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”»

Виконав: студент КН-206
Филипчук Богдан

Прийняв: Скрибайло-Леськів Д. Ю.

Львів – 2022

Лабораторна робота № 3

Тема роботи: Гра «Битва дроїдів»

Завдання 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

Програмний код

```
package droids;

public class Archer extends Droid{

    public Archer(int numb) {
        super("Archer " + numb, 100, 30);
    }
}

public class Knight extends Droid {

    public Knight(int numb) {
        super("Knight " + numb, 120, 20);
    }

}

public class Sorcerer extends Droid {
    public Sorcerer(int numb) {
        super("Sorcerer " + numb, 80, 40);
    }
}
```

```java
package droids;

public class Droid {
    protected String Name;
    protected int currHealth;
    protected int Damage;

    protected int defaultHealth;

    public Droid(String name, int health, int damage) {
        Name = name;
        currHealth = health;
        defaultHealth = health;
        Damage = damage;
    }

    public String getName() {
        return Name;
    }

    public void setName(String name) {
        Name = name;
    }


    public int getDamage() {
        return Damage;
    }

    public int getCurrHealth() {
        return currHealth;
    }

    public void setCurrHealth(int currHealth) {
        this.currHealth = currHealth;
    }

    @Override
    public String toString() {
        return Name;
    }

    public void ResetHealth() {
        currHealth = defaultHealth;
    }

}
```

```java
package battles;

import UI.Colors;
import droids.Droid;

import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class Battle {
    public static void Attack(Droid attacker, Droid defender, FileWriter log)
throws IOException {
        defender.setCurrHealth(defender.getCurrHealth() -
attacker.getDamage());
        System.out.println(attacker + " -> " + defender + " - " + " -" +
attacker.getDamage() + " HP");
        log.write(attacker + " -> " + defender + " - " + " -" +
attacker.getDamage() + " HP\n");

    }

    public static int GetDroid(List<Droid> list) {
        for (int i = 0; i < list.size(); i++) {
            System.out.println(i + ": " + list.get(i) + " - " +
list.get(i).getCurrHealth());
        }

        Scanner scan = new Scanner(System.in);
        return scan.nextInt();
    }

    public static void ShowTeams(ArrayList<Droid> teamA, ArrayList<Droid>
teamB) {
        System.out.println(Colors.ANSI_YELLOW + "\nFirst team: " +
Colors.ANSI_RESET);
        for (int i = 0; i < teamA.size(); i++) {
            System.out.println(i + ": " + teamA.get(i) + " - " +
teamA.get(i).getCurrHealth() + " HP");
        }

        System.out.println(Colors.ANSI_YELLOW + "\nSecond team: " +
Colors.ANSI_RESET);
        for (int i = 0; i < teamB.size(); i++) {
            System.out.println(i + ": " + teamB.get(i) + " - " +
teamB.get(i).getCurrHealth() + " HP");
        }
    }
}
```

```java
package battles;

import UI.Colors;
import droids.Droid;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.Scanner;

public class Duel {
    private final Droid a;
    private final Droid b;

    private File file;
    private FileWriter DuelLog;

    public Duel(Droid a, Droid b) {
        this.a = a;
        this.b = b;
    }

    public void StartDuel() throws IOException {
        System.out.println(Colors.ANSI_RED + "\n" + a + " vs " + b + "" +
Colors.ANSI_RESET);
        CreateLog();
        DuelOrder();
        FinishDuel();
    }

    private void DuelOrder() throws IOException {
        while (true) {
            Battle.Attack(a, b, DuelLog);
            if (b.getCurrHealth() <= 0) {
                System.out.println(a + " is a winner!!!");
                DuelLog.write(a + " is a winner\n");
                break;
            }
            Battle.Attack(b, a, DuelLog);
            if (a.getCurrHealth() <= 0) {
                System.out.println(b + " is a winner!!!");
                DuelLog.write(b + " is a winner\n");
                break;
            }
        }
    }

    private void FinishDuel() throws IOException {
        a.ResetHealth();
        b.ResetHealth();
        FileSave();

        //Todo: переможець отримує один з предметів програвшого
    }
```

```
    private void CreateLog() throws IOException {
        DateTimeFormatter dtf =
DateTimeFormatter.ofPattern("yyyy_MM_dd_HH_mm");
        LocalDateTime now = LocalDateTime.now();

        String name = "E:\\Projects\\Test\\Duel_" + dtf.format(now) + ".txt";
        file = new File(name);
        file.createNewFile();
        DuelLog = new FileWriter(file);
        DuelLog.write("Player 1: " + a + "\n");

        DuelLog.write("Player 2: " + b + "\n");

        DuelLog.write("\nDuel: \n");
    }

    private void FileSave() throws IOException {
        DuelLog.close();
        System.out.println("Do you want to save duel log? (Yes/No)");

        Scanner scan = new Scanner(System.in);
        String input = scan.nextLine();
        if (input.equals("No")) {
            file.delete();
        }
        if (input.equals("Yes")) {
            System.out.println("Файл збережено у " + file.getPath());
        }

    }
}
```

```java
package battles;

import UI.Colors;
import droids.Droid;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;

public class TeamBattle {
    private final ArrayList<Droid> teamA;
    private final ArrayList<Droid> teamB;
    private FileWriter BattleLog;
    private File file;

    public TeamBattle(ArrayList<Droid> teamA, ArrayList<Droid> teamB) {
        this.teamA = teamA;
        this.teamB = teamB;
    }

    public void StartBattle() throws IOException {
        ReorderDroids();
        CreateLog();
        BattleOrder();
        FinishBattle();
    }

    private void ReorderDroids() {
        Random r = new Random();
        int from = 0;
        int toA = teamA.size();
        int toB = teamB.size();
        for (int i = 0; i < toA || i < toB; i++) {
            if (i < toA) {
                int swapIndexA = r.nextInt(toA - from) + from;
                Collections.swap(teamA, i, swapIndexA);
            }
            if (i < toB) {
                int swapIndexB = r.nextInt(toB - from) + from;
                Collections.swap(teamB, i, swapIndexB);
            }
        }

        Battle.ShowTeams(teamA, teamB);
    }
```

```
    private void BattleOrder() throws IOException {
        int a = 0, b = 0;
        while (true) {
            System.out.print(Colors.ANSI_YELLOW + "\nPlayer 1 " +
Colors.ANSI_RESET);
            AttackProc(a, teamA, teamB);
            if (teamB.size() == 0) {
                System.out.println(Colors.ANSI_GREEN + "Player 1 has won" +
Colors.ANSI_RESET);
                BattleLog.write("Player 1 has won");
                break;
            }

            System.out.println(Colors.ANSI_CYAN + "\nPlayer 2 " +
Colors.ANSI_RESET);
            AttackProc(b, teamB, teamA);
            if (teamA.size() == 0) {
                System.out.println(Colors.ANSI_GREEN + "Player 2 has won" +
Colors.ANSI_RESET);
                BattleLog.write("Player 2 has won");
                break;
            }

            a++;
            if (a >= teamA.size())
                a = 0;

            b++;
            if (b >= teamB.size())
                b = 0;
        }
    }

    private void AttackProc(int index, ArrayList<Droid> attacker,
ArrayList<Droid> defender) throws IOException {
        if (index >= attacker.size())
            index = 0;

        System.out.print(Colors.ANSI_CYAN + "with " + attacker.get(index) +
": " + Colors.ANSI_RESET);
        int attackIndex = SelectAttack(defender);
        Battle.Attack(attacker.get(index), defender.get(attackIndex),
BattleLog);
        if (defender.get(attackIndex).getCurrHealth() <= 0) {
            defender.remove(attackIndex);
        }
    }

    private int SelectAttack(List<Droid> toAttack) {
        System.out.println("Select who to attack");
        return Battle.GetDroid(toAttack);
    }

    private void FinishBattle() throws IOException {
        for (Droid item : teamA) {
            item.ResetHealth();
        }
        for (Droid item : teamB) {
            item.ResetHealth();
        }
        FileSave();
    }
```

```
    private void CreateLog() throws IOException {
        DateTimeFormatter dtf =
DateTimeFormatter.ofPattern("yyyy_MM_dd_HH_mm");
        LocalDateTime now = LocalDateTime.now();

        String name = "E:\\Projects\\Test\\TeamBattle_" + dtf.format(now) +
".txt";
        file = new File(name);
        file.createNewFile();
        BattleLog = new FileWriter(file);
        BattleLog.write("Player 1: ");
        for (Droid item : teamA) {
            BattleLog.write(item.toString() + "\n");
        }
        BattleLog.write("Player 2: ");
        for (Droid item : teamB) {
            BattleLog.write(item.toString() + "\n");
        }

        BattleLog.write("\nTeamBattle: \n");
    }

    private void FileSave() throws IOException {
        BattleLog.close();
        System.out.println("Do you want to save battle log? (Yes/No)");

        Scanner scan = new Scanner(System.in);
        String input = scan.nextLine();
        if (input.equals("No")) {
            file.delete();
        }
        if (input.equals("Yes")) {
            System.out.println("Файл збережено у " + file.getPath());
        }
    }
}
```

```java
package UI;

import battles.Battle;
import battles.Duel;
import battles.TeamBattle;
import droids.Archer;
import droids.Droid;
import droids.Knight;
import droids.Sorcerer;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Scanner;

public class MainScreen {

    private ArrayList<Droid> userA;
    private ArrayList<Droid> userB;


    public void Startup() throws IOException {
        System.out.println(Colors.ANSI_GREEN + "Welcome to Droid Attack" +
Colors.ANSI_RESET);
        while (true) {
            System.out.println("\nСтворити команду - 1");
            System.out.println("Показати команди - 2");
            System.out.println("Розрочати дуель - 3");
            System.out.println("Розпочати битву команда на команду - 4");
            System.out.println("Відтворити файл - 5");
            System.out.println("Завершити гру - 6\n");

            Scanner scan = new Scanner(System.in);
            int input = scan.nextInt();
            switch (input) {
                case 1:
                    CreateTeam();
                    break;
                case 2:
                    ShowTeams();
                    break;
                case 3:
                    PreStartDuel();
                    break;
                case 4:
                    PreStartTeamBattle();
                    break;
                case 5:
                    ReadFile();
                    break;
                case 6:
                    return;
            }
        }
    }

    private void CreateTeam() {
        System.out.print("\nНомер команди: ");
        Scanner scan = new Scanner(System.in);
        int input = scan.nextInt();
        if (input == 1)
            userA = FormTeam();
        if (input == 2)
            userB = FormTeam();
    }
```

```
    private ArrayList<Droid> FormTeam() {
        Scanner scan = new Scanner(System.in);
        ArrayList<Droid> team = new ArrayList<>();
        int i = 0;
        while (true) {
            System.out.println("Виберіть " + (i + 1) + "-ого дройда: ");
            System.out.println("1: Archer - 100 HP - 30 DMG");
            System.out.println("2: Knight - 120 HP - 20 DMG");
            System.out.println("3: Sorcerer - 80 HP - 40 DMG");
            System.out.println("0: Закінчити формувати команду");

            int input = scan.nextInt();
            switch (input) {
                case 1:
                    team.add(new Archer(i + 1));
                    i++;
                    break;
                case 2:
                    team.add(new Knight(i + 1));
                    i++;
                    break;
                case 3:
                    team.add(new Sorcerer(i + 1));
                    i++;
                    break;
                case 0:
                    return team;
            }
        }
    }

    private void ShowTeams() {
        if (userA == null || userB == null) {
            System.out.println("Not all teams are formed");
            return;
        }

        Battle.ShowTeams(userA, userB);
    }

    private void PreStartDuel() throws IOException {
        System.out.println("User1: Select whom to attack with");
        Droid first = userA.get(Battle.GetDroid(userA));
        System.out.println("User2: Select whom to attack with");
        Droid second = userB.get(Battle.GetDroid(userB));

        Duel duel = new Duel(first, second);
        duel.StartDuel();
    }

    private void PreStartTeamBattle() throws IOException {
        TeamBattle teamBattle = new TeamBattle(userA, userB);
        teamBattle.StartBattle();
    }

    private void ReadFile() {
        System.out.println("Name of file to read: ");
        Scanner scanConsole = new Scanner(System.in);
        String fileName = scanConsole.nextLine();
        File file = new File(fileName);
        try {
            System.out.println();
            Scanner scanFile = new Scanner(file);
            while (scanFile.hasNextLine()) {
                System.out.println(scanFile.nextLine());
            }
            System.out.println(Colors.ANSI_RED + "\nEnd of file" + Colors.ANSI_RESET);
        } catch (FileNotFoundException e) {
            System.out.println("Error in reading file");
        }
    }
}
```

Висновок

Метою даної лабораторної роботи було навчитися працювати з наслідуванням та поліморфізмом у мові програмування Java. Створив базовий клас ***Droid***, від якого унаслідуються класи ***Archer***, ***Knight***, ***Sorcerer***. Реалізував битви один на один та команда на команду. За бажанням користувача ходи битв записуються у файл, який потім можна відторити.