Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**ЗВІТ**

Про виконання лабораторної роботи **№ 3**

**«Гра “Битва дроїдів”»**

з дисципліни **«Прикладне програмування»**

**Лектор:**

доц. кафедри АСУ

Скорохода О.В.

**Виконав:**

студ. групи КН-203

Репей А.В.

**Прийняв:**

асис. кафедри АСУ

Мельник Р. В.

«\_\_\_» \_\_\_\_\_\_ 2022 р.

∑ = \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_

Львів – 2022

**Тема:** Гра “Битва дроїдів”

**Мета:** Написати свою реалізацію консольної гри “Битва дроїдів”

**Посилання на репозиторій Github**: https://github.com/temerep/labs\_ap/tree/main/lab3

**Хід виконання лабораторної роботи:**

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.

**Вихідний код програми**

**App.java**

package com.repei.droidbattle;

import java.io.BufferedReader;

import java.io.FileReader;

import java.util.ArrayList;

import java.util.List;

import java.util.Random;

import java.util.Scanner;

import com.repei.droidbattle.droid.BaseDroid;

import com.repei.droidbattle.droid.Bowman;

import com.repei.droidbattle.droid.Knight;

import com.repei.droidbattle.droid.Wizard;

public class App {

public static void main(String[] args) throws InterruptedException {

System.out.print("\033\143");

Scanner sc = new Scanner(System.in);

boolean menuChecker = false;

int select;

List<BaseDroid> units = new ArrayList<>();

List<String> names = new ArrayList<>();

BufferedReader reader;

try {

reader = new BufferedReader(new FileReader("names.txt"));

String line = reader.readLine();

while (line != null) {

names.add(line);

line = reader.readLine();

}

reader.close();

} catch (Exception e) {

}

do {

welcome();

select = sc.nextInt();

Random random = new Random();

String name;

int unitCount;

switch (select) {

case 1:

menuChecker = false;

System.out.print("\033\143");

System.out.print(Font.BOLD + Font.WHITE + "👤 Select warrior:\n\n(1) Bowman\t" + Font.YELLOW

+ "(HP: 120 DAM: 25)" + Font.WHITE + "\n(2) Knight\t" + Font.YELLOW + "(HP: 240 DAM: 35)\n" + Font.WHITE

+ "(3) Wizard\t" + Font.YELLOW + "(HP: 180 DAM: 45)\n\n=> " + Font.RESET);

select = sc.nextInt();

name = names.get(random.nextInt(names.size()));

units.add(select == 1 ? new Bowman(name) : select == 2 ? new Knight(name) : new Wizard(name));

name = names.get(random.nextInt(names.size()));

units.add(select == 1 ? new Bowman(name) : select == 2 ? new Knight(name) : new Wizard(name));

new DroidBattle().battle(1, units);

break;

case 2:

menuChecker = false;

do {

System.out.print("\033\143");

System.out.print(Font.BOLD + Font.WHITE + "👥 Number of warriors in the team: " + Font.RESET);

unitCount = sc.nextInt();

} while (unitCount < 1 || unitCount > 15);

System.out.print(Font.BOLD + Font.WHITE + "\n👤 Select warriors:\n\n(1) Bowman\t" + Font.YELLOW

+ "(HP: 120 DAM: 25)" + Font.WHITE + "\n(2) Knight\t" + Font.YELLOW + "(HP: 240 DAM: 35)\n" + Font.WHITE

+ "(3) Wizard\t" + Font.YELLOW + "(HP: 180 DAM: 45)\n " + Font.RESET);

for (int i = 0; i < unitCount; i++) {

do {

System.out.printf(Font.BOLD + Font.YELLOW + "\n#%d" + Font.WHITE + " warrior => " + Font.RESET, i + 1);

select = sc.nextInt();

name = names.get(random.nextInt(names.size()));

units.add(select == 1 ? new Bowman(name) : select == 2 ? new Knight(name) : new Wizard(name));

name = names.get(random.nextInt(names.size()));

units.add(select == 1 ? new Bowman(name) : select == 2 ? new Knight(name) : new Wizard(name));

} while (select < 1 || select > 3);

}

new DroidBattle().battle(unitCount, units);

break;

case 3:

menuChecker = false;

System.out.print("\033\143");

OutBattle out = new OutBattle();

break;

case 4:

menuChecker = false;

System.out.print("\033\143");

break;

default:

System.out.print("\033\143");

menuChecker = true;

}

} while (menuChecker);

}

public static void welcome() {

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛⬛⬛⬛⬛⬛⬛⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️");

System.out.println(Font.BOLD + Font.WHITE

+ "⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛🟫🟫🟫🟫🟫🟫🟫🟫⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️ \_\_ \_\_ \_ \_\_\_\_\_\_\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛🟫🏼🏼🏼🏼🏼🏼🏼🟫🟫⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️ | \\/ | | | \_\_\_\_|\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛🏼🏼🏼🏼🏼🏼🏼🏼🏼🟫⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️ | \\ / | \_\_ \_ \_\_| | |\_\_ \_\_\_ \_ \_\_ \_\_\_ \_\_\_\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛🏼🏼⬛🏼🏼🏼⬛🏼🏼🟫⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️ | |\\/| |/ \_` |/ \_` | \_\_/ \_ \\| '\_\_/ \_\_/ \_ \\\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛🏼🏼🏼🏼🏼🏼🏼🏼🏼🏼⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️ | | | | (\_| | (\_| | | | (\_) | | | (\_| \_\_/\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛🏼🏼🏼⬛⬛⬛🏼🏼🏼🏼⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️ |\_| |\_|\\\_\_,\_|\\\_\_,\_|\_| \\\_\_\_/|\_| \\\_\_\_\\\_\_\_| \_ \_. .\_ \_ \_ \t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️🟧⬜️⬜️⬛⬛🏼🏼🏼🏼🏼🏼🏼🏼⬛⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t\t\t\t\t\t (\_| (\_| | | | (/\_ \t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️🟧⬜️⬜️⬜️⬛⬛⬛⬛⬛⬛⬛⬛⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\tBattle terminal-game\t\t\t \_| \t⬜️⬜️⬜️⬜️⬜️"

+ Font.RESET);

System.out.println("⬜️⬜️⬜️⬜️⬜️🟧⬜️⬜️⬛🏽🟦🟨🟨🟨🏽🏽🏽⬛⬜️⬜️⬜️🟧⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t" + Font.BLACK

+ "Author: @temerep\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜" + Font.RESET);

System.out.println("⬜️⬜️⬜️🟧⬜️⬜️🟧⬛🏽⬛🟨🟦🟨🟨🟨🟨⬛🏽⬛⬜️🟧⬜⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t\t\t\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️⬜️⬜️🟧🟧🟥⬛🏼⬛🟨🟨🏽🟦🟨🟨⬛🏼⬛🟧🟧⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t\t\t\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️🟧⬜️🟧🟥⬛🏼⬛⬛🟨🟨🟨🟨🟦🟦⬛🏼⬛🟥🟧🟧⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t" + Font.BOLD + Font.YELLOW

+ "Menu:\t\t\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️⬜️🟧🟥🟥⬛⬛⬜️⬛🟧🟧🟦🟧🟧🟧⬛⬛🟥🟧🟧⬜️🟧⬜️⬜️⬜️⬜️⬜️⬜️\t\t\t\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️⬜️⬜️🟧⬜️🟧🟧🟥⬛⬜️🟦🟦🟦🟦🟦⬛🟥🟥🟧🟧⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t" + "1️⃣ Single Battle (1x1)"

+ "\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️⬜️🟧⬜️⬜️🟧⬜️⬜️⬛🏽🟦⬛⬛🏽🟦⬛🟧🟧⬜️⬜️🟧⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t" + "2️⃣ Big Battle (Team x Team)"

+ "\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println(

"⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬛⬛⬛⬜️⬜️⬛⬛⬛🟧⬜️🟧⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t" + "3️⃣ Play Game From File" + "\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️🟧⬜️⬜️🟧🟧⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t" + "4️⃣ Exit" + "\t\t\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t\t\t\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.println("⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t\t\t\t\t\t\t\t\t⬜️⬜️⬜️⬜️⬜️");

System.out.print("⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️⬜️\t=> " + Font.RESET);

}

}

**DroidBattle.java**

package com.repei.droidbattle;

import java.io.File;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.List;

import com.repei.droidbattle.controller.BattleArena;

import com.repei.droidbattle.droid.\*;

public class DroidBattle {

public File log;

PrintWriter pw;

public void battle(int cnt, List<BaseDroid> units) throws InterruptedException {

int logNum = 1;

try{

log = new File("battle" + Integer.toString(logNum) + ".txt");

while (log.exists()) {

++logNum;

log = new File("battle" + Integer.toString(logNum) + ".txt");

}

log.createNewFile();

pw = new PrintWriter(log);

} catch (Exception e) {

System.out.println(e);

}

List<BaseDroid> me = new ArrayList<>();

List<BaseDroid> enemies = new ArrayList<>();

for (int i = 0; i < cnt; i++) {

me.add(units.get(i));

enemies.add(units.get(cnt + i));

}

System.out.print("\033\143");

BattleArena arena = new BattleArena(me, enemies, cnt, pw);

BaseDroid winner = arena.startFight();

System.out.println("\n" + Font.BOLD + Font.GREEN

+ "╔═══════════════════════════════════════════════════════════════════════╗" + Font.RESET);

pw.println("\n" + Font.BOLD + Font.GREEN

+ "╔═══════════════════════════════════════════════════════════════════════╗" + Font.RESET);

System.out.println("\n" + Font.BOLD + Font.GREEN + "║\t\t " + Font.GREETING + " The winner is "

+ winner.getName() + "'s Team " + Font.GREETING + "\t\t\t║" + Font.RESET);

pw.println("\n" + Font.BOLD + Font.GREEN + "║\t\t " + Font.GREETING + " The winner is "

+ winner.getName() + "'s Team " + Font.GREETING + "\t\t\t║" + Font.RESET);

System.out.println("\n" + Font.GREEN + "║\t\t\tBattle saved to \"" + log.getName() + "\"\t\t\t║" + Font.RESET);

System.out.println("\n" + Font.BOLD + Font.GREEN

+ "╚═══════════════════════════════════════════════════════════════════════╝" + Font.RESET);

pw.println("\n" + Font.BOLD + Font.GREEN

+ "╚═══════════════════════════════════════════════════════════════════════╝" + Font.RESET);

pw.close();

}

}

**BattleArena.java**

package com.repei.droidbattle.controller;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.List;

import java.util.Random;

import java.util.concurrent.TimeUnit;

import com.repei.droidbattle.Font;

import com.repei.droidbattle.droid.BaseDroid;

public class BattleArena {

private final List<BaseDroid> me;

private final List<BaseDroid> enemies;

private BaseDroid attacker;

private BaseDroid defender;

private int teamCount = 0;

private int currentRound = 0;

private PrintWriter pw;

public BattleArena(List<BaseDroid> me, List<BaseDroid> enemies, int cnt, PrintWriter pw) {

this.me = new ArrayList<>(me);

this.enemies = new ArrayList<>(enemies);

this.teamCount = cnt;

this.pw = pw;

}

public boolean isAlive(List<BaseDroid> team) {

int sum = 0;

for (int i = 0; i < team.size(); i++) {

sum += team.get(i).getHealth();

}

return sum > 0;

}

public BaseDroid startFight() throws InterruptedException {

do {

prepareRound();

int actualDamage = doFight();

printRoundInfo(actualDamage, attacker, defender);

TimeUnit.MILLISECONDS.sleep(100);

} while (isAlive(me) && isAlive(enemies));

return attacker;

}

private void prepareRound() {

initFighters(teamCount);

currentRound++;

System.out.println(Font.BOLD

+ "\n◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎\n◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎\n"

+ Font.RESET);

System.out.println(Font.BOLD + Font.YELLOW + "Round " + currentRound + Font.RESET);

pw.println(Font.BOLD

+ "\n◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎\n◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎◼︎\n"

+ Font.RESET);

pw.println(Font.BOLD + Font.YELLOW + "Round " + currentRound + Font.RESET);

}

private int doFight() {

return defender.getHit(attacker.getDamage());

}

private void printRoundInfo(int actualDamage, BaseDroid attacker, BaseDroid defender) {

System.out.println("\n" + Font.RED + Font.BOLD + "ACTION: " + Font.BLOOD + defender.getWarrior() + " "

+ defender.getName() + Font.RESET

+ Font.RED + " get " + Font.BOLD + "-" + actualDamage + " " + Font.HEART + Font.RESET + Font.RED

+ " of damage from "

+ Font.GREEN + Font.BOLD + Font.AXE + attacker.getWarrior() + " " + attacker.getName() + Font.RESET + "\n");

pw.println("\n" + Font.RED + Font.BOLD + "ACTION: " + Font.BLOOD + defender.getWarrior() + " " + defender.getName()

+ Font.RESET

+ Font.RED + " get " + Font.BOLD + "-" + actualDamage + " " + Font.HEART + Font.RESET + Font.RED

+ " of damage from "

+ Font.GREEN + Font.BOLD + Font.AXE + attacker.getWarrior() + " " + attacker.getName() + Font.RESET + "\n");

System.out

.println(Font.BOLD + "-------------------------------------------------------------------------" + Font.RESET);

pw.println(Font.BOLD + "-------------------------------------------------------------------------" + Font.RESET);

System.out.println(Font.BOLD + Font.CYAN + "\t Your Team" + "\t\t|" + Font.RESET + Font.BOLD + Font.YELLOW

+ "\t\tRival Team\t\t|" + Font.RESET);

pw.println(Font.BOLD + Font.CYAN + "\t Your Team" + "\t\t|" + Font.RESET + Font.BOLD + Font.YELLOW

+ "\t\tRival Team\t\t|" + Font.RESET);

System.out.println(Font.BOLD + Font.CYAN + "Name" + "\t\t\tHealth\t|" + Font.RESET + Font.BOLD + Font.YELLOW

+ "\tName\t\t\tHealth\t|" + Font.RESET);

pw.println(Font.BOLD + Font.CYAN + "Name" + "\t\t\tHealth\t|" + Font.RESET + Font.BOLD + Font.YELLOW

+ "\tName\t\t\tHealth\t|" + Font.RESET);

System.out

.println(Font.BOLD + "-------------------------------------------------------------------------" + Font.RESET);

pw.println(Font.BOLD + "-------------------------------------------------------------------------" + Font.RESET);

for (int i = 0; i < teamCount; i++) {

System.out.println((attacker == me.get(i)

? Font.GREEN + Font.BOLD + Font.AXE

: defender == me.get(i)

? Font.RED + Font.BOLD + Font.BLOOD

: me.get(i).getHealth() == 0 ? Font.BLACK : Font.CYAN)

+ me.get(i) + Font.RESET + "\t"

+ (attacker == enemies.get(i)

? Font.GREEN + Font.BOLD + Font.AXE

: defender == enemies.get(i)

? Font.RED + Font.BOLD + Font.BLOOD

: enemies.get(i).getHealth() == 0 ? Font.BLACK : Font.YELLOW)

+ enemies.get(i) + Font.RESET);

pw.println((attacker == me.get(i)

? Font.GREEN + Font.BOLD + Font.AXE

: defender == me.get(i)

? Font.RED + Font.BOLD + Font.BLOOD

: me.get(i).getHealth() == 0 ? Font.BLACK : Font.CYAN)

+ me.get(i) + Font.RESET + "\t"

+ (attacker == enemies.get(i)

? Font.GREEN + Font.BOLD + Font.AXE

: defender == enemies.get(i)

? Font.RED + Font.BOLD + Font.BLOOD

: enemies.get(i).getHealth() == 0 ? Font.BLACK : Font.YELLOW)

+ enemies.get(i) + Font.RESET);

System.out.println(

Font.BOLD + "-------------------------------------------------------------------------" + Font.RESET);

pw.println(Font.BOLD + "-------------------------------------------------------------------------" + Font.RESET);

}

}

private void initFighters(int teamCount) {

Random random = new Random();

int rand;

if (random.nextBoolean()) {

do {

rand = random.nextInt(me.size());

attacker = me.get(rand);

} while (attacker.getHealth() == 0);

do {

rand = random.nextInt(enemies.size());

defender = enemies.get(rand);

} while (defender.getHealth() == 0);

} else {

do {

rand = random.nextInt(enemies.size());

attacker = enemies.get(rand);

} while (attacker.getHealth() == 0);

do {

rand = random.nextInt(me.size());

defender = me.get(rand);

} while (defender.getHealth() == 0);

}

}

}

**BaseDroid.java**

package com.repei.droidbattle.droid;

import java.util.Random;

import com.repei.droidbattle.Font;

public class BaseDroid {

private final String name;

private final int damage;

private String warrior = "BaseDroid";

private int energy;

private int health;

public BaseDroid(int health, int damage, String name, String warrior) {

this.health = health;

this.damage = damage;

this.name = name;

this.warrior = warrior;

this.energy = 0;

}

public int getDamage() {

if (energy >= 25) {

energy -= 25;

System.out.println("\n" + Font.BOLD + Font.BLUE + "ULT:\t" + Font.RESET + Font.BOLD + Font.BLUE + Font.CYCLONE + this.getWarrior() + " " + this.getName() + Font.RESET + Font.BLUE + " used x2 damage!" + Font.RESET);

return damage\*2;

}

return damage;

}

public String getWarrior() {

return warrior;

}

public int getEnergy() {

return energy;

}

public void setEnergy(int energy) {

this.energy = energy;

}

public int getHealth() {

return health;

}

public String getName() {

return name;

}

public boolean isAlive() {

return health > 0;

}

public int getHit(int damage) {

Random random = new Random();

int actualDamage = random.nextInt(damage);

this.health -= actualDamage;

this.energy += actualDamage;

if (health < 0) {

health = 0;

}

return actualDamage;

}

@Override

public String toString() {

return warrior + " " + name + "\t\t" + Font.HEART + " " + health +"\t|";

}

}

**Bowman.java**

package com.repei.droidbattle.droid;

public class Bowman extends BaseDroid {

public Bowman(String name) {

super(120, 25, name, "Bowman");

}

}

**Knight.java**

package com.repei.droidbattle.droid;

public class Knight extends BaseDroid{

public Knight(String name) {

super(240, 35, name, "Knight");

}

}

**Wizard.java**

package com.repei.droidbattle.droid;

public class Wizard extends BaseDroid{

public Wizard(String name) {

super(180, 45, name, "Wizard");

}

}

**OutBattle.java**

package com.repei.droidbattle;

import java.io.BufferedReader;

import java.io.FileReader;

import java.util.Scanner;

public class OutBattle {

BufferedReader br;

public OutBattle() {

Scanner sc = new Scanner(System.in);

System.out.print(Font.BOLD + Font.WHITE + "🖋 Type name of battle (ex. battle1.txt): " + Font.RESET);

String name = sc.nextLine();

System.out.print("\033\143");

try{

br = new BufferedReader(new FileReader(name));

String line;

while ((line = br.readLine()) != null) {

System.out.println(line);

}

} catch (Exception e) {

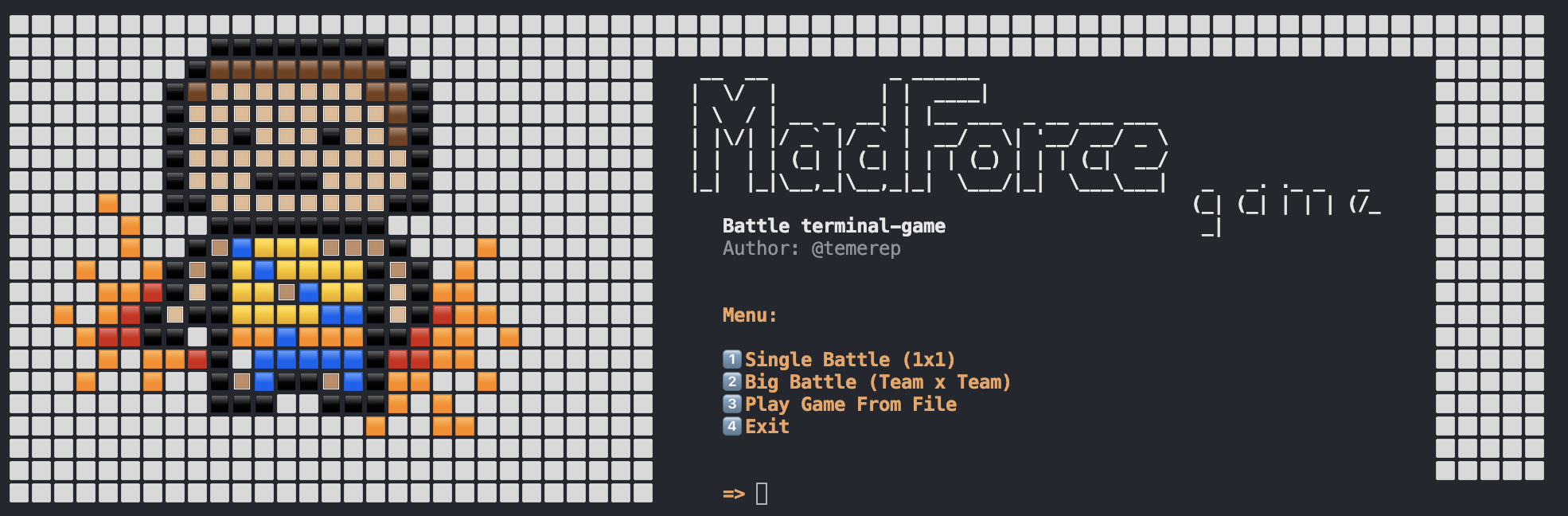
System.out.println("File not exist!");

}

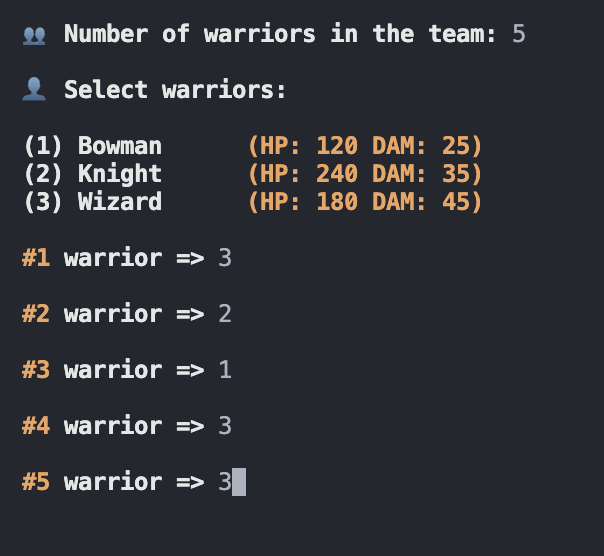
}

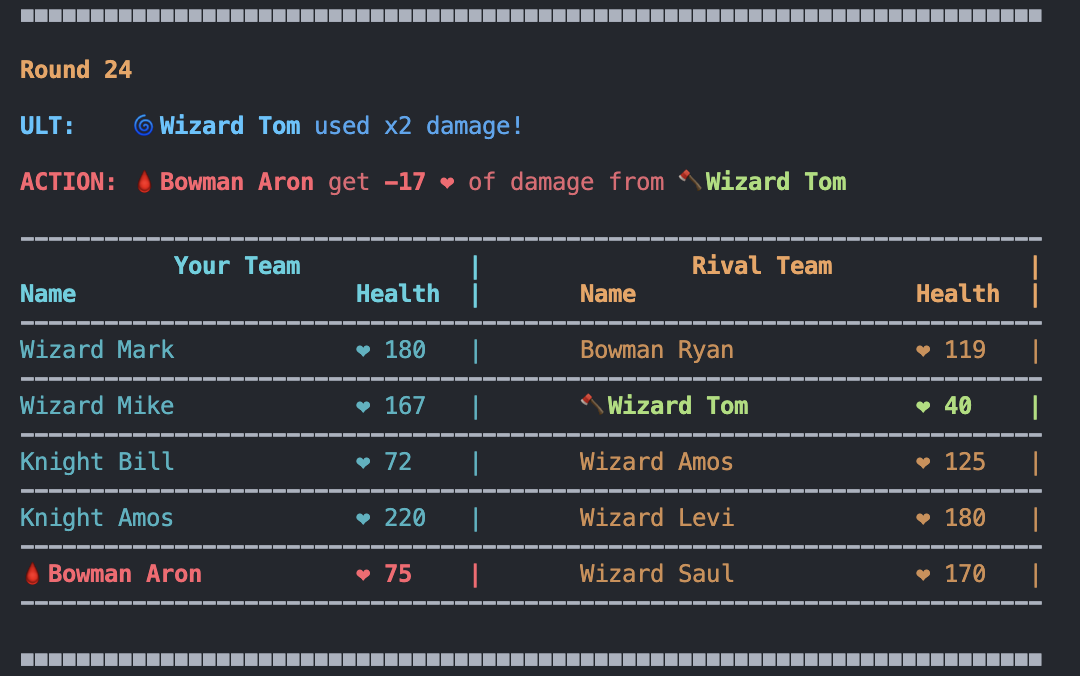
}

**Результат виконання програми**

****

**input: 2**

****

****

****

**Висновок**

На лабораторній роботі я створив свою реалізацію гри “Битва дроїдів” застосувавши мінімальний набір вимог до гри та свою фантазію, для додаткового функціоналу і візуалізації.