НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Программирование

Лабораторная работа № 2

Выполнил студент

Хвостова Ирина Леонидовна

Группа № Р3124

Преподаватель: Харитонова Анастасия Евгеньевна

г. Санкт-Петербург

2022

**Вариант 4254**

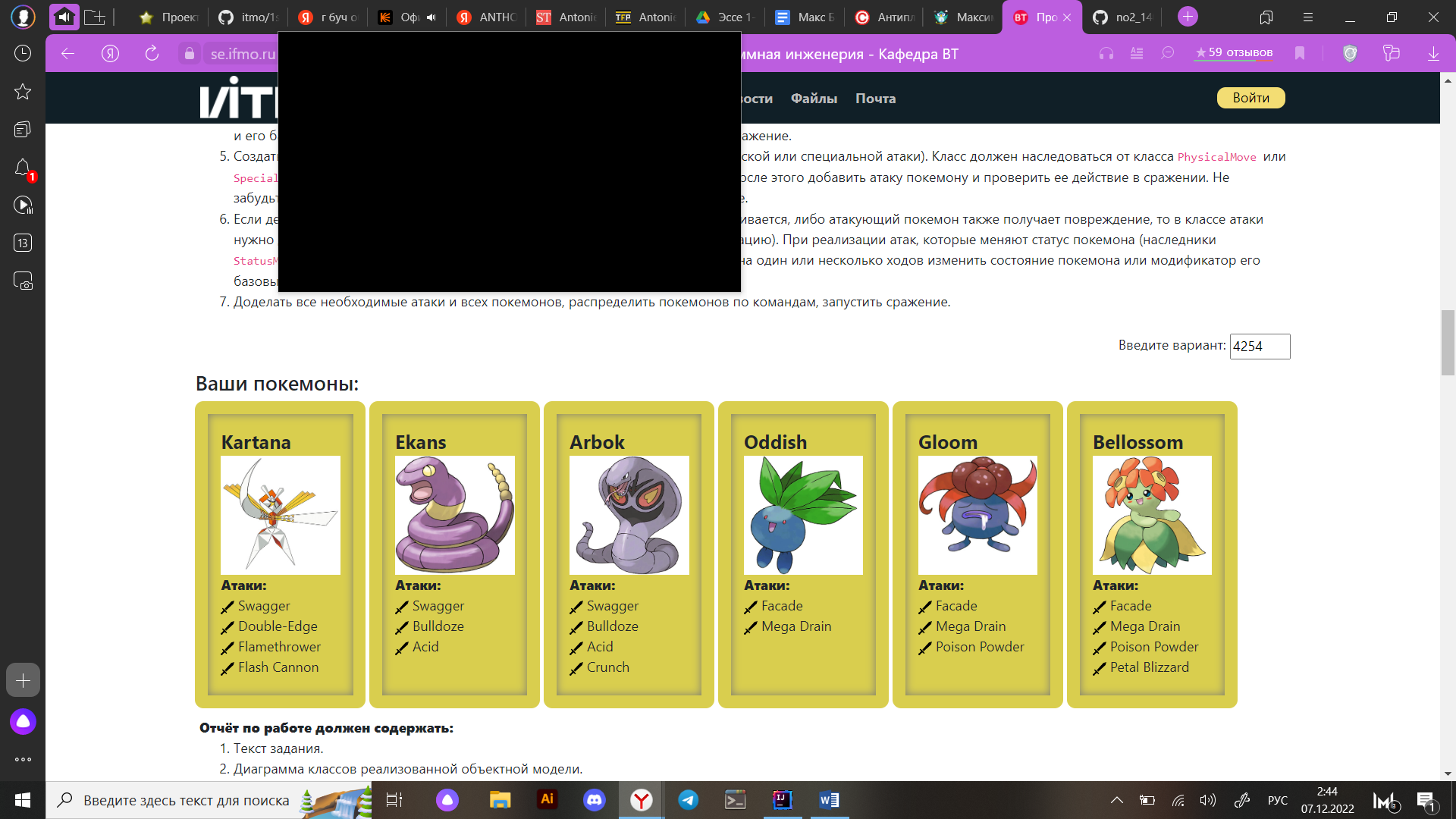
Задание:

На основе базового класса Pokemon написать свои классы для заданных видов покемонов. Каждый вид покемона должен иметь один или два типа и стандартные базовые характеристики:

* очки здоровья (HP)
* атака (attack)
* защита (defense)
* специальная атака (special attack)
* специальная защита (special defense)
* скорость (speed)

Классы покемонов должны наследоваться в соответствии с цепочкой эволюции покемонов. На основе базовых классов PhysicalMove, SpecialMove и StatusMove реализовать свои классы для заданных видов атак.

Атака должна иметь стандартные тип, силу (power) и точность (accuracy). Должны быть реализованы стандартные эффекты атаки. Назначить каждому виду покемонов атаки в соответствии с вариантом. Уровень покемона выбирается минимально необходимым для всех реализованных атак.

Используя класс симуляции боя Battle, создать 2 команды покемонов (каждый покемон должен иметь имя) и запустить бой.

Исходный код:

import ru.ifmo.se.pokemon.\*;  
import pokemons.\*;  
  
public class Main {  
 public static void main(String[] args) {  
  
 Battle b = new Battle();  
 Pokemon p1 = new Kartana("Константин", 10);  
 Pokemon p2 = new Ekans("Боба", 10);  
 Pokemon p3 = new Arbok("Биба", 10);  
 Pokemon p4 = new Oddish("Пупа", 10);  
 Pokemon p5 = new Gloom("Лупа", 10);  
 Pokemon p6 = new Bellossom("Палупа", 10);  
 b.addAlly(p1);  
 b.addFoe(p3);  
 b.addAlly(p2);  
 b.addFoe(p4);  
 b.addAlly(p5);  
 b.addFoe(p6);  
 b.go();  
 }  
}

Блок атак:

* PhysicalAttacks

1. Bulldoze

package attacks.PhysicalAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Bulldoze extends PhysicalMove {  
 public Bulldoze() {  
  
 super(Type.*GROUND*, 60, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 pokemon.setMod(Stat.*SPEED*, -1); //доп. эффект, накладывающийся на оборон. покемона  
 super.applyOppEffects(pokemon);  
 }  
  
 @Override  
 protected String describe() {  
  
 return "вызывает Bulldoze";  
 }  
}

1. Crunch

package attacks.PhysicalAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Crunch extends PhysicalMove {  
 public Crunch() {  
 super(Type.*DARK*, 80, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 Effect effect = new Effect().chance(0.2).stat(Stat.*DEFENSE*, -1);  
 pokemon.addEffect(effect); //добавляет эффект, влияющий на покемона  
 super.applyOppEffects(pokemon);  
 }  
  
 @Override  
 protected String describe() {  
  
 return "вызывает Crunch";  
 }  
}

1. Double Edge

package attacks.PhysicalAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class DoubleEdge extends PhysicalMove {  
 public DoubleEdge() {  
 super(Type.*NORMAL*, 120, 100);  
 }  
  
 @Override  
 protected void applySelfDamage(Pokemon pokemon, double damage) {  
 pokemon.setMod(Stat.*HP*, -(int) (Math.*round*(damage)) / 3);  
 System.*out*.println("Я исцеляюсь!");  
 }  
  
 @Override  
 protected String describe() {  
 return "применяет Double Edge";  
 }  
  
}

1. Façade

package attacks.PhysicalAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Facade extends PhysicalMove {  
 public Facade() {  
 super(Type.*NORMAL*, 70, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 super.applyOppEffects(pokemon);  
 if (pokemon.getCondition() == Status.*BURN* || pokemon.getCondition() == Status.*POISON*||pokemon.getCondition()==Status.*PARALYZE*){  
 Effect effect = new Effect().stat(Stat.*ATTACK*, 2);  
 pokemon.addEffect(effect);  
 }  
 }  
 @Override  
 protected String describe() {  
  
 return "вызывает Facade";  
 }  
  
}

1. Petal Blizzard

package attacks.PhysicalAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class PetalBlizzard extends PhysicalMove {  
 public PetalBlizzard() {  
  
 super(Type.*GRASS*, 90, 100);  
 }  
  
 @Override  
 protected String describe() {  
  
 return "вызывает Petal Blizzard";  
 }  
}

* SpecialAttacks

1. Acid

package attacks.SpecialAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Acid extends SpecialMove {  
 public Acid() {  
 super(Type.*POISON*, 40, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 Effect effect = new Effect().chance(0.1).stat(Stat.*SPECIAL\_DEFENSE*, -1);  
 pokemon.addEffect(effect); //добавляет эффект, влияющий на покемона  
 super.applyOppEffects(pokemon);  
 }  
  
 @Override  
 protected String describe() {  
  
 return "вызывает Acid";  
 }  
}

1. Flamethrower

package attacks.SpecialAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Flamethrower extends SpecialMove{  
 public Flamethrower() {  
 super(Type.*FIRE*, 90, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 Effect effect = new Effect().chance(0.1).condition(Status.*BURN*);  
 pokemon.addEffect(effect); //добавляет эффект, влияющий на покемона  
 super.applyOppEffects(pokemon);  
 }  
  
 @Override  
 protected String describe() {  
 return "вызывает Flamethrower";  
 }  
}

1. Flash Cannon

package attacks.SpecialAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class FlashCannon extends SpecialMove {  
 public FlashCannon() {  
 super(Type.*STEEL*, 80, 100);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 Effect effect = new Effect().chance(0.1).stat(Stat.*SPECIAL\_DEFENSE*, -1);  
 pokemon.addEffect(effect); //добавляет эффект, влияющий на покемона  
 super.applyOppEffects(pokemon);  
 }  
  
 @Override  
 protected String describe() {  
  
 return "вызывает FlashCannon";  
 }  
  
}

1. Mega Drain

package attacks.SpecialAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class MegaDrain extends SpecialMove {  
 public MegaDrain() {  
 super(Type.*GRASS*, 40, 100);  
 }  
  
 @Override  
 protected void applySelfDamage(Pokemon pokemon, double damage) {  
 pokemon.setMod(Stat.*HP*, -(int) (Math.*round*(damage)) / 2);  
 System.*out*.println("Я исцеляюсь!");  
 }  
  
 @Override  
 protected String describe() {  
 return "применяет Mega Drain";  
 }  
}

* StatusAttacks

1. Poison Powder

package attacks.StatusAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class PoisonPowder extends StatusMove {  
 public PoisonPowder() {  
 super(Type.*POISON*, 0, 75);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 super.applyOppEffects(pokemon);  
 pokemon.setCondition(new Effect().condition(Status.*POISON*));  
 }  
  
 @Override  
 protected String describe() {  
  
 return "применяет Poison Powder";  
 }  
  
}

1. Swagger

package attacks.StatusAttacks;  
  
import ru.ifmo.se.pokemon.\*;  
  
public class Swagger extends StatusMove {  
 public Swagger() {  
 super(Type.*NORMAL*, 0, 85);  
 }  
  
 @Override  
 protected void applyOppEffects(Pokemon pokemon) {  
 pokemon.confuse();  
 pokemon.setMod(Stat.*ATTACK*, 2);  
 super.applyOppEffects(pokemon);  
 }  
  
 @Override  
 protected String describe() {  
  
 return "применяет Confide";  
 }  
}

Блок покемонов:

* Arbok

package pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import attacks.StatusAttacks.Swagger;  
import attacks.PhysicalAttacks.Bulldoze;  
import attacks.SpecialAttacks.Acid;  
import attacks.PhysicalAttacks.Crunch;  
  
public class Arbok extends Pokemon {  
 public Arbok(String name, int lvl) {  
 super(name, lvl);  
 this.setStats(60, 95, 69, 65, 79, 80);  
 this.setType(Type.*POISON*);  
 this.addMove(new Swagger());  
 this.addMove(new Bulldoze());  
 this.addMove(new Acid());  
 this.addMove(new Crunch());  
 }  
}

* Bellossom

package pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import attacks.PhysicalAttacks.Facade;  
import attacks.SpecialAttacks.MegaDrain;  
import attacks.StatusAttacks.PoisonPowder;  
import attacks.PhysicalAttacks.PetalBlizzard;  
  
public class Bellossom extends Pokemon {  
 public Bellossom(String name, int lvl) {  
 super(name, lvl);  
 this.setStats(75, 80, 95, 90, 100, 50);  
 this.setType(Type.*GRASS*);  
 this.addMove(new Facade());  
 this.addMove(new MegaDrain());  
 this.addMove(new PoisonPowder());  
 this.addMove(new PetalBlizzard());  
 }  
}

* Ekans

package pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import attacks.StatusAttacks.Swagger;  
import attacks.PhysicalAttacks.Bulldoze;  
import attacks.SpecialAttacks.Acid;  
  
public class Ekans extends Pokemon {  
 public Ekans(String name, int lvl) {  
 super(name, lvl);  
 this.setStats(35, 60, 44, 40, 54, 55);  
 this.setType(Type.*POISON*);  
 this.addMove(new Swagger());  
 this.addMove(new Bulldoze());  
 this.addMove(new Acid());  
 }  
}

* Gloom

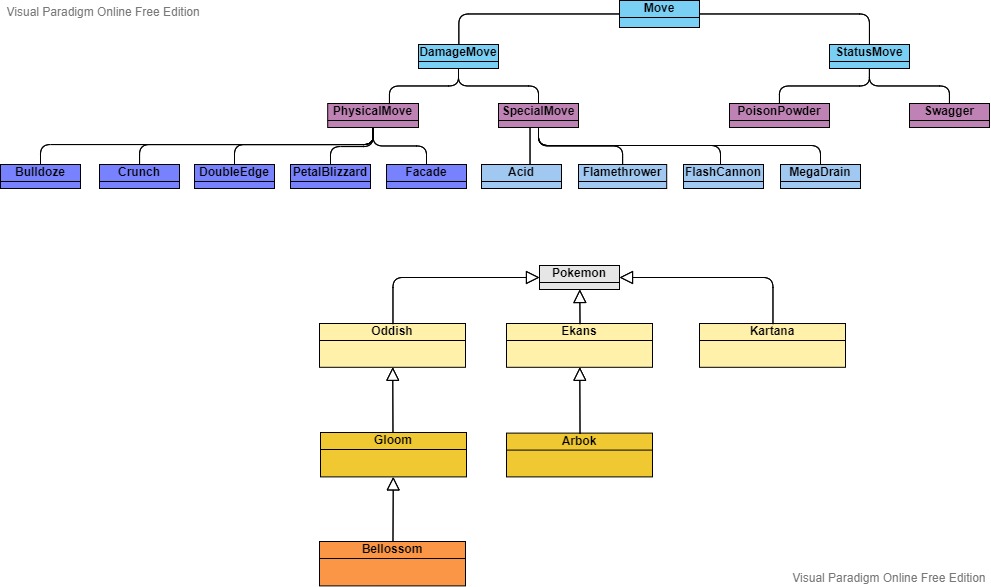
package pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import attacks.PhysicalAttacks.Facade;  
import attacks.SpecialAttacks.MegaDrain;  
import attacks.StatusAttacks.PoisonPowder;  
  
public class Gloom extends Pokemon {  
 public Gloom(String name, int lvl) {  
 super(name, lvl);  
 this.setStats(60, 65, 70, 85, 75, 40);  
 this.setType(Type.*GRASS*, Type.*POISON*);  
 this.addMove(new Facade());  
 this.addMove(new MegaDrain());  
 this.addMove(new PoisonPowder());  
 }  
}

* Kartana

package pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import attacks.StatusAttacks.Swagger;  
import attacks.PhysicalAttacks.DoubleEdge;  
import attacks.SpecialAttacks.Flamethrower;  
import attacks.SpecialAttacks.FlashCannon;  
  
public class Kartana extends Pokemon {  
 public Kartana(String name, int lvl) {  
 super(name, lvl);  
 this.setStats(59, 181, 131, 59, 31, 109);  
 this.setType(Type.*GRASS*, Type.*STEEL*);  
 this.addMove(new Swagger());  
 this.addMove(new DoubleEdge());  
 this.addMove(new Flamethrower());  
 this.addMove(new FlashCannon());  
 }  
  
}

* Oddish

package pokemons;  
  
import ru.ifmo.se.pokemon.\*;  
import attacks.PhysicalAttacks.Facade;  
import attacks.SpecialAttacks.MegaDrain;  
  
public class Oddish extends Pokemon {  
 public Oddish(String name, int lvl) {  
 super(name, lvl);  
 this.setStats(45, 50, 55, 75, 65, 30);  
 this.setType(Type.*GRASS*, Type.*POISON*);  
 this.addMove(new Facade());  
 this.addMove(new MegaDrain());  
 }  
}

Диаграмма классов реализованной объектной модели. Иерархия объектов.

Результат работы программы:

Kartana Константин из команды желтых вступает в бой!

Arbok Биба из команды полосатых вступает в бой!

Kartana Константин вызывает FlashCannon.

Arbok Биба теряет 10 здоровья.

Arbok Биба вызывает Crunch.

Kartana Константин теряет 5 здоровья.

Kartana Константин применяет Double Edge.

Arbok Биба теряет 12 здоровья.

Kartana Константин восстанавливает 4 здоровья.

Я исцеляюсь!

Arbok Биба вызывает Bulldoze.

Kartana Константин теряет 5 здоровья.

Kartana Константин уменьшает скорость.

Kartana Константин применяет Confide.

Arbok Биба увеличивает атаку.

Arbok Биба вызывает Acid.

Kartana Константин теряет 1 здоровья.

Kartana Константин не замечает воздействие типа POISON

Kartana Константин применяет Confide.

Arbok Биба увеличивает атаку.

Arbok Биба вызывает Crunch.

Kartana Константин теряет 10 здоровья.

Kartana Константин применяет Confide.

Arbok Биба увеличивает атаку.

Arbok Биба вызывает Acid.

Kartana Константин теряет 1 здоровья.

Kartana Константин не замечает воздействие типа POISON

Kartana Константин вызывает FlashCannon.

Arbok Биба теряет 11 здоровья.

Arbok Биба вызывает Crunch.

Kartana Константин теряет 4 здоровья.

Kartana Константин вызывает Flamethrower.

Arbok Биба теряет 6 здоровья.

Arbok Биба теряет сознание.

Oddish Пупа из команды полосатых вступает в бой!

Kartana Константин применяет Confide.

Oddish Пупа увеличивает атаку.

Oddish Пупа применяет Mega Drain.

Kartana Константин теряет 2 здоровья.

Oddish Пупа восстанавливает 1 здоровья.

Я исцеляюсь!

Kartana Константин применяет Double Edge.

Oddish Пупа теряет 27 здоровья.

Kartana Константин восстанавливает 9 здоровья.

Я исцеляюсь!

Oddish Пупа применяет Mega Drain.

Kartana Константин теряет 3 здоровья.

Oddish Пупа восстанавливает 1 здоровья.

Я исцеляюсь!

Kartana Константин вызывает Flamethrower.

Критический удар!

Oddish Пупа теряет 38 здоровья.

Oddish Пупа теряет сознание.

Bellossom Палупа из команды полосатых вступает в бой!

Kartana Константин вызывает Flamethrower.

Bellossom Палупа теряет 13 здоровья.

Bellossom Палупа применяет Poison Powder.

Kartana Константин не замечает воздействие типа POISON

Kartana Константин применяет Double Edge.

Bellossom Палупа теряет 10 здоровья.

Kartana Константин восстанавливает 3 здоровья.

Я исцеляюсь!

Bellossom Палупа применяет Poison Powder.

Kartana Константин не замечает воздействие типа POISON

Kartana Константин вызывает FlashCannon.

Bellossom Палупа теряет 7 здоровья.

Bellossom Палупа вызывает Facade.

Kartana Константин теряет 3 здоровья.

Kartana Константин применяет Confide.

Bellossom Палупа увеличивает атаку.

Bellossom Палупа вызывает Petal Blizzard.

Kartana Константин теряет 1 здоровья.

Kartana Константин применяет Double Edge.

Bellossom Палупа теряет 19 здоровья.

Kartana Константин восстанавливает 6 здоровья.

Я исцеляюсь!

Bellossom Палупа теряет сознание.

В команде полосатых не осталось покемонов.

Команда желтых побеждает в этом бою!

Вывод:

Во время выполнения данной лабораторной работы я

* Ознакомилась с базовыми понятиями ООП в ЯП Java.
* Применила базовые возможности ООП на практике:

1. На практике ознакомилась с понятием инкапсуляции и понятием ключевых слов public, private, protected.
2. Использовала наследование от базового класса, переопределение методов базового класса.
3. Ознакомилась с конструированием объектов, в связке с принципом инкапсуляции.
4. Ознакомилась с понятием области видимости переменных внутри различных блоков кода.

Полученные мною знания будут использоваться для дальнейшего изучения языка и обучения в целом.