**Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики.**

Факультет Программной Инженерии и Компьютерной Техники

A picture containing logo

Description automatically generated

**ПРОГРАММИРОВАНИЕ**

Лабораторная работа №3.

Вариант 33130

Выполнил:

ФИО: Окечукву Александр Чуквуэмека

ГРУППА: P3133

ПРЕПОДАВАТЕЛЬ: Шешуков Д. М.

Санкт Петербург

2021 г.

**Задача**

**Описание предметной области, по которой должна быть построена объектная модель:**

Но прежде чем Малыш успел ответить, Карлсон, ликуя, поднял руку с пистолетом над головой и выстрелил. Резкий звук прокатился по крышам и замер. В соседних домах загудели голоса: то испуганные, то сердитые, а кто-то крикнул, что нужно вызвать полицию. Тут Малыш совсем вышел из себя. Но Карлсон сидел с невозмутимым видом и жевал булочку, уже последнюю.

**Программа должна удовлетворять следующим требованиям:**

1. Доработанная модель должна соответствовать [принципам SOLID](https://en.wikipedia.org/wiki/SOLID_(object-oriented_design)).
2. Программа должна содержать как минимум два интерфейса и один абстрактный класс (номенклатура должна быть согласована с преподавателем).
3. В разработанных классах должны быть переопределены методы equals(), toString() и hashCode().
4. Программа должна содержать как минимум один перечисляемый тип (enum).

**Порядок выполнения работы:**

1. Доработать объектную модель приложения.
2. Перерисовать диаграмму классов в соответствии с внесёнными в модель изменениями.
3. Согласовать с преподавателем изменения, внесённые в модель.
4. Модифицировать программу в соответствии с внесёнными в модель изменениями.

**Отчёт по работе должен содержать:**

1. Текст задания.
2. Диаграмма классов объектной модели.
3. Исходный код программы.
4. Результат работы программы.
5. Выводы по работе.

Диаграмма классов

Graphical user interface

Description automatically generated

КОД

Main.java

package com;  
public class Main{  
 public static void main (String [] arg){  
 Kid free1 = new Kid ();  
 Carlson free2 = new Carlson ();  
 Pistol free3 = new Pistol ();  
 Rooftop free4 = new Rooftop ();  
 Harshsound free5 = new Harshsound ();  
 Voice free6 = new Voice ();  
 Someone free7 = new Someone ();  
 Neighboringhouses free8 = new Neighboringhouses ();  
 System.*out*.println();  
  
 free1.pickup();  
 free2.exultingraisedhand(free3.getName());  
 free5.rolled(free4.getName());  
 free6.buzzed(free8.getName());  
 free7.shouted();  
 free1.reckless();  
 free2.chewed();  
 }  
}

object.java

import java.util.Objects;  
  
public abstract class object implements ShowName {  
 private String name;  
 private mood value;  
  
 public object(String name) {  
 this.name = name;  
 }  
  
 protected void Join(String name) {  
  
 System.*out*.println(name + "Арнольд");  
 }  
  
 public String getName() {  
 return this.name;  
 }  
  
 public object (String name, mood value) {  
 this.name = name;  
 this.value = value;  
 }  
  
 public mood getType() {  
 return this.value;  
 }  
 public String toString() {  
 return getName();  
 }  
  
 public int Hashcode() {  
  
 return this.name.hashCode();  
 }  
 public boolean equals(object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass() ) return false;  
 object object = (object) o;  
 return Objects.*equals*(name, object.name) && value == object.value;  
 }  
 public int hashcode () {  
 return Objects.*hash*(name, value);  
}  
 }

Kid.java

package com;  
public class Kid extends object{  
 public Kid(){  
 super("Малыш");  
 Join(getName());  
 }  
 public void pickup(){  
 System.*out*.println(getName() + " успел ответить ");  
 }  
 public void reckless(){  
 System.*out*.println(getName() + "Малыш совсем вышел из себя");  
 }  
}

Pistol.java

package com;  
public class Pistol extends object{  
 public Pistol (){  
 super("пистолетом");  
 Join(getName());  
 }  
}

Neighboringhouses.java

package com;  
public class Neighboringhouses extends object{  
 public Neighboringhouses(){  
 super(" В соседних домах ");  
 Join(getName());  
 }  
}

Carlson.java

package com;  
public class Carlson extends object{  
 public Carlson(){  
 super("Карлсон");  
 Join(getName());  
 }  
 public void exultingraisedhand (String val){  
 System.*out*.println(getName() +" ликуя, поднял руку с " + val+ "истолетом над головой и выстрелил");  
 }  
 public void chewed(){  
 System.*out*.println(getName()+ "сидел с невозмутимым видом и жевал булочку");  
 }  
}

Rooftop.java

package com;  
public class Rooftop extends object{  
 public Rooftop(){  
 super("крышам");  
 Join(getName());  
 }  
}

Someone.java

package com;  
public class Someone extends object{  
 public Someone(){  
 super("кто-то ");  
 Join(getName());  
 }  
 public void shouted(){  
 System.*out*.println(getName() + " крикнул, что нужно вызвать полицию ");  
 }  
}

Harshsound.java

package com;  
public class Harshsound extends object{  
 public Harshsound(){  
 super("Резкий звук");  
 Join(getName());  
 }  
 public void rolled(String val){  
 System.*out*.println(getName()+" прокатился по "+val+"крышам и замер" );  
 }  
}

Voice.java

package com;  
  
public class Voice extends object{  
 public Voice(){  
 super("голоса");  
 Join(getName());  
 }  
 public void buzzed(String val){  
 System.*out*.println("В соседних"+val+"домах загудели голоса"+ getName());  
 }  
}

ИНТЕРФЕЙСЫ

ShowName.java

package com;  
public interface ShowName extends ShowType{  
 String getName();  
}

ShowType.java

package com;  
public interface ShowType {  
 mood getType();  
}

mood.java

package com;  
public enum mood {  
 *exulting*,  
 *frightened*,  
 *angry*,  
 *mad*,  
}

Вывод в IntelliJ

Text

Description automatically generated