Плотніков Артем КБ-232

**Об'єктно-орієнтоване програмування  
Звіт про виконання лабораторної роботи №3**

Тема: Базові принципи ООП та їх застосування для організації планування спринту (SCRUM) в Java  
Мета роботи: Ознайомлення з основними принципами об'єктно-орієнтованого програмування (інкапсуляція, спадкування, поліморфізм), створення ієрархії класів для моделювання планування спринту, реалізація логіки взаємодії між задачами (Ticket, UserStory, Bug) та обмеженнями спринту.

Github link: <https://github.com/artembombardir/OOP-KB232-Artem-Plotnikov/tree/main/lab03>

**Ticket.java**

public class Ticket {

private int id;

private String name;

private int estimate;

private boolean completed;

public Ticket(int id, String name, int estimate) {

this.id = id;

this.name = name;

this.estimate = estimate;

this.completed = false;

}

public int getId() { return id; }

public String getName() { return name; }

public int getEstimate() { return estimate; }

public boolean isCompleted() { return completed; }

public void complete() { completed = true; }

}

**UserStory.java**

import java.util.Arrays;

public class UserStory extends Ticket {

private UserStory[] dependencies;

public UserStory(int id, String name, int estimate, UserStory[] dependencies) {

super(id, name, estimate);

this.dependencies = (dependencies == null) ? new UserStory[0] : dependencies.clone();

}

@Override

public void complete() {

for (UserStory dep : dependencies) {

if (!dep.isCompleted()) {

return;

}

}

super.complete();

}

public UserStory[] getDependencies() {

return dependencies.clone();

}

@Override

public String toString() {

return "[US " + getId() + "] " + getName();

}

}

**Bug.java**

public class Bug extends Ticket {

private UserStory userStory;

private Bug(int id, String name, int estimate, UserStory userStory) {

super(id, name, estimate);

this.userStory = userStory;

}

public static Bug createBug(int id, String name, int estimate, UserStory userStory) {

if (userStory == null || !userStory.isCompleted()) {

return null;

}

return new Bug(id, name, estimate, userStory);

}

@Override

public String toString() {

return "[Bug " + getId() + "] " + userStory.getName() + ": " + getName();

}

}

**Sprint.java**

import java.util.ArrayList;

import java.util.List;

public class Sprint {

private int capacity;

private int ticketLimit;

private List<Ticket> tickets;

public Sprint(int capacity, int ticketLimit) {

this.capacity = capacity;

this.ticketLimit = ticketLimit;

this.tickets = new ArrayList<>();

}

public boolean addUserStory(UserStory userStory) {

return addTicket(userStory);

}

public boolean addBug(Bug bug) {

return addTicket(bug);

}

private boolean addTicket(Ticket ticket) {

if (ticket == null || ticket.isCompleted()) return false;

if (tickets.size() >= ticketLimit) return false;

if (getTotalEstimate() + ticket.getEstimate() > capacity) return false;

tickets.add(ticket);

return true;

}

public Ticket[] getTickets() {

return tickets.toArray(new Ticket[0]);

}

public int getTotalEstimate() {

int sum = 0;

for (Ticket t : tickets) sum += t.getEstimate();

return sum;

}

}

**Результат тестування (Main.java):**

public class Main {

public static void main(String[] args) {

UserStory regForm = new UserStory(1, "Форма реєстрації", 5, null);

UserStory emailVer = new UserStory(2, "Підтвердження email", 3, new UserStory[]{regForm});

regForm.complete(); // завершиться

emailVer.complete(); // не завершиться

System.out.println(regForm + " isCompleted: " + regForm.isCompleted());

System.out.println(emailVer + " isCompleted: " + emailVer.isCompleted());

emailVer.complete();

System.out.println(emailVer + " isCompleted: " + emailVer.isCompleted());

Bug bug1 = Bug.createBug(10, "Додати повторюваний пароль", 2, regForm);

Bug bug2 = Bug.createBug(11, "Email некоректний", 1, emailVer);

Sprint sprint = new Sprint(10, 3);

System.out.println("Add regForm: " + sprint.addUserStory(regForm)); // true

System.out.println("Add emailVer: " + sprint.addUserStory(emailVer)); // true

System.out.println("Add bug1: " + sprint.addBug(bug1)); // true

Bug extraBug = Bug.createBug(12, "Ще один баг", 5, regForm);

System.out.println("Add extraBug: " + sprint.addBug(extraBug)); // false

for (Ticket t : sprint.getTickets()) {

System.out.println(t.toString());

}

System.out.println("Total estimate: " + sprint.getTotalEstimate());

}

}

**Вивід програми:**

[US 1] Форма реєстрації isCompleted: true

[US 2] Підтвердження email isCompleted: false

[US 2] Підтвердження email isCompleted: false

Add regForm: true

Add emailVer: true

Add bug1: true

Add extraBug: false

[US 1] Форма реєстрації

[US 2] Підтвердження email

[Bug 10] Форма реєстрації: Додати повторюваний пароль

Total estimate: 10

**Посилання на github із кодом:**  
<https://github.com/artembombardir/OOP-KB232-Artem-Plotnikov/tree/main/lab03>