Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



Звіт

# до лабораторної роботи № 3

# з дисципліни

*Прикладне програмування*

# на тему:

## “Гра «Битва дроїдів»”

Виконала: студентка групи ОІ-24

Окілка Марта

Прийняв: асистент каф. АСУ

Чорненький В. Я.

**Львів – 2024**

***Лабораторна робота № 3***

**Тема роботи: “Гра «Битва дроїдів»”**

**Мета роботи:** розробка консольної гри «Битва дроїдів» з використанням об’єктно-орієнтованого програмування на мові Java. Завдання передбачає створення базових класів дроїдів, реалізацію різних видів бою та додавання функціоналу для взаємодії з користувачем через консольне меню. Це дозволить закріпити знання з організації класів, роботи з ієрархією класів, а також опрацювати методи запису та відтворення інформації з файлів.

**Мінімальний набір вимог до програми:**

**1.** Створіть базовий клас Droid, від якого будуть походити інші підкласи (види дроїдів), які будуть відрізнятися різними характеристиками. Мінімальний набір характеристик: name, health, damage.

**2.** Додайте можливість різних видів бою: 1 на 1, або команда на команду.

**3.** Класи потрібно грамотно розкласти по пакетах.

**4.** У програмі має бути консольне меню. Мінімальний набір команд:

* створити дроїда (обраного виду);
* показати список створених дроїдів;
* запустити бій 1 на 1 (вибрати дроїдів, які будуть змагатися);
* запустити бій команда на команду (сформувати команди суперників з дроїдів, яких ви створили у першому пункті);
* записати проведений бій у файл;
* відтворити проведений бій зі збереженого файлу;
* вийти з програми.

**Додаткове завдання:**

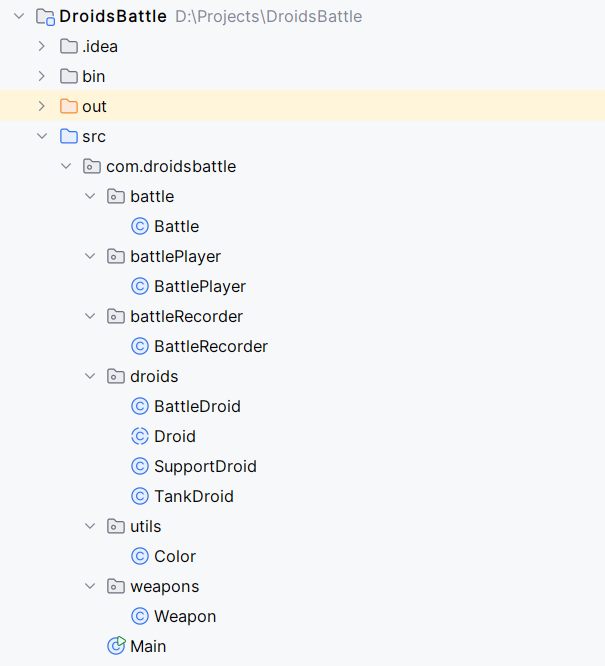
Використайте свою фантазію, щоб гра вийшла цікавою для Вас і для глядача. Жодних обмежень немає.

**Приклади того, що можна доробити:**

* Дроїди можуть мати специфічні для свого виду характеристики – рівень енергії, регенерація, точність, здатність відновлювати здоров’я членів команди тощо.
* Деякі дроїди можуть мати додаткову зброю (утворюють окрему ієрархію класів).
* Звернути увагу на збалансованість гри (відсутність або низький рівень деяких характеристик дроїдів певного виду компенсуються наявністю або вищим рівнем інших).
* Можна створити арени різних видів, що міняють характеристики бою (наприклад, зменшують точність).
* Бій 1 на 1 може бути реалізований як бій між командами з 1 дроїда або як окремий різновид бою зі своїми правилами.
* При бої команда на команду можна застосовувати різні стратегії бою, наприклад, вибирати випадкового дроїда зі своєї команди і команди суперника, дроїдів по черзі, або атакувати дроїда з найменшим здоров’ям.
* Візуалізувати процес бою, використовуючи псевдографіку, кольори та інші можливості змін вигляду тексту у консольному вікні.

**Виконання завдань лабораторної роботи:**

**1.** Структура проєкту:



**2.** Вміст файлу Battle.java:

package com.droidsbattle.battle;  
  
import com.droidsbattle.droids.Droid;  
import com.droidsbattle.battleRecorder.BattleRecorder;  
import java.util.List;  
  
public class Battle {  
 *// Об'єкт для запису бою* private static BattleRecorder *recorder* = new BattleRecorder();  
  
 *// Метод для бою 1 на 1* public static void oneOnOne(Droid droid1, Droid droid2) {  
 *recorder*.startRecording("Бій 1 на 1"); *// Починаємо запис бою 1 на 1* System.*out*.println("Починається бій між " + droid1.getName() + " і " + droid2.getName());  
 *// Почергові атаки* while (droid1.isAlive() && droid2.isAlive()) {  
 droid1.attack(droid2);  
 if (droid2.isAlive()) {  
 droid2.attack(droid1);  
 }  
 }  
 if (droid1.isAlive()) {  
 System.*out*.println(droid1.getName() + " перемагає!");  
 *recorder*.recordResult(droid1.getName() + " перемагає!");  
 } else {  
 System.*out*.println(droid2.getName() + " перемагає!");  
 *recorder*.recordResult(droid2.getName() + " перемагає!");  
 }  
 }  
  
 *// Метод для командного бою* public static void teamBattle(List<Droid> team1, List<Droid> team2) {  
 *recorder*.startRecording("Командний бій"); *// Починаємо запис командного бою* System.*out*.println("Починається командний бій!");  
 while (*teamIsAlive*(team1) && *teamIsAlive*(team2)) {  
 Droid attacker = *getNextAliveDroid*(team1);  
 Droid defender = *getNextAliveDroid*(team2);  
  
 if (attacker != null && defender != null) {  
 attacker.attack(defender);  
 }  
  
 *// Змінюємо команди місцями для наступного ходу* List<Droid> temp = team1;  
 team1 = team2;  
 team2 = temp;  
 }  
  
 if (*teamIsAlive*(team1)) {  
 System.*out*.println("Команда 1 перемагає!");  
 *recorder*.recordResult("Команда 1 перемагає!");  
 } else {  
 System.*out*.println("Команда 2 перемагає!");  
 *recorder*.recordResult("Команда 2 перемагає!");  
 }  
 }  
  
 *// Метод для перевірки, чи жива команда* private static boolean teamIsAlive(List<Droid> team) {  
 for (Droid d : team) {  
 if (d.isAlive()) {  
 return true;  
 }  
 }  
 return false;  
 }  
  
 *// Метод для отримання наступного живого дроїда* private static Droid getNextAliveDroid(List<Droid> team) {  
 for (Droid d : team) {  
 if (d.isAlive()) {  
 return d;  
 }  
 }  
 return null;  
 }  
  
 *// Метод для отримання об'єкта для запису бою* public static BattleRecorder getRecorder() {  
 return *recorder*;  
 }  
}

**3.** Вміст файлу BattlePlayer.java:

package com.droidsbattle.battlePlayer;  
  
import java.io.BufferedReader;  
import java.io.FileReader;  
import java.io.IOException;  
  
public class BattlePlayer {  
  
 *// Метод для відтворення бою із файлу* public void playBattleFromFile(String filename) {  
 try (BufferedReader reader = new BufferedReader(new FileReader(filename))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 System.*out*.println(line); *// Виводимо кожен рядок із файлу на консоль* try {  
 Thread.*sleep*(500); *// Затримка для читання* } catch (InterruptedException e) {  
 Thread.*currentThread*().interrupt();  
 }  
 }  
 } catch (IOException e) {  
 System.*err*.println("Помилка під час читання файлу: " + e.getMessage());  
 }  
 }  
}

**4.** Вміст файлу BattleRecorder.java:

package com.droidsbattle.battleRecorder;  
  
import com.droidsbattle.droids.Droid;  
  
import java.io.FileWriter;  
import java.io.IOException;  
  
public class BattleRecorder {  
 private StringBuilder battleLog;  
  
 *// Конструктор класу* public BattleRecorder() {  
 battleLog = new StringBuilder();  
 }  
  
 *// Метод для початку запису бою* public void startRecording(String battleType) {  
 battleLog.append(battleType).append(":\n");  
 }  
  
 *// Метод для запису ходу бою* public void recordTurn(Droid attacker, Droid defender, int damage) {  
 battleLog.append(attacker.getClass().getSimpleName())  
 .append(" ")  
 .append(attacker.getName())  
 .append(" атакує ")  
 .append(defender.getClass().getSimpleName())  
 .append(" ")  
 .append(defender.getName())  
 .append(" і наносить ")  
 .append(damage)  
 .append(" шкоди. Здоров'я ")  
 .append(defender.getName())  
 .append(": ")  
 .append(defender.getHealth())  
 .append(".\n");  
 }  
  
 *// Метод для запису лікування* public void recordHeal(Droid healer, Droid target, int healAmount) {  
 battleLog.append(healer.getClass().getSimpleName())  
 .append(" ")  
 .append(healer.getName())  
 .append(" лікує ")  
 .append(target.getClass().getSimpleName())  
 .append(" ")  
 .append(target.getName())  
 .append(" на ")  
 .append(healAmount)  
 .append(" одиниць здоров'я. Здоров'я ")  
 .append(target.getName())  
 .append(": ")  
 .append(target.getHealth())  
 .append(".\n");  
 }  
  
 *// Метод для запису результату бою* public void recordResult(String result) {  
 battleLog.append(result).append("\n---\n");  
 }  
  
 *// Метод для збереження бою в файл* public void saveToFile(String filename) {  
 try (FileWriter writer = new FileWriter(filename, true)) { *// true - для дозапису у файл* writer.write(battleLog.toString());  
 } catch (IOException e) {  
 System.*err*.println("Помилка під час запису бою у файл: " + e.getMessage());  
 }  
 }  
}

**5.** Вміст файлу BattleDroid.java:

package com.droidsbattle.droids;  
  
import com.droidsbattle.battle.Battle;  
import com.droidsbattle.utils.Color;  
  
public class BattleDroid extends Droid {  
  
 *// Конструктор класу BattleDroid* public BattleDroid(String name) {  
 super(name, 100, 20);  
 }  
  
 @Override  
 public void attack(Droid opponent) {  
 int totalDamage = getTotalDamage();  
 System.*out*.println(Color.*BLUE* + name + Color.*RESET* + " атакує " + Color.*RED* + opponent.getName() + Color.*RESET* + " і завдає " + Color.*YELLOW* + totalDamage + Color.*RESET* + " шкоди.");  
 opponent.takeDamage(totalDamage);  
 Battle.*getRecorder*().recordTurn(this, opponent, totalDamage);  
 }  
}

**6.** Вміст файлу Droid.java:

package com.droidsbattle.droids;  
  
import com.droidsbattle.weapons.Weapon;  
  
*// Абстрактний клас дроїда (базовий)*public abstract class Droid {  
 protected String name;  
 protected int health;  
 protected int maxDamage;  
 protected Weapon weapon;  
  
 *// Конструктор класу* public Droid(String name, int health, int maxDamage) {  
 this.name = name;  
 this.health = health;  
 this.maxDamage = maxDamage;  
 this.weapon = null;  
 }  
  
 *// Метод екіпірування зброї* public void equipWeapon(Weapon weapon) {  
 this.weapon = weapon;  
 System.*out*.println(name + " екіпірував зброю: " + weapon.getName());  
 }  
  
 *// Отримати загальну шкоду (базову + бонус від зброї)* public int getTotalDamage() {  
 int totalDamage = (int) (Math.*random*() \* (maxDamage + 1)); *// Випадкове число в межах від 0 до maxDamage* if (weapon != null) {  
 totalDamage += weapon.getDamageBonus();  
 }  
 return totalDamage;  
 }  
  
 *// Отримати ім'я дроїда* public String getName() {  
 return name;  
 }  
  
 *// Отримати поточне здоров'я дроїда* public int getHealth() {  
 return health;  
 }  
  
 *// Завдати шкоди дроїду* public void takeDamage(int amount) {  
 health -= amount;  
 if (health < 0) health = 0;  
 }  
  
 *// Метод атаки, що визначається у підкласах* public abstract void attack(Droid opponent);  
  
 *// Перевірка, чи живий дроїд* public boolean isAlive() {  
 return health > 0;  
 }  
}

**7.** Вміст файлу SupportDroid.java:

package com.droidsbattle.droids;  
  
import com.droidsbattle.battle.Battle;  
import com.droidsbattle.utils.Color;  
  
public class SupportDroid extends Droid {  
  
 private int healAmount; *// Поле для зберігання кількості здоров'я, яке може відновити дроїд  
  
 // Конструктор класу SupportDroid* public SupportDroid(String name) {  
 super(name, 80, 10);  
 this.healAmount = 15;  
 }  
  
 @Override  
 public void attack(Droid opponent) {  
 int totalDamage = getTotalDamage();  
 System.*out*.println(Color.*CYAN* + name + Color.*RESET* + " атакує " + Color.*RED* + opponent.getName() + Color.*RESET* + " і завдає " + Color.*YELLOW* + totalDamage + Color.*RESET* + " шкоди.");  
 opponent.takeDamage(totalDamage);  
 Battle.*getRecorder*().recordTurn(this, opponent, totalDamage);  
 }  
  
 *// Метод лікування союзника* public void heal(Droid ally) {  
 System.*out*.println(Color.*CYAN* + name + Color.*RESET* + " лікує " + Color.*GREEN* + ally.getName() + Color.*RESET* + " на " + Color.*GREEN* + healAmount + Color.*RESET* + " одиниць здоров'я.");  
 ally.health += healAmount;  
 Battle.*getRecorder*().recordHeal(this, ally, healAmount);  
 }  
}

**8.** Вміст файлу TankDroid.java:

package com.droidsbattle.droids;  
  
import com.droidsbattle.battle.Battle;  
import com.droidsbattle.utils.Color;  
  
public class TankDroid extends Droid {  
  
 *// Конструктор класу TankDroid* public TankDroid(String name) {  
 super(name, 150, 15);  
 }  
  
 @Override  
 public void attack(Droid opponent) {  
 int totalDamage = getTotalDamage();  
 System.*out*.println(Color.*PURPLE* + name + Color.*RESET* + " завдає потужного удару по " + Color.*RED* + opponent.getName() + Color.*RESET* + ", завдаючи " + Color.*YELLOW* + totalDamage + Color.*RESET* + " шкоди.");  
 opponent.takeDamage(totalDamage);  
 Battle.*getRecorder*().recordTurn(this, opponent, totalDamage);  
 }  
}

**9.** Вміст файлу Color.java:

package com.droidsbattle.utils;  
  
public class Color {  
 public static final String *RESET* = "\u001B[0m";  
  
 public static final String *BLACK* = "\u001B[30m";  
 public static final String *RED* = "\u001B[31m";  
 public static final String *GREEN* = "\u001B[32m";  
 public static final String *YELLOW* = "\u001B[33m";  
 public static final String *BLUE* = "\u001B[34m";  
 public static final String *PURPLE* = "\u001B[35m";  
 public static final String *CYAN* = "\u001B[36m";  
 public static final String *WHITE* = "\u001B[37m";  
}

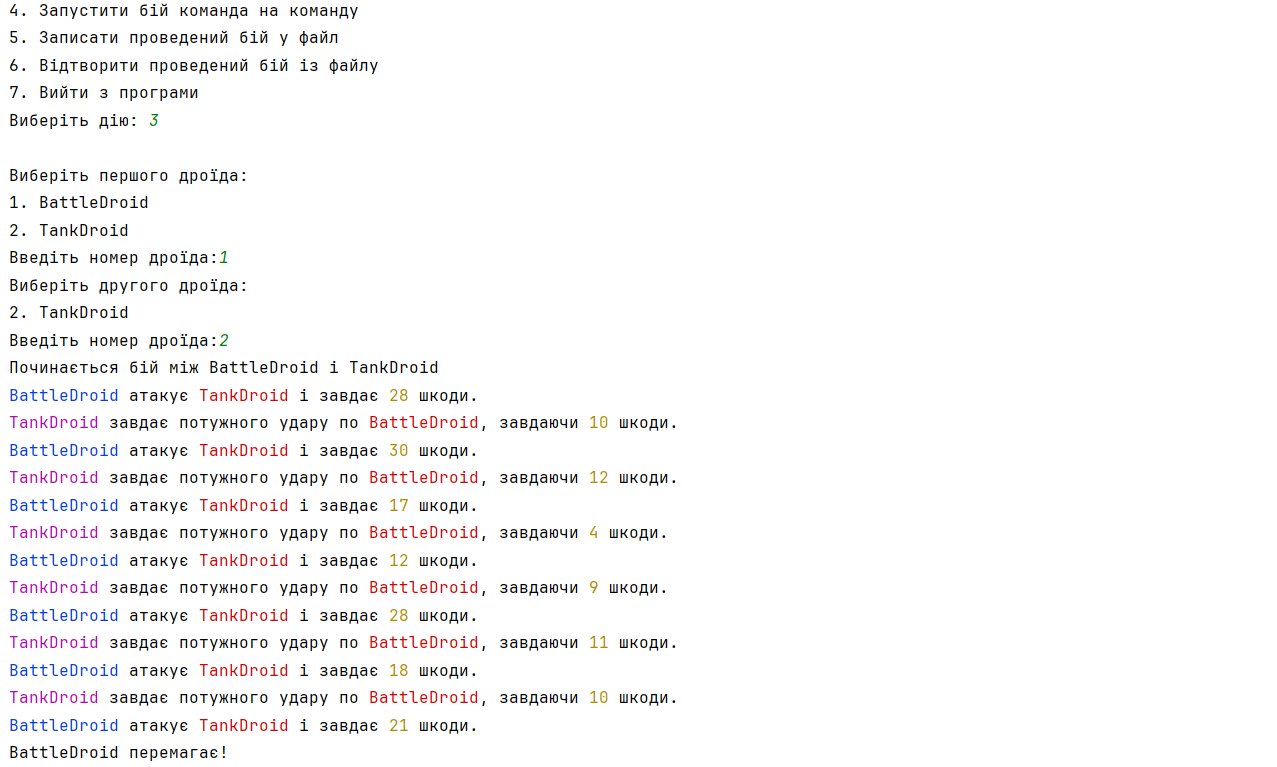
**10.** Вміст файлу Weapon.java:

package com.droidsbattle.weapons;  
  
public class Weapon {  
 private String name;  
 private int damageBonus;  
  
 *// Конструктор класу* public Weapon(String name, int damageBonus) {  
 this.name = name;  
 this.damageBonus = damageBonus;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public int getDamageBonus() {  
 return damageBonus;  
 }  
}

**11.** Вміст файлу Main.java:

package com.droidsbattle;  
  
import com.droidsbattle.droids.\*;  
import com.droidsbattle.weapons.\*;  
import com.droidsbattle.battle.Battle;  
import com.droidsbattle.battlePlayer.BattlePlayer;  
  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
  
public class Main {  
  
 private static List<Droid> *droidList* = new ArrayList<>(); *// список для збереження всіх дроїдів* private static Scanner *scanner* = new Scanner(System.*in*);  
  
 public static void main(String[] args) {  
 boolean exit = false;  
 while (!exit) {  
 *printMenu*();  
 int choice = *getUserChoice*();  
 switch (choice) {  
 case 1:  
 *createDroid*();  
 break;  
 case 2:  
 *showDroids*();  
 break;  
 case 3:  
 *startOneOnOneBattle*();  
 break;  
 case 4:  
 *startTeamBattle*();  
 break;  
 case 5:  
 System.*out*.print("Введіть ім'я файлу для запису бою (з розширенням .txt): ");  
 String filename = *scanner*.next();  
 Battle.*getRecorder*().saveToFile(filename); *// Запис бою у файл* System.*out*.println("Бій записаний у файл " + filename);  
 break;  
 case 6:  
 System.*out*.print("Введіть ім'я файлу для відтворення бою: ");  
 String nameOfFile = *scanner*.next();  
 BattlePlayer player = new BattlePlayer();  
 player.playBattleFromFile(nameOfFile); *// Відтворення бою із файлу* break;  
 case 7:  
 exit = true;  
 System.*out*.println("Вихід із програми.");  
 break;  
 default:  
 System.*out*.println("Неправильний вибір. Спробуйте ще раз.");  
 }  
 }  
 }  
  
 *// Метод виведення меню* private static void printMenu() {  
 System.*out*.println("\n--- Меню ---");  
 System.*out*.println("1. Створити дроїда");  
 System.*out*.println("2. Показати список дроїдів");  
 System.*out*.println("3. Запустити бій 1 на 1");  
 System.*out*.println("4. Запустити бій команда на команду");  
 System.*out*.println("5. Записати проведений бій у файл");  
 System.*out*.println("6. Відтворити проведений бій із файлу");  
 System.*out*.println("7. Вийти з програми");  
 System.*out*.print("Виберіть дію: ");  
 }  
  
 *// Метод отримання вибору користувача* private static int getUserChoice() {  
 int choice = -1;  
 if (*scanner*.hasNextInt()) {  
 choice = *scanner*.nextInt();  
 } else {  
 *scanner*.next(); *// Очистити неправильне введення* }  
 return choice;  
 }  
  
 *// Метод створення дроїду* private static void createDroid() {  
 System.*out*.println("\nВиберіть тип дроїда:");  
 System.*out*.println("1. BattleDroid");  
 System.*out*.println("2. SupportDroid");  
 System.*out*.println("3. TankDroid");  
 System.*out*.print("Введіть номер дроїду: ");  
 int type = *getUserChoice*();  
 *scanner*.nextLine(); *// Очистити перехід на новий рядок* System.*out*.print("Введіть ім'я дроїда: ");  
 String name = *scanner*.nextLine();  
  
 Droid newDroid = null;  
 switch (type) {  
 case 1:  
 newDroid = new BattleDroid(name);  
 break;  
 case 2:  
 newDroid = new SupportDroid(name);  
 break;  
 case 3:  
 newDroid = new TankDroid(name);  
 break;  
 default:  
 System.*out*.println("Неправильний вибір типу дроїду.");  
 return;  
 }  
  
 *// Можливість екіпірувати зброю* System.*out*.print("Хочете екіпірувати дроїда зброєю? (так/ні): ");  
 String equipChoice = *scanner*.nextLine();  
 if (equipChoice.equalsIgnoreCase("так")) {  
 *equipWeapon*(newDroid);  
 }  
  
 *droidList*.add(newDroid);  
 System.*out*.println("Дроїд " + newDroid.getName() + " створений та доданий до списку.");  
 }  
  
 *// Метод екіпірування зброї* private static void equipWeapon(Droid droid) {  
 System.*out*.println("Виберіть зброю: ");  
 System.*out*.println("1. Лазерний меч (+15 до шкоди)");  
 System.*out*.println("2. Бластер (+10 до шкоди)");  
 System.*out*.println("3. Плазмова гармата (+20 до шкоди)");  
 System.*out*.print("Введіть номер зброї: ");  
 int weaponChoice = *getUserChoice*();  
 *scanner*.nextLine(); *// Очистити перехід на новий рядок* Weapon weapon = null;  
 switch (weaponChoice) {  
 case 1:  
 weapon = new Weapon("Лазерний меч", 15);  
 break;  
 case 2:  
 weapon = new Weapon("Бластер", 10);  
 break;  
 case 3:  
 weapon = new Weapon("Плазмова гармата", 20);  
 break;  
 default:  
 System.*out*.println("Неправильний вибір зброї.");  
 return;  
 }  
 droid.equipWeapon(weapon);  
 }  
  
 *// Метод для відображення списку дроїдів* private static void showDroids() {  
 if (*droidList*.isEmpty()) {  
 System.*out*.println("Список дроїдів порожній.");  
 return;  
 }  
 System.*out*.println("\n--- Список дроїдів ---");  
 for (int i = 0; i < *droidList*.size(); i++) {  
 Droid d = *droidList*.get(i);  
 System.*out*.println((i + 1) + ". " + d.getName() + " - Здоров'я: " + d.getHealth());  
 }  
 }  
  
 *// Метод для запуску бою 1 на 1* private static void startOneOnOneBattle() {  
 if (*droidList*.size() < 2) {  
 System.*out*.println("Недостатньо дроїдів для бою.");  
 return;  
 }  
 System.*out*.println("\nВиберіть першого дроїда:");  
 for (int i = 0; i < *droidList*.size(); i++) {  
 System.*out*.println((i + 1) + ". " + *droidList*.get(i).getName());  
 }  
 System.*out*.print("Введіть номер дроїда:");  
 int firstChoice = *getUserChoice*() - 1;  
  
 System.*out*.println("Виберіть другого дроїда:");  
 for (int i = 0; i < *droidList*.size(); i++) {  
 if (i != firstChoice) {  
 System.*out*.println((i + 1) + ". " + *droidList*.get(i).getName());  
 }  
 }  
 System.*out*.print("Введіть номер дроїда:");  
 int secondChoice = *getUserChoice*() - 1;  
  
 Droid droid1 = *droidList*.get(firstChoice);  
 Droid droid2 = *droidList*.get(secondChoice);  
 Battle.*oneOnOne*(droid1, droid2);  
 }  
  
 *// Метод для запуску командного бою* private static void startTeamBattle() {  
 if (*droidList*.size() < 4) {  
 System.*out*.println("Недостатньо дроїдів для командного бою (потрібно мінімум 4).");  
 return;  
 }  
 *// Формуємо команди* List<Droid> team1 = new ArrayList<>();  
 List<Droid> team2 = new ArrayList<>();  
 System.*out*.println("Формування команд.");  
 for (int i = 0; i < *droidList*.size(); i++) {  
 System.*out*.println((i + 1) + ". " + *droidList*.get(i).getName());  
 }  
 *scanner*.nextLine(); *// Очистити перехід на новий рядок* System.*out*.println("Додайте дроїдів до команди 1 (введіть номери через кому):");  
 String team1Input = *scanner*.nextLine();  
 System.*out*.println("Додайте дроїдів до команди 2 (введіть номери через кому):");  
 String team2Input = *scanner*.nextLine();  
  
 *// Парсинг введення та заповнення команд* try {  
 for (String s : team1Input.split(",")) {  
 int idx = Integer.*parseInt*(s.trim()) - 1;  
 team1.add(*droidList*.get(idx));  
 }  
 for (String s : team2Input.split(",")) {  
 int idx = Integer.*parseInt*(s.trim()) - 1;  
 team2.add(*droidList*.get(idx));  
 }  
 } catch (Exception e) {  
 System.*out*.println("Помилка для формування команд.");  
 return;  
 }  
  
 Battle.*teamBattle*(team1, team2);  
 }  
}

**11.** Зразок виконання програми:



**12.** Посилання на публічний GitHub репозиторій:

https://github.com/OkayMarta/LR3-DroidsBattle

**Висновок**

У цій лабораторній роботі я навчилася створювати ієрархію класів для різних об'єктів (дроїдів), організовувати їх взаємодію через методи класів, а також працювати з файлами для запису та відтворення подій. Я закріпила навички об'єктно-орієнтованого програмування на практиці та розширила свої знання щодо створення консольних ігор.