Лабораторна робота №3

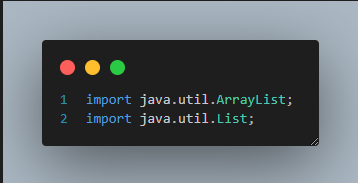
Тема: Базові принципи ООП

Мета: Використовуючи теоретичне підґрунтя про базові принципи ООП та підходи використання в Java виконати дії що будуть вказано в завданні до лабораторної роботи.

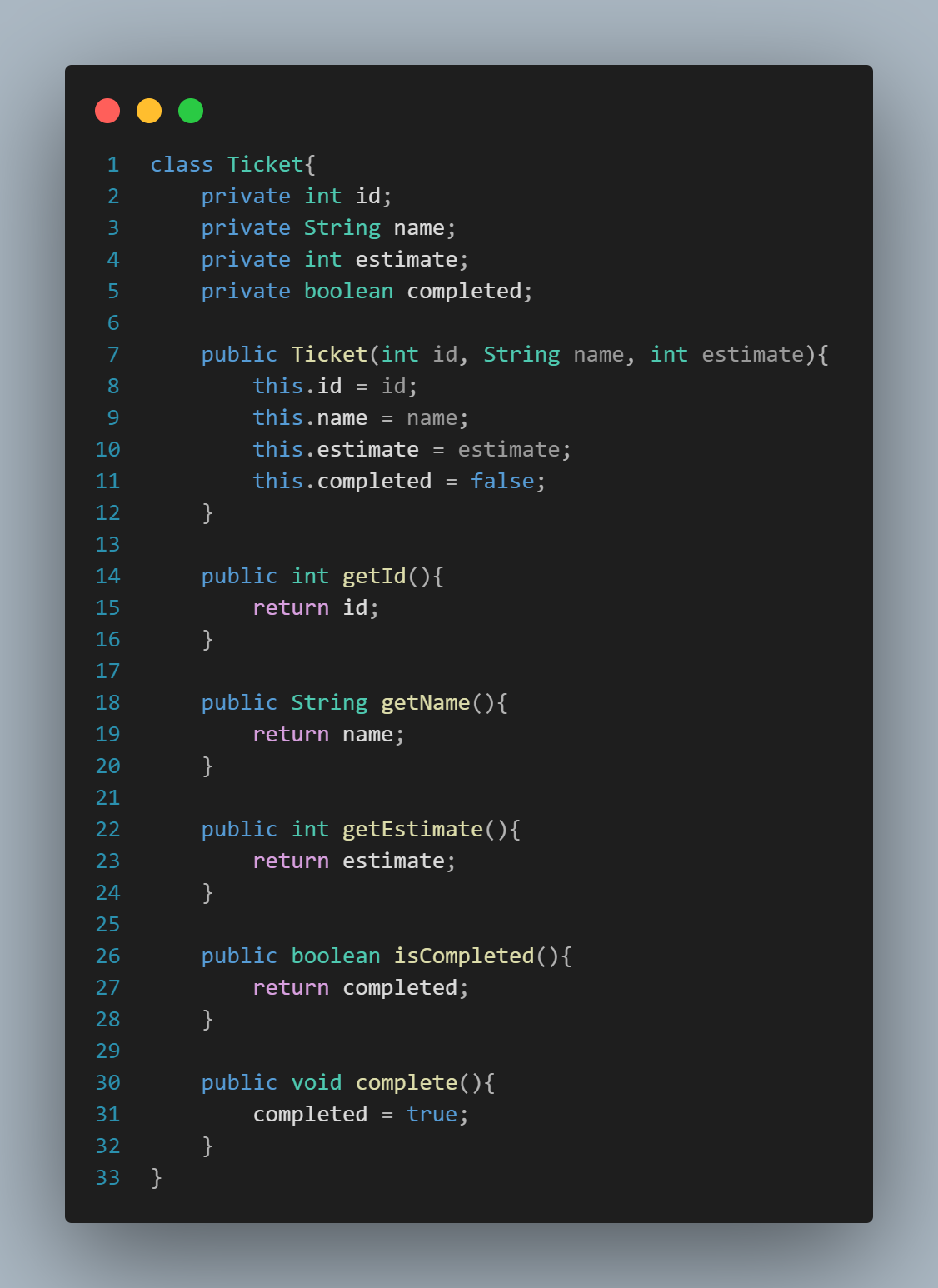
Хід роботи

Завдання 1:

Імпортував додаткові бібліотеки



Написав код для Ticket



Тікет містить свій ідентифікатор, ім’я та приблизний час для його виконання. Ці значення надаються через конструктор класу Тікет. Тікет може вважатися завершеним або не завершеним. Щойно створений тікет вважається незавершеним.

*getId*() - повертає ідентифікатор ticket'у.

*getName*() - повертає ім'я ticket'у.

*getEstimate*() - повертає оцінку ticket'у.

*isCompleted*() - повертає *true*, якщо ticket завершено, інакше — *false*.

*complete*() - переводить ticket у завершений стан.

Написав код для UserStory:



UserStory являє собою історію користувача та представляє тікет, який може містити залежності. Залежності – це екземпляри, які мають бути завершені в першу чергу, щоб потім вдалося завершити і залежне від них UserStory. Залежності передаються через конструктор класу UserStory.

*complete*() - як і метод Ticket#complete() - переводить ticket у завершений стан. Різниця лише в тому, що історія користувача може не бути завершена, якщо його залежність ще не завершена.

*getDependencies*() - повертає копію масиву ticket'ів на спринт, що захищена від змін.

*toString*() - повертає рядок, що представляє *UserStory*, використовуючи його ідентифікатор та ім'я. Приклад: з ідентифікатором =1, ім’я *UserStory* = "Об'єкт реєстрації користувача", отримаємо "[US 1] Об'єкт реєстрації користувача".

Написав код для Bug:



Цей клас також є підкласом підтипом класу тікет і представляє помилку або дефект, пов’язаний з певною історією користувача.

Написав код для Sprint:



Цей клас представляє спринт у методології Scrum. Має поля для обмеження ємності (capacity), обмеження кількості тікетів (ticketLimit) та список tickets. Методи addUserStory() та addBug() додають історію користувача або баг до спринту з урахуванням різних умов (наприклад, завершеності, обмежень по ємності тощо). Метод getTickets() повертає список всіх тікетів у спринті. Метод getTotalEstimate() обчислює загальну оцінку часу виконання всіх тікетів у спринті.

Результат:

Для того, щоб перевірити роботу програми, мені потрібно було написати іншу програму, яка б створила помилку та вивела її на екран.





Висновок: Використав теоретичне підґрунтя про базові принципи ООП та підходи використання в Java виконав дії що були вказані в завданні до лабораторної роботи