САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И КОМПЬЮТЕРНОЙ ТЕХНИКИ

ЛАБОРАТОРНАЯ №3

ПО ДИСЦИПЛИНЕ

«ПРОГРАММИРОВАНИЕ»

ВАРИАНТ №311909

Выполнила:
Студентка группы P3119
Петрова Анастасия Александровна

Преподаватель:
Рыбаков Степан Дмитриевич

Санкт-Петербург
2022

Оглавление

Задание ..... *Ошибка! Закладка не определена.*

Исходный код программы ..... *Ошибка! Закладка не определена.*

Компиляция и разворачивание программы ..... *Ошибка! Закладка не определена.*

Диаграмма классов ..... *Ошибка! Закладка не определена.*

Выводы ..... *Ошибка! Закладка не определена.*

Задание:

> Дети стали расходиться. Гунилла и Кристер тоже ушли. Малыш и Альберг остались вдвоем, чему Малыш был очень рад. Он взял щенка на колени и стал ему что-то нашептывать. Щенок лизнул Малыша в лицо и заснул, сладко посапывая. Потом пришла мама из прачечной, и сразу все изменилось. Малышу сделалось очень грустно: мама вовсе не считала, что Альбергу негде жить, -- она позвонила по номеру, который был выгравирован на ошейнике Альберга, и рассказала, что ее сын нашел маленького черного щенка-пуделя.

Программа должна удовлетворять следующим требованиям:

1. Доработанная модель должна соответствовать принципам SOLID.
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.

Исходный код программы:

```java
package org.itmo.lab3;
import java.util.Objects;
public class People extends StoryCharacters {

    public People(NamePerson name){
        super(name);
    }

    @Override
    public boolean equals(Object obj)
    {
        if (obj == this) {
            return true;
        }

        if (obj == null || obj.getClass() != getClass()) {
            return false;
        }

        Puppy p = (Puppy) obj;

        return Objects.equals(p.getName().toString(), super.getName().toString());
    }

    @Override
    public int hashCode()
    {
        return Objects.hash(super.getName().toString());
    }

    @Override
    public String toString() {
        return getClass().getName() + " [" + super.getName()  + "]";
    }

    public void asleep(String pDescription) {
        System.out.println(this + " "+ pDescription);
    }
    public void come(String pDescription) {
        System.out.println(this + " "+ pDescription);
    }
    public void goOut() {
        System.out.println(this+" ушел(ла)");
    }

    // Руководствоваясь принципом SOLID перенесу этот метод в отедьный класс тк звонок можно сделать разными способоами
    public Boolean call(String pDescription,TypeConn pTypeConn) {
        Call vCall = new Call(pTypeConn);
        return vCall.makeCall(this,pDescription);
        //System.out.println(this + " "+ pDescription);
        //return true;
    }

    public void tell(String pDescription) {
        //System.out.println(this +" сказала,что "+ pDescription);
       Tell vTell = new Tell();
       vTell.methTell(this,pDescription);
    }
```

```java
package org.itmo.lab3;
import java.util.Objects;
class Tell{
    public Tell(){ //конструктор
    }
    public void methTell(People pTeller,String pDescription) {
            System.out.println(pTeller +" сказала,что "+ pDescription);
    }

}
```

```java
package org.itmo.lab3;
public interface IParents  {
     public void toMakeDecisions(String pDescription);
}
```

```java
package org.itmo.lab3;
import java.util.Objects;
public class Call  {
    private TypeConn typeConn;

    public TypeConn getType() {
        return typeConn;
    }
    public Call(TypeConn pTypeConn) { //конструктор
        this.typeConn=pTypeConn;

    }

    public Boolean makeCall(People pCaller,String pDescription) {
            System.out.println(pCaller.toString()+" позвонила по "+ this +" "+ pDescription);
            return true;
    }

    @Override
    public String toString() {
        return getClass().getName() + " [" + getType()  + "]";
    }

    @Override
    public boolean equals(Object obj)
    {
        if (obj == this) {
            return true;
        }

        if (obj == null || obj.getClass() != getClass()) {
            return false;
        }

        Call p = (Call) obj;

        return Objects.equals(p.getType().toString(), this.getType().toString());
    }

    @Override
    public int hashCode()
    {
        return Objects.hash(this.getType().toString());
    }
}
```

```java
package org.itmo.lab3;
public class Excerpt {
    public static void main(String[] args) {
        Puppy vPuppy = new Puppy(NamePerson.ALBERG,true,"black","pudel");

        Friends vChildGunnila = new Friends(NamePerson.GUNILLA);
        Friends vChildKrister = new Friends(NamePerson.KRISTER);
        Friends vChildBaby = new Friends(NamePerson.BABY);

        Adults vAdultMother = new Adults(NamePerson.MOTHER);

        //Дети стали расходиться. Гунилла и Кристер тоже ушли.
        vChildGunnila.goOut();
        vChildKrister.goOut();

        //Малыш и Альберг остались вдвоем, чему Малыш был очень рад.
        vChildBaby.stay();
        vPuppy.stay();
        vChildBaby.glade("был очень рад");
        //Он взял щенка на колени и стал ему что-то нашептывать. Щенок лизнул Малыша в лицо и заснул, сладко посапывая.
        vChildBaby.take("взял "+vPuppy+" на колени");
        vChildBaby.whisper("стал ему что-то нашептывать");
        vPuppy.lick("лизнул "+vChildBaby+ " в лицо");
        vPuppy.asleep("заснул, сладко посапывая.");
        //vPuppy.snoring();

        //Потом пришла мама из прачечной, и сразу все изменилось.
        //Малышу сделалось очень грустно: мама вовсе не считала, что Альбергу негде жить, -- она позвонила по номеру, который был выгравиров
        //и рассказала, что ее сын нашел маленького черного щенка-пуделя
        vAdultMother.come("пришла из прачечной, и сразу все изменилось.");
        vChildBaby.beSad("сделалось очень грустно");
        vAdultMother.toMakeDecisions("вовсе не считала, что "+ vPuppy+" негде жить");
        if (vAdultMother.call("по номеру, который был выгравирован на ошейнике "+vPuppy,TypeConn.MOBILE)) {
            vAdultMother.tell("сын нашел маленького черного щенка-пуделя");
        }


    }
}
```

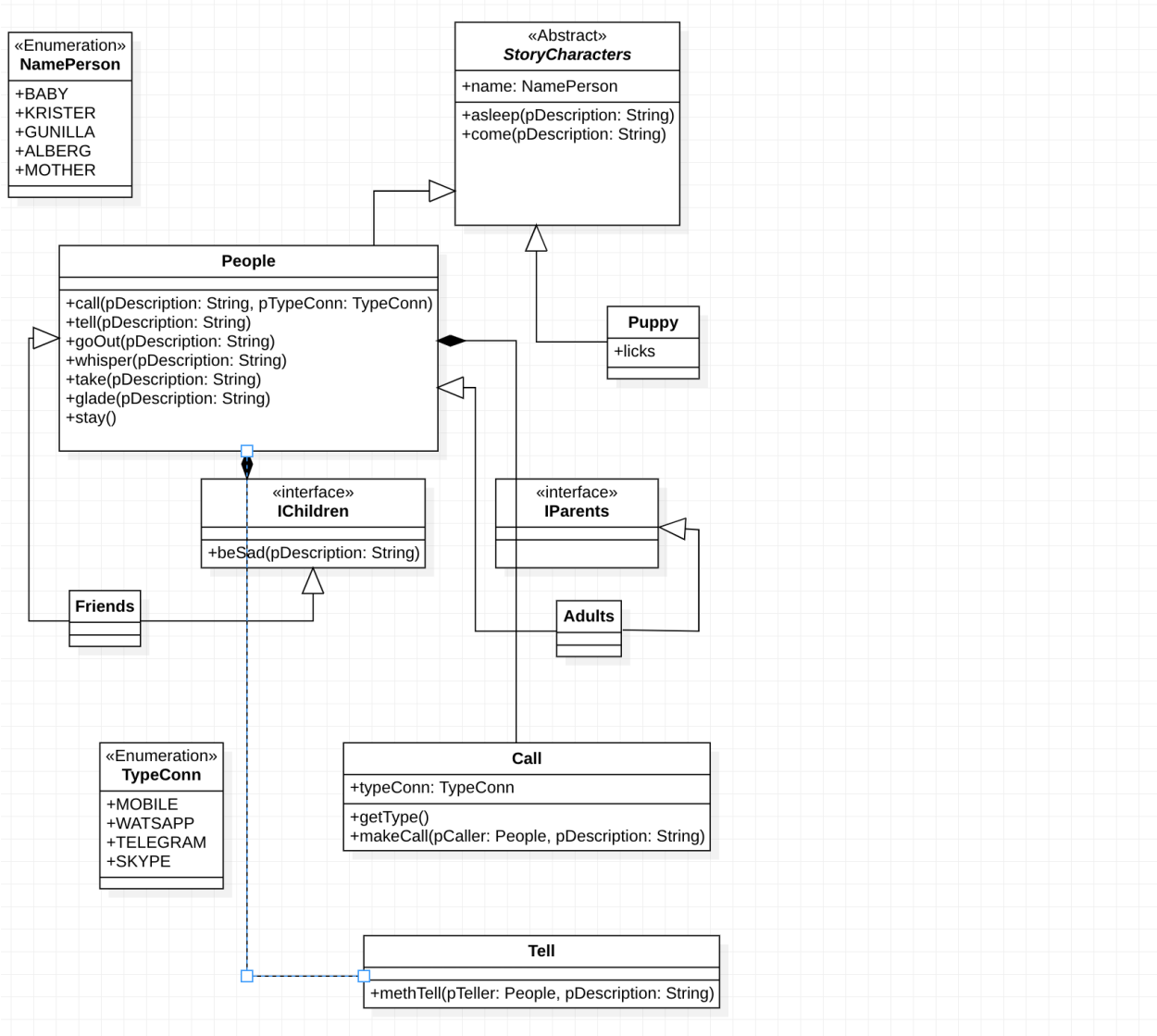
```java
package org.itmo.lab3;
import java.util.Objects;
public class Friends extends People implements IChildren {
    public Friends(NamePerson pName) { //Конструктор
        super(pName);
    }
    @Override
    public boolean equals(Object obj)
    {
        if (obj == this) {
            return true;
        }

        if (obj == null || obj.getClass() != getClass()) {
            return false;
        }

        Puppy p = (Puppy) obj;

        return Objects.equals(p.getName().toString(), super.getName().toString());
    }

    @Override
    public int hashCode()
    {
        return Objects.hash(super.getName().toString());
    }

    @Override
    public String toString() {
        return getClass().getName() + " [" + super.getName()  + "]";
    }

    public void beSad(String pDescription){
        System.out.println(this + " "+ pDescription);
    }
}
```

```java
package org.itmo.lab3;
import java.util.Objects;
public class Puppy extends StoryCharacters {
    private Boolean isCollar = false;
    private String color;
    private String breed;

    public Puppy(NamePerson pName,Boolean pIsCollar,String pColor,String pBreed){ //Конструктор
        super(pName);
        this.isCollar=pIsCollar;
        this.color=pColor;
        this.breed=pBreed;
    }
    @Override
    public boolean equals(Object obj)
    {
        if (obj == this) {
            return true;
        }

        if (obj == null || obj.getClass() != getClass()) {
            return false;
        }

        Puppy p = (Puppy) obj;

        return Objects.equals(p.getName().toString(), super.getName().toString()) &&
                p.isCollar == this.isCollar &&
                Objects.equals(p.color, this.color) &&
                Objects.equals(p.breed, this.breed);
    }
    @Override
    public int hashCode()
    {
        return Objects.hash(super.getName().toString(), this.isCollar, this.color,this.breed);
    }

    @Override
    public String toString() {
        return getClass().getName() + " [" + super.getName()  + "]";
    }

    public void lick(String pDescription) {
        System.out.println(this + " "+ pDescription);
    }

    public void goOut() {
        System.out.println(this +" ушел(ла)");
    }
```

```java
package org.itmo.lab3;
public abstract class StoryCharacters {
    private NamePerson name;

    public NamePerson getName() {
        return name;
    }

    public StoryCharacters(NamePerson pName) { //конструктор
        this.name=pName;
    }
    //@Override
    //public String toString() {
     //    return getClass().getName() + " [" + this.name  + "]";
    //}

    abstract void asleep(String pDescription);
    abstract void come(String pDescription);
    abstract void goOut();
    abstract void stay();

}
```

```java
package org.itmo.lab3;
public  enum NamePerson {
    BABY,
    GUNILLA,
    KRISTER,
    ALBERG,
    MOTHER
}
```

Диаграмма классов:



Вывод:
Познакомилась с принципами SOLID.Было получены знания как создавать и реализовывать интерфейсы. Познакомилась с абстрактными классами и принципами их реализации.