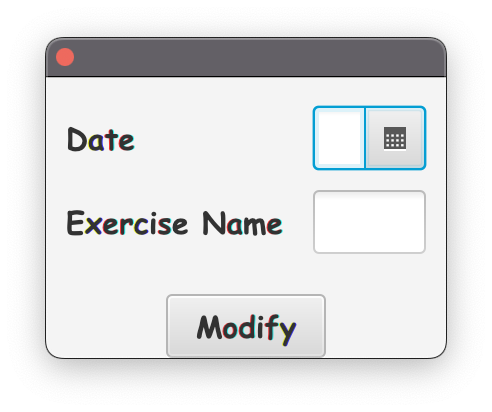
자동 생성된 설명**

Left is aerobic UI. Aerobic UI has time attribute.

Right is anaerobic UI. Anaerobic UI has weight, set, rep attribute.

Name and date is common attribute.

* **modify & delete UI**

****

User writes date and exercise name. If user write correct data, it prints the correct UI. Otherwise, if user write incorrect data, error UI which we made will be shown.

If user writes aerobic exercise, program gives the UI of write time attribute. If user write anaerobic exercise, program gives the UI of write weight, set , rep attributes.

Delete UI is as same as first.

* **search UI**

**텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명**

When user chooses the search function, first it shows search option and exercise name to select.

If user chooses max, maximum record of the exercise will be printed. And if user chooses avg, average record of the exercise will be printed.

**텍스트, 모니터, 스크린샷이(가) 표시된 사진

자동 생성된 설명**

If user chooses normal exercise record, program show the table of date, exercise, and exercise’s options.

Conclusion

We focused on expand the program. However, it was not easy, and the output was not like we expected as we designed. It was able to separate each function so we can upgrade program easily but the polymorphism on exercise class and record class is not perfect. We intended add types of exercise more flexible without touching the record class, but when we add new types on exercise, we should fix record class. This leaves much to be desired.