Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



Звіт

з лабораторної роботи № 2

з дисципліни: «Кросплатформенні засоби програмування»

на тему: «Класи та пакети»

**Виконав:**

студент групи КІ-306

Фодор Андрій

**Прийняв:**

доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

**Мета роботи:** ознайомитися з процесом розробки класів та пакетів мовою Java.

**Завдання (варіант № 21)**

1. Написати та налагодити програму на мові Java, що реалізує у вигляді класу предметну область згідно варіанту. Програма має задовольняти наступним вимогам:

• програма має розміщуватися в пакеті Група.Прізвище.Lab2;

• клас має містити мінімум 3 поля, що є об’єктами класів, які описують складові частини предметної області;

• клас має містити кілька конструкторів та мінімум 10 методів;

• для тестування і демонстрації роботи розробленого класу розробити клас-драйвер;

• методи класу мають вести протокол своєї діяльності, що записується у файл;

• розробити механізм коректного завершення роботи з файлом (не надіятися на метод finalize());

• програма має володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

2. Автоматично згенерувати документацію до розробленої програми.

3. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.

4. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.

5. Дати відповідь на контрольні запитання.

ВАРІАНТ 21 – ПІСТОЛЕТ

**Вихідний код програми**

**Файл PistolApp.java**

/\*\*

\* lab 2 package

\*/

package KI306.Fodor.Lab2;

import java.io.\*;

/\*\*

\* Pistol Application class implements main method for

\* Pistol class abilities demonstration

\*/

public class PistolApp {

/\*\*

\* @param args

\* @throws FileNotFoundException

\*/

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

Pistol p1 = new Pistol();

//Representation

System.out.println("Model of your pistol is "+ p1.getGunsModel());

System.out.println("Now you have "+ p1.getGunsBulletCount() + " bullets");

//Make 7 shoots at distance 10 m.

System.out.println("-----Start shooting------");

p1.makeSomeShoots(7, 10);

//Statistics

System.out.println("Your accuracy is " + p1.getAccuracy());

System.out.println("You have " + p1.getHitShoots() + " hit shoots");

System.out.println("You have " + p1.getMissedShoots() + " missed shoots");

System.out.println("Your pistol serviceability is " + p1.getGunsServiceability());

System.out.println("Now you have "+ p1.getGunsBulletCount() + " bullets");

//Pistol reloading

p1.reloadGun();

System.out.println("-----Reloaded pistol-----");

System.out.println("Now you have "+ p1.getGunsBulletCount() + " bullets");

//Pistol repairing

p1.RepairGun(500);

System.out.println("Your pistol serviceability is " + p1.getGunsServiceability());

p1.dispose();

}

}

**Файл Pistol.java**

/\*\*

\* lab 2 package

\*/

package KI306.Fodor.Lab2;

import java.io.\*;

/\*\*

\* Class <code>Pistol</code> implements pistol

\*/

public class Pistol {

private Bullet bullets;

private Serviceability serviceability;

private Shooting shoot;

private PrintWriter fout;

/\*\*

\* Constructor

\* @throws FileNotFoundException

\*/

public Pistol()throws FileNotFoundException {

bullets = new Bullet();

serviceability = new Serviceability();

shoot = new Shooting();

fout = new PrintWriter(new File("Log.txt"));

}

/\*\*

\* Constructor

\* @param <code>bulletCount</code> bullet count, default 20

\* @param <code>model</code> represent model of pistol, default random value of array

\* @throws FileNotFoundException

\*/

public Pistol(int bulletCount, String model)throws FileNotFoundException {

bullets = new Bullet(bulletCount);

serviceability = new Serviceability(model);

shoot = new Shooting();

fout = new PrintWriter(new File("Log.txt"));

}

/\*\*

\* Method makeSomeShoots simulate (howMany) shots on distance (distance)

\* @param <code>howMany</code> is how many shots do you want to do

\* @param <code>distance</code> is from what distance you want shooting

\*/

public void makeSomeShoots(int howMany, int distance){

fout.println("-----You started shooting-----");

fout.flush();

while (howMany != 0){

if(shoot.makeShoot(distance, serviceability, bullets)){

System.out.println("You hit the target");

fout.println("You hit the target");

fout.flush();

}else{

System.out.println("You missed the target");

fout.println("You missed the target");

fout.flush();

}

howMany--;

}

}

/\*\*

\* Method returns how many hit shots do you done

\* @return hit shoots

\*/

public int getHitShoots(){

return shoot.getHitShoots();

}

/\*\*

\* Method returns how many missed shots do you done

\* @return missed shoots

\*/

public int getMissedShoots(){

return shoot.getMissedShoots();

}

/\*\*

\* Method allow to get calculated shooting`s accuracy

\* @return current shooting`s accuracy

\*/

public String getAccuracy(){

return shoot.getAccuracyOfShots();

}

//Service

/\*\*

\* Method allow you to repair your weapon for (money)

\* @param <code>money</code> is your money, that you pay for repairing

\*/

public void RepairGun(int money){

if(serviceability.repair(money)){

fout.println("-----You have repaired your gun-----");

fout.flush();

}else{

fout.println("-----You failed to repair your gun-----");

fout.flush();

}

}

/\*\*

\* Method return serviceability of your pistol

\* @return pistol serviceability

\*/

public int getGunsServiceability(){

return serviceability.getServ();

}

/\*\*

\* Method return model of your pistol

\* @return pistol model

\*/

public String getGunsModel(){

return serviceability.getModel();

}

/\*\*

\* Method releases used recourses

\*/

public void dispose(){

fout.close();

}

//Bullets

/\*\*

\* Method return current count of bullets in your pistol

\* @return current pistol bullets

\*/

public int getGunsBulletCount(){

return bullets.getBullets();

}

/\*\*

\* Method release weapon reloading

\*/

public void reloadGun(){

bullets.reload();

fout.println("-----You have reloaded your gun-----");

}

//...пістолет має вміти стріляти, перезаряджатися, ремонтуватися за допомогою цих класів

}

/\*\*

\* Class helper

\*/

class Bullet{

//...кількість, перезарядка

private int bullet;

/\*\*

\* Default constructor release initialization

\*/

public Bullet(){

bullet = 20;

}

/\*\*

\* Due to this constructor user can load as many bullets as he wants

\* @param <code>count</code> how many bullets user want to load to pistol

\*/

public Bullet(int count){

if (count <= 20){

bullet = count;

}else {

bullet = 0;

System.out.println("Pistol can consist maximum 20 bullets");

}

}

/\*\*

\* Method is used when user make shoot and count of his bullets decrements

\* @return boolean value, that indicate whether the shooting was released and how

\*/

public boolean reduce(){

if(bullet == 0){

System.out.println("Your gun is unloaded");

return false;

} else if(bullet == 1){

System.out.println("It was your last bullet");

bullet--;

return true;

}else{

bullet--;

return true;

}

}

/\*\*

\* Method allows to refresh count of bullets to maximum

\*/

public void reload(){

bullet = 20;

}

/\*\*

\* Method return current bullet count

\* @return

\*/

public int getBullets(){

return bullet;

}

}

/\*\*

\* Class-helper

\*/

class Serviceability{

//...справність пістолета, модель, ремонт

private int serv;

private String model;

private String[] availableModels = new String[] {"FN Five-seven", "Glock 17", "HK45 Tactical", "Sig Sauer M11-A1 9mm", "Beretta M9"};

/\*\*

\* Constructor that implements default initialization of serviceability

\*/

public Serviceability(){

int random = (int)(Math.random()\*4);

serv = 100;

model = availableModels[random];

}

/\*\*

\* Constructor that allow to choose specific model user want

\* @param <code>model</code> specific model, mot random

\*/

public Serviceability(String model){

model = model;

serv = 100;

}

/\*\*

\* Method returns model of pistol

\* @return model of pistol

\*/

public String getModel (){

return model;

}

/\*\*

\* Method returns current pistol serviceability

\* @return pistol serviceability

\*/

public int getServ(){

return serv;

}

/\*\*

\* Method allow making shoots if weapon is not broken

\*/

public void reduce(){

serv -= (int)(Math.random()\*3);

if(serv < 10){

System.out.println("WARNING: Serviceability less then 10%");

}

}

/\*\*

\* Method allow repair your weapon for 500 money

\* @param <code>modey</code> your money that you pay for repairing

\* @return boolean value that indicate whether weapon was repaired

\*/

public boolean repair(int money){

if(money >= 500){

int res = (int) (Math.random() \* 10);

if(res >= 4){

System.out.println("Your weapon is repaired successfully");

serv = 100;

return true;

}

else {

System.out.println("Sorry, your weapon could not be repaired");

return false;

}

}else{

System.out.println("Repairs cost 500 units");

return false;

}

}

}

/\*\*

\* Class-helper

\*/

class Shooting{

//...виконує постріл, зменшує справність та кількість пуль

//there is only default constructor

private int hitShoots = 0, missedShoots = 0;

/\*\*

\* Method returns count of hit shoots

\* @return count of hit shoots

\*/

public int getHitShoots(){

return hitShoots;

}

/\*\*

\* Method return missed shoots

\* @return count of missed shoots

\*/

public int getMissedShoots(){

return missedShoots;

}

/\*\*

\* Method returns string with current accuracy

\* @return current accuracy

\*/

public String getAccuracyOfShots(){

return Math.round(((double)hitShoots/(double) (hitShoots+missedShoots)\*100)) + "%";

}

/\*\*

\* Method simulate one shoot

\* @param <code>distance</code> distance for shooting

\* @param <code>s</code> pistol serviceability

\* @param <code>b</code> pistol bullets

\* @return boolean value that indicate whether the shoot wes successful

\*/

public boolean makeShoot(int distance, Serviceability s, Bullet b){

if(s.getServ() <= 0){

System.out.println("Your weapon is broken");

return false;

}else if(!b.reduce()){

return false;

}else{

s.reduce();

int res = distance\*(int)(Math.random() \* 10);

if(res < 50){

hitShoots++;

return true;

}else{

missedShoots++;

return false;

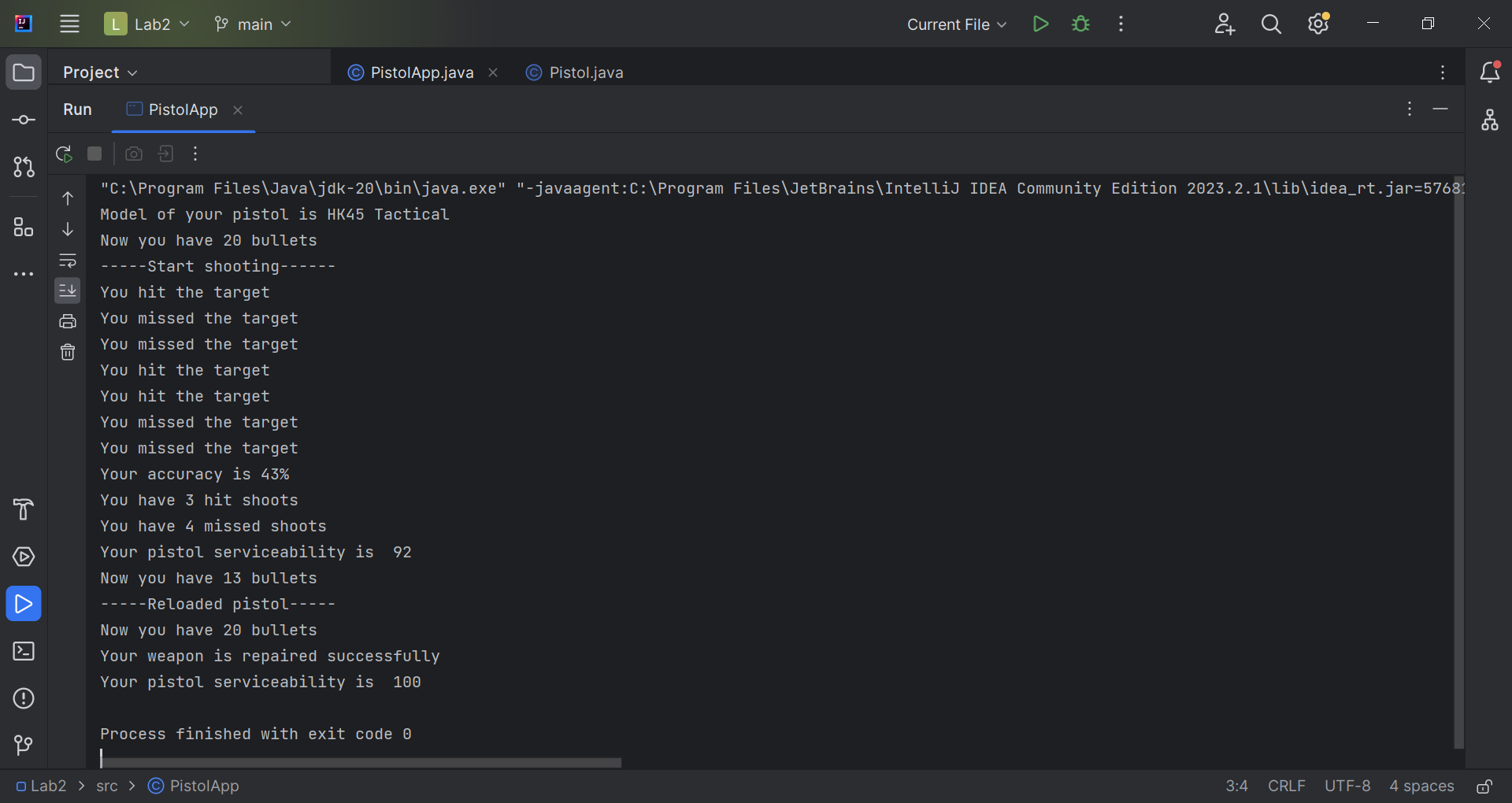
}

}

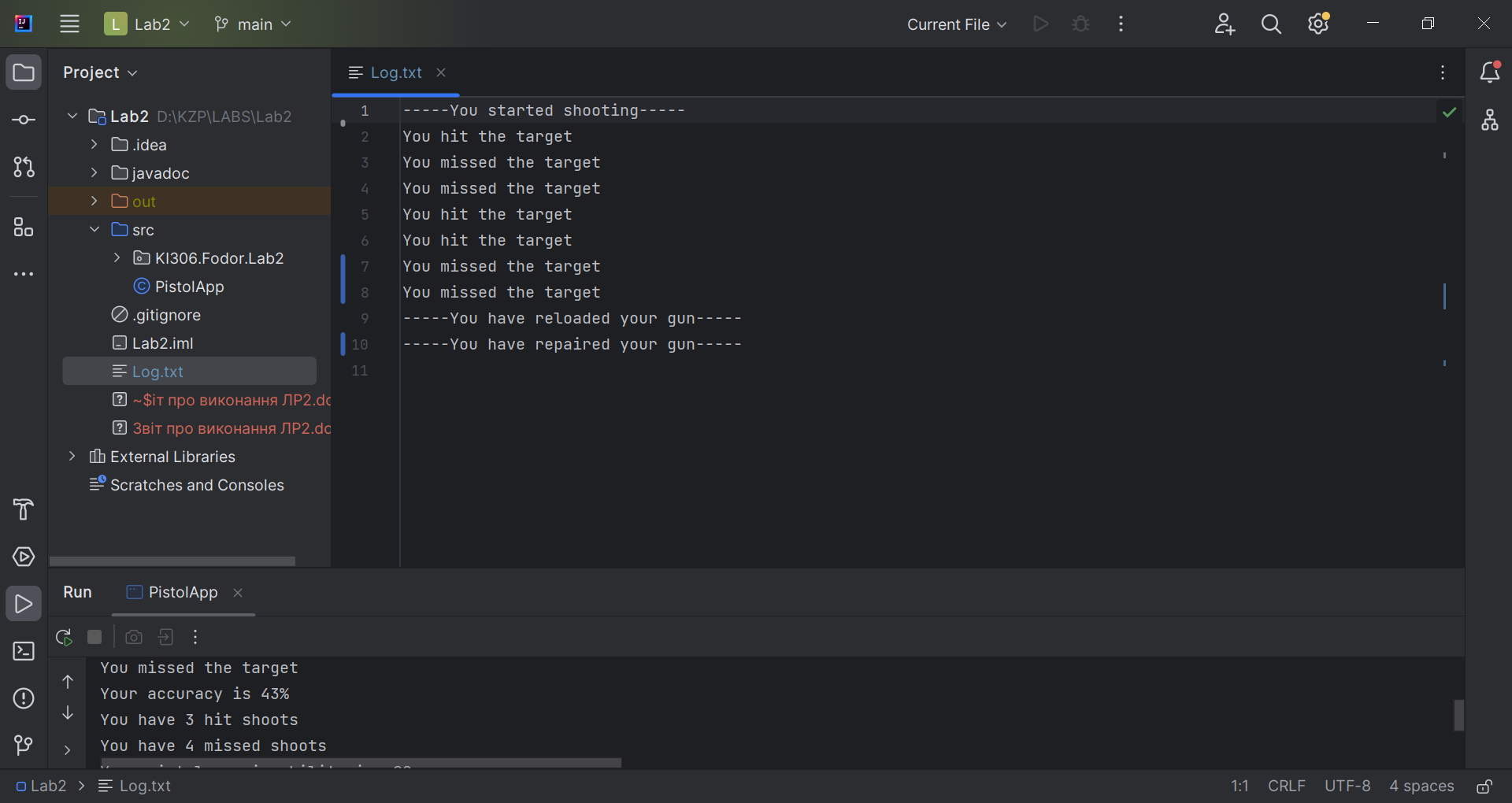
}

}

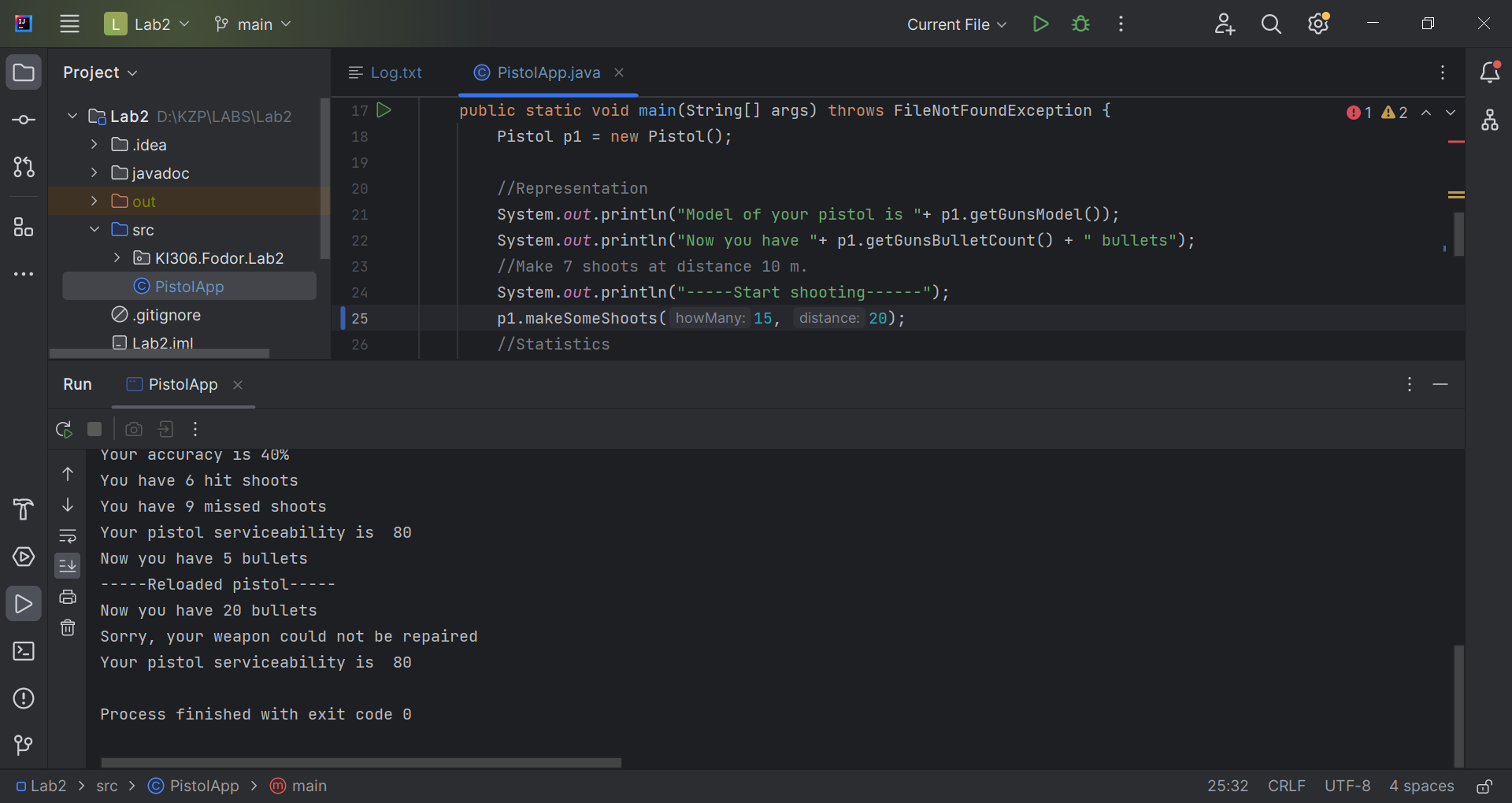
**Результат виконання програми**



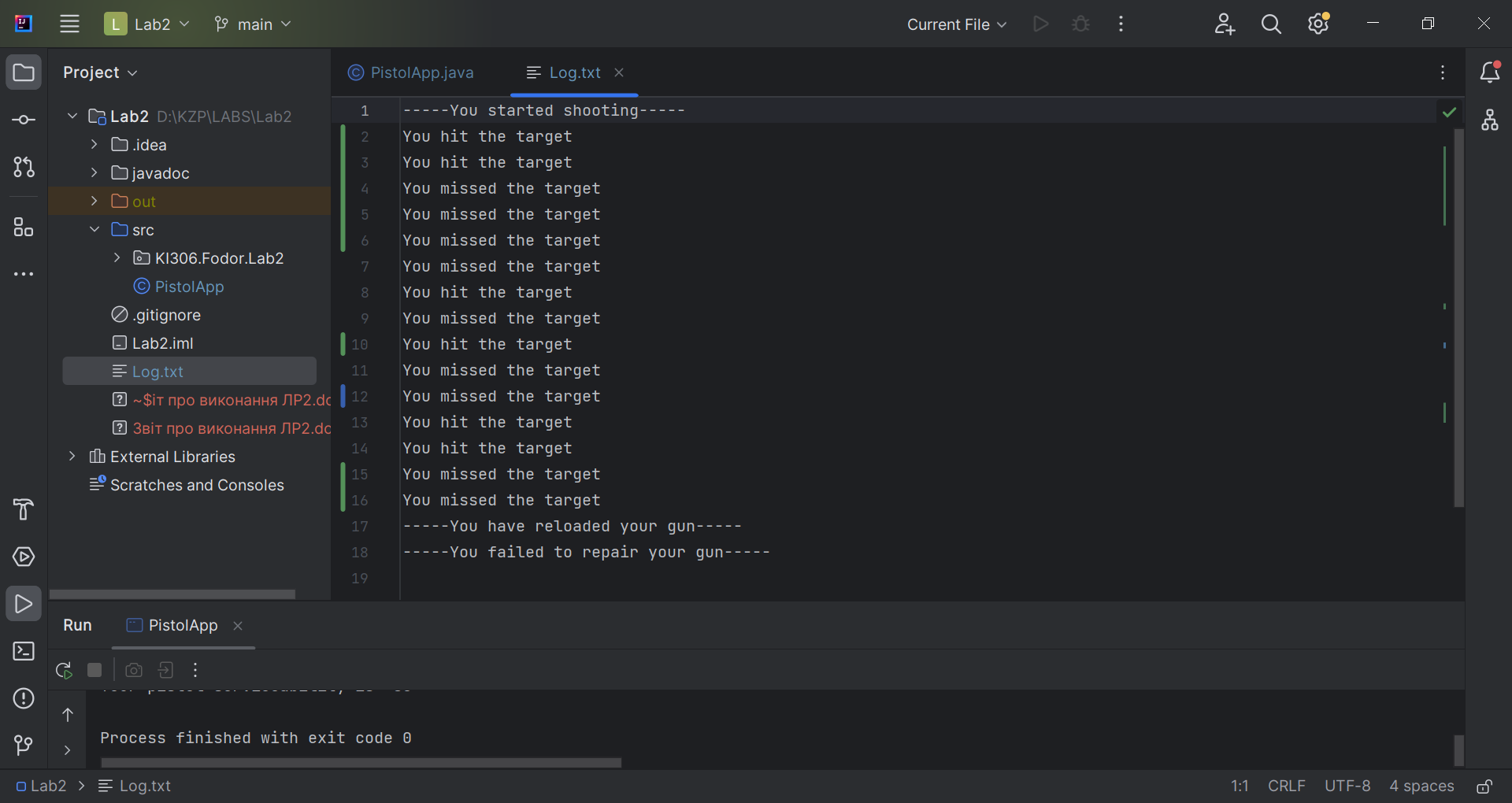
Вміст файлу Log.txt після виконання вказаного фрагменту коду:



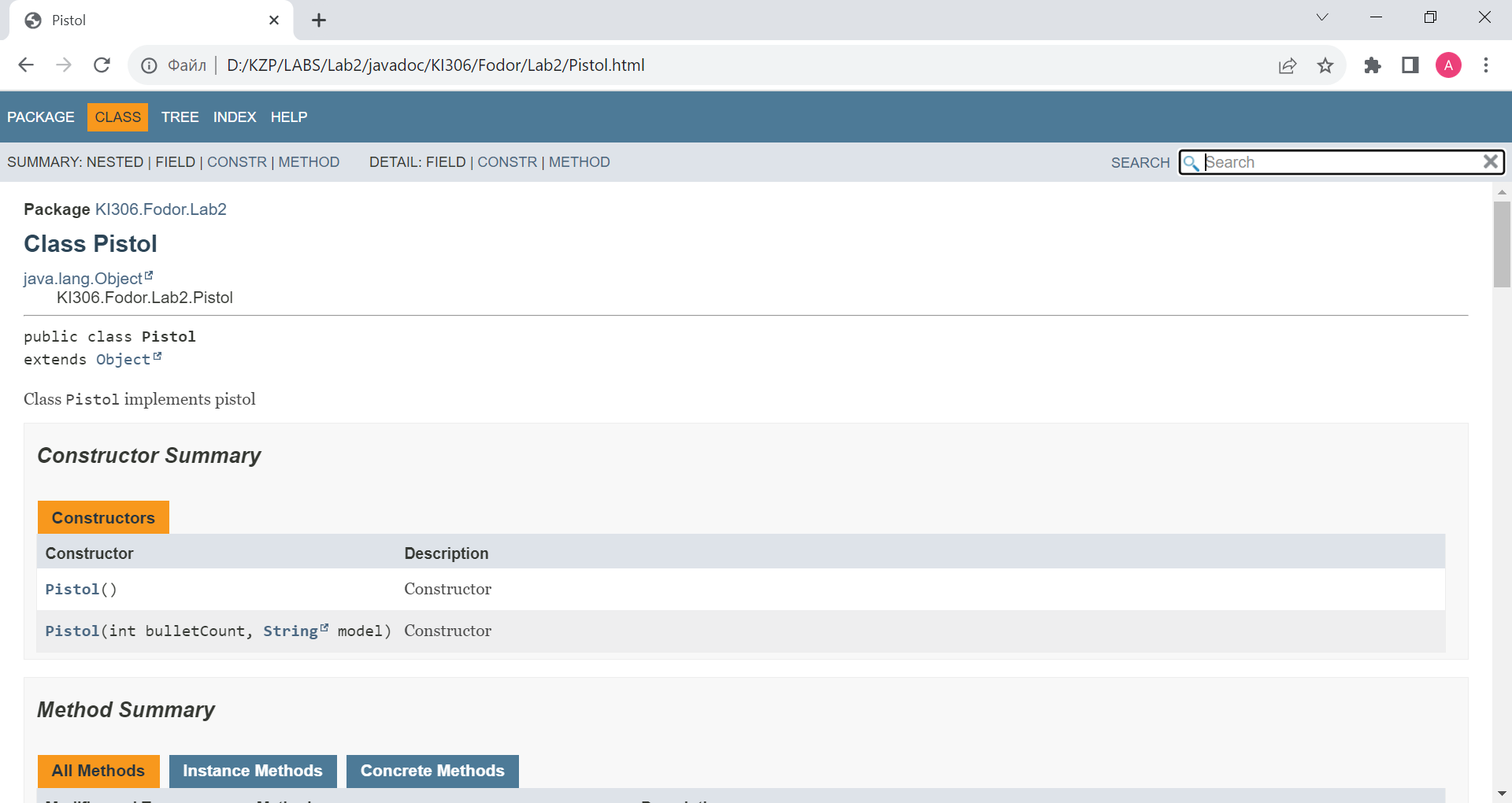
Змінив значення кількості пострілів на 15, дистанцію – на 20 м



Вміст файлу Log.txt після виконання вказаного фрагменту коду:



**Фрагмент згенерованої документації**



**Висновок**

На цій лабораторній роботі я ознайомився з процесом розробки класів та пакетів мовою Java. Закріпив отримані знання на прикладі індивідуального завдання, що полягало в реалізації класу пістолета та допоміжних йому класів. Отримав хороші навчики створення класів та їх коректного використання