Университет ИТМО

Факультет ПИиКТ

Дисциплина: программирование

Лабораторная работа №4

Выполнил: Григорьев Александр Алексеевич,

группа P3130

Преподаватель: Блохина Елена Николаевна

г. Санкт-Петербург, 2021 год

Задание к лабораторной работе:

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

Когда Медуница и весь обслуживающий персонал вернулись в больницу, то сразу же обнаружили исчезновение Ворчуна. Они бросились в кладовую, и тут же была обнаружена пропажа двух комплектов одежды. В кладовой осталась только одежда Пульки.
Таким образом выяснился план побега, который был задуман Ворчуном и доктором Пилюлькиным. По этому плану доктор Пилюлькин должен был бежать в голом виде через окно. Злоумышленники рассчитывали, что весь персонал больницы бросится за ним в погоню, — тогда Ворчун свободно проникнет в кладовую и похитит одежду, как свою, так и Пилюлькина. План оправдался во всех деталях.
После бегства Ворчуна и Пилюлькина весь обслуживающий персонал больницы был занят лечением единственного больного — Пульки, который, видя со стороны всех такое внимание к своей особе, совсем избаловался. То он требовал, чтобы ему на обед варили суп из конфет и кашу из мармелада; то заказывал котлеты из земляники с грибным соусом, хотя каждому известно, что таких котлет не бывает; то приказывал принести яблочное пюре, а когда приносили яблочное пюре, он говорил, что просил грушевого квасу; когда же приносили квас, он говорил, что квас воняет луком, или ещё что нибудь выдумывал.
Все нянечки сбились с ног, исполняя его капризы. Они говорили, что у них спокон веку такого больного не было, что это сущее наказание, а не больной, и чтобы он выздоравливал уж поскорее, что ли.
Каждое утро он посылал одну из нянечек искать по городу свою собаку Бульку. Когда нянечка, устав шататься по городу, возвращалась в больницу в надежде, что он уже забыл о своей собаке.
Медуница видела, что характер больного день ото дня портится, и говорила, что он сделался в двадцать раз хуже Ворчуна и Пилюлькина, вместе взятых. Помочь больному могла только выписка из больницы, но нога у него все ещё болела. К тому же Пулька сам себе повредил.
Однажды, проснувшись утром, он почувствовал, что нога не болит. Вскочив с постели, он побежал по палате, но не пробежал и десяти шагов, как нога у него подвернулась, и он упал. Беднягу перенесли на руках в постель. Сразу появилась опухоль, а к вечеру подскочила температура. Медуница просидела целую ночь у его постели, не смыкая глаз. Благодаря её стараниям опухоль опала, но лечение ноги из за этого случая затянулось.

Программа должна удовлетворять следующим требованиям:

1.В программе должны быть реализованы 2 собственных класса исключений (checked и unchecked), а также обработка исключений этих классов.

2.В программу необходимо добавить использование локальных, анонимных и вложенных классов (static и non-static).

Порядок выполнения работы:

1.Доработать объектную модель приложения.

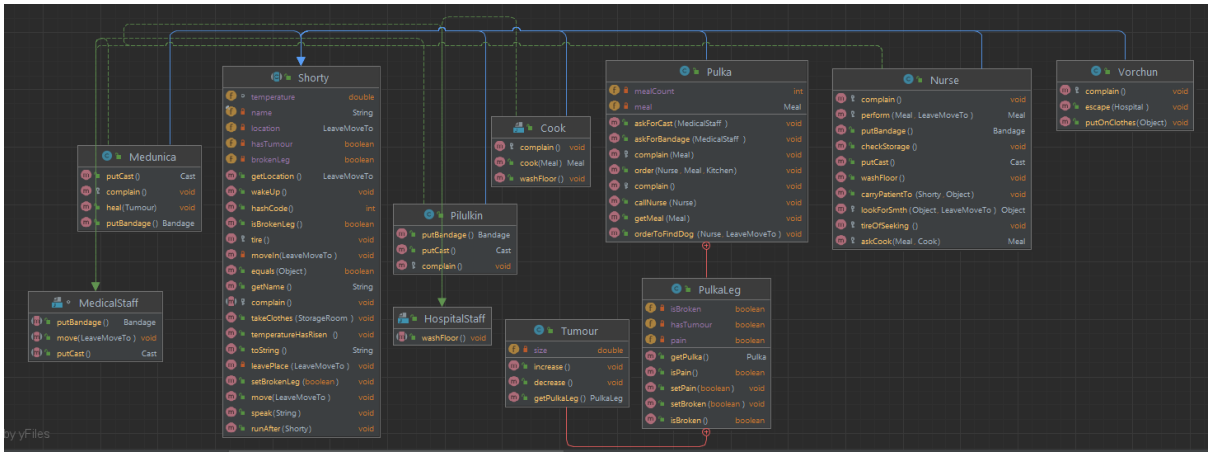
2.Перерисовать диаграмму классов в соответствии с внесёнными в модель изменениями.

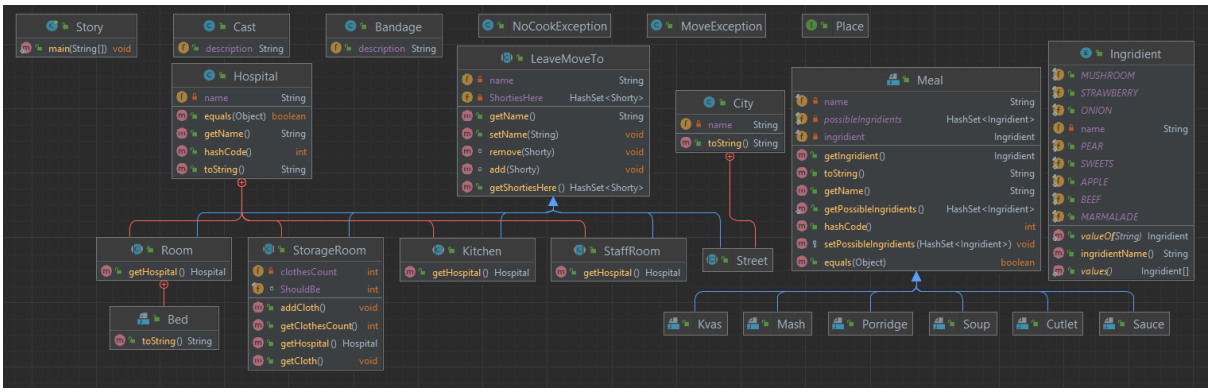
3.Согласовать с преподавателем изменения, внесённые в модель.

4.Модифицировать программу в соответствии с внесёнными в модель изменениями.

Выполнение:

Объектная модель:





Исходный код:

`Story.java`

```
import Meals.*;
import Shorties.*;
import Shorties.Pulka.PulkaLeg;
import Shorties.Pulka.PulkaLeg.Tumour;

public class Story {
    public static void main(String[] args) {
        Hospital durka = new Hospital("durka");
        Hospital.Kitchen kitchen = durka.new Kitchen();
        Hospital.StaffRoom staffRoom = durka.new StaffRoom();
        Hospital.Room room = durka.new Room();
        Hospital.StorageRoom storageRoom = durka.new StorageRoom();
        City city = new City("city");
        City.Street street = new City.Street();
        Medunica medunica = new Medunica(street);
        Nurse nurse = new Nurse(street);
        Cook cook = new Cook(street);
        Vorchun vorchun = new Vorchun(room);
        Pilulkin pilulkin = new Pilulkin(room);
        Pulka pulka = new Pulka(room);
        PulkaLeg pulkaLeg = pulka.new PulkaLeg();
```

```
        Hospital.Room.Bed bed = new Hospital.Room.Bed();
        System.out.println();

        pilulkin.move(street);
        nurse.runAfter(pilulkin);
        medunica.runAfter(pilulkin);
        cook.runAfter(pilulkin);
        vorchun.takeClothes(storageRoom);
        vorchun.move(street);
        medunica.move(room);
        nurse.move(room);
        cook.move(kitchen);
        if (room.getShortiesHere().size() < 5) {
            System.out.println("Medunica: Vorchun and Pilulkin ran away!");
        }
        nurse.checkStorage();
        System.out.println();

        pulka.callNurse(nurse);
        pulka.order(nurse, new Soup(Ingridient.SWEETS), kitchen);
        pulka.order(nurse, new Porridge(Ingridient.MARMALADE), kitchen);
        pulka.order(nurse, new Cutlet(Ingridient.STRAWBERRY), kitchen);
        pulka.order(nurse, new Mash(Ingridient.APPLE), kitchen);
        System.out.println();

        pulka.orderToFindDog(nurse, street);
        System.out.println();

        medunica.speak("Pulka's character get worse day by day, only hospital discharge can help him");
        pulka.wakeUp();
        if (!pulkaLeg.isPain()) {
            pulka.speak("I feel no pain!");
        }
        pulka.move(street);
        nurse.carryPatientTo(pulka, bed);
        Tumour tumour = pulkaLeg.new Tumour(3.4);
        pulka.riseTemperature();
        medunica.heal(tumour);
    }
}
```

LeaveMoveTo.java

```
import java.util.HashSet;

public abstract class LeaveMoveTo {
    private String name;
    private final HashSet<Shorty> ShortiesHere = new HashSet<>();

    void add(Shorty shorty) {
        ShortiesHere.add(shorty);
    }

    void remove(Shorty shorty) {
        ShortiesHere.remove(shorty);
    }

    public HashSet<Shorty> getShortiesHere() {
```

```
        return ShortiesHere;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

Hospital.java

```
import java.util.Objects;

public class Hospital {
    private final String name;

    public class Kitchen extends LeaveMoveTo {
        public Kitchen() {
            setName("kitchen");
        }

        public Hospital getHospital() {
            return Hospital.this;
        }
    }


    public class Room extends LeaveMoveTo {
        public Room() {
            setName("room");
        }
        public static class Bed{
            @Override
            public String toString() {
                return "bed";
            }
        }

        public Hospital getHospital() {
            return Hospital.this;
        }
    }


    public class StaffRoom extends LeaveMoveTo {
        public StaffRoom() {
            setName("staffRoom");
        }

        public Hospital getHospital() {
            return Hospital.this;
        }
    }
```

```
    public class StorageRoom extends LeaveMoveTo {
        private int clothesCount = 3;

        public StorageRoom() {
            setName("storageRoom");
        }

        public void addCloth() {
            clothesCount++;
        }

        public void takeCloth() {
            clothesCount--;
        }

        public int getClothesCount() {
            return clothesCount;
        }

        public Hospital getHospital() {
            return Hospital.this;
        }

    }


    public Hospital(String hospName) {
        name = hospName;
    }

    public String getName() {
        return name;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Hospital)) return false;
        Hospital hospital = (Hospital) o;
        return Objects.equals(name, hospital.name);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name);
    }

    @Override
    public String toString() {
        return name;
    }
}
```

City.java

```
public class City{
    private final String name;

    public static class Street extends LeaveMoveTo{
```

```
    }

    public City(String cityName) {
        name = cityName;
    }

    @Override
    public String toString() {
        return name;
    }
}
```

Shorty.java

```
import Shorties.Hospital.StorageRoom;
import utility.MoveException;
import java.util.Objects;
import java.lang.*;

public abstract class Shorty {
    private final String name;
    private LeaveMoveTo location;
    private boolean brokenLeg = false;
    double temperature = 36.6;

    public Shorty(String n) {
        name = n;
        location = null;
    }

    public Shorty(String n, LeaveMoveTo notDefault) {
        name = n;
        location = notDefault;
        moveIn(notDefault);
    }

    public void move(LeaveMoveTo newLocation) {
        if (newLocation != location) {
            try {
                leavePlace(location);
//                if (newLocation.getClass().getEnclosingClass() != Hospital.class) {
//                    System.out.println(getName() + " escapes from Hospital");
//                }
                if (location.getClass().getEnclosingClass() == City.class) {
                    System.out.println(this + " returned to Hospital");
                }
                if (location.getClass() == StorageRoom.class) {
                    if (this instanceof MedicalStaff) {
                        System.out.println(this + " went to Storage Room");
                    }
//                    else{
//                        System.out.println(this + " sneaked to Storage Room");
//                    }
                }
                if (newLocation.getClass().getEnclosingClass() == City.class) {
                    System.out.println(this + " ran to city");
                }
```

```
            moveIn(newLocation);
            location = newLocation;
        } catch (MoveException ex) {
            System.out.println(this + " tried to run, but he twisted his leg and fell!");
        }
    }
}

private void leavePlace(LeaveMoveTo location) throws MoveException {
    if (isBrokenLeg()) {
        throw new MoveException();
    } else {
        location.remove(this);
    }
}

private void moveIn(LeaveMoveTo location) {
    location.add(this);
}

public void riseTemperature() {
    temperature++;
    System.out.println(this + "'s temperature has risen");
}

public void decreaseTemperature() {
    temperature--;
    System.out.println(this + "'s temperature has risen");
}

public void speak(String text) {
    System.out.println(this + ": " + text);
}

public void wakeUp() {
    System.out.println(this + " woke up");
}

public String getName() {
    return name;
}

public LeaveMoveTo getLocation() {
    return location;
}

protected abstract void complain();

protected void tire() {
    System.out.println(this.getName() + ": I'm incredibly tired!");
}

public void runAfter(Shorty shorty) {
    System.out.println(this + " ran after " + shorty);
}

public void takeClothes(StorageRoom storageRoom) {
    move(storageRoom);
```

```
        storageRoom.takeCloth();
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Shorty)) return false;
        Shorty shorty = (Shorty) o;
        return Objects.equals(name, shorty.name);
    }

    @Override
    public int hashCode() {
        return name.hashCode();
    }

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

    public boolean isBrokenLeg() {
        return brokenLeg;
    }

    public void setBrokenLeg(boolean brokenLeg) {
        this.brokenLeg = brokenLeg;
    }
}
```

Pulka.java

```
public class Pulka extends Shorty {
    private Meal meal;
    private int mealCount = 0;

    public class PulkaLeg {
        private boolean isBroken;
        private boolean hasTumour = false;
        private boolean pain = false;

        public class Tumour {
            private double size;

            public Tumour(double s) {
                size = s;
                PulkaLeg.this.hasTumour = true;
                PulkaLeg.this.pain = true;
                System.out.println("Tumour has appeared");
            }

            public void increase() {
                size++;
                System.out.println(Pulka.this + "'s tumour increased");
            }
```

```
        public void decrease() {
            size--;
            System.out.println(Pulka.this + "'s tumour decreased");
        }

        public PulkaLeg getPulkaLeg() {
            return PulkaLeg.this;
        }
    }

    public PulkaLeg() {
        isBroken = true;
        Pulka.this.setBrokenLeg(true);
    }

    public void setBroken(boolean broken) {
        isBroken = broken;
    }

    public boolean isBroken() {
        return isBroken;
    }

    public Pulka getPulka() {
        return Pulka.this;
    }

    public boolean isPain() {
        return pain;
    }

    public void setPain(boolean pain) {
        this.pain = pain;
    }
}

public Pulka() {
    super("Pulka");
    setBrokenLeg(true);
}

public Pulka(LeaveMoveTo notDefault) {
    super("Pulka", notDefault);
    setBrokenLeg(true);
}

public void callNurse(Nurse nurse) {
    nurse.move(this.getLocation());
}

public void order(Nurse nurse, Meal meal, Hospital.Kitchen kitchen) {
    callNurse(nurse);
    System.out.println("Pulka: Cook me " + meal);
    receiveMeal(nurse.perform(meal, kitchen));
    nurse.move(getLocation());
    if (mealCount == 5) {
        nurse.complain();
        nurse.tire();
```

```
        }
        if (mealCount == 4) {
            order(nurse, new Kvas(Ingridient.PEAR), kitchen);
            nurse.move(getLocation());
        }
        //complain(this.meal);
    }

    public void receiveMeal(Meal meal) {
        System.out.println("Pulka gets " + meal);
        this.meal = meal;
        mealCount++;
        if (mealCount == 4 || mealCount == 5) {
            complain(meal);
        }
    }

    public void orderToFindDog(Nurse nurse, LeaveMoveTo where) {
        class Dog {
            final String name;

            Dog(String n) {
                name = n;
            }
        }
        Dog bulka = new Dog("Bul'ka");
        System.out.println(this + " find my dog Bul'ka");
        nurse.lookForSmth(bulka, where);
    }

    public void askForBandage(MedicalStaff medic) {
        medic.move(this.getLocation());
        medic.putBandage();
    }

    public void askForCast(MedicalStaff medic) {
        medic.move(this.getLocation());
        medic.putCast();
    }

    @Override
    protected void complain() {
        System.out.println("This hospital is awful.");
    }

    protected void complain(Meal meal) {
        if (mealCount == 5) {
            System.out.println("This " + meal.getName() + " stinks of onion");
        }
        if (mealCount == 4) {
            System.out.println("I asked for Pear Kvas");
        }
    }
}
```

Nurse.java

```
import Meals.Meal;
import utility.*;


public class Nurse extends Shorty implements HospitalStaff, MedicalStaff {


    public Nurse() {
        super("Nurse");
    }

    public Nurse(LeaveMoveTo notDefault) {

        super("Nurse", notDefault);
    }

    protected Meal perform(Meal meal, LeaveMoveTo kitchen) {
        boolean f = false;
        move(kitchen);
        for (Shorty shorty : kitchen.getShortiesHere()) {
            if (shorty instanceof Cook) {
                f = true;
                askCook(meal, (Cook) shorty);
            }
        }
        if (!f) {
            throw new NoCookException();
        } else {
            return meal;
        }
    }

    protected Object lookForSmth(Object o, LeaveMoveTo where){
        LeaveMoveTo loc = getLocation();
        move(where);
        tireOfSeeking();
        move(loc);
        return null;
    }

    public void checkStorage(){
        System.out.println(this + ": two clothes missing!");
    }

    public void carryPatientTo(Shorty patient, Object o){
        System.out.println(patient + " was carried to " + o);
    }

    protected void tireOfSeeking(){
        System.out.println(this + ": I'm tired of walking. I hope he's forgotten about it");
    }

    protected void askCook(Meal meal, Cook c) {
       // System.out.println("Nurse: Cook "+ meal.getIngridient().ingridientName() + " " + meal.getName());
        c.cook(meal);
    }

    @Override
```

```
    public void washFloor() {
        move(getLocation());
        System.out.println(this.getName() + " washes the floor in StaffRoom");
    }

    @Override
    public Bandage putBandage() {
        return new Bandage();
    }

    @Override
    public Cast putCast() {
        return new Cast();
    }

    @Override
    protected void complain(){
        System.out.println("What a terrible patient! He'd rather get well soon");
    }
}
```

Cook.java

```
public class Cook extends Shorty implements HospitalStaff{
    public Cook() {
        super("Cook");
    }

    public Cook(LeaveMoveTo notDefault) {
        super("Cook", notDefault);
    }

    public Meal cook(Meal meal){
        if (!(Meal.getPossibleIngridients().contains(meal.getIngridient()))){
            this.complain();
        }
        return meal;
    }

    @Override
    protected void complain() {
        System.out.println("This meal doesn't exist");
    }

    @Override
    public void washFloor(){
        move(this.getLocation());
        System.out.println(this.getName() + " washes the floor in Kitchen");
    }
}
```

Pilulkin.java

```
import utility.Bandage;
import utility.Cast;

public class Pilulkin extends Shorty implements MedicalStaff {
    public Pilulkin() {
```

```
        super("Pilulkin");
    }

    public Pilulkin(LeaveMoveTo notDefault) {
        super("Pilulkin", notDefault);
    }

    @Override
    public Bandage putBandage() {
        return new Bandage();
    }

    @Override
    public Cast putCast() {
        return new Cast();
    }

    @Override
    protected void complain() {
    }
}
```

Vorchun.java

```
public class Vorchun extends Shorty {
    public Vorchun() {
        super("Vorchun");
    }

    public Vorchun(LeaveMoveTo notDefault) {

        super("Vorchun", notDefault);
    }

    @Override
    public void takeClothes(Hospital.StorageRoom storageRoom) {
        System.out.println(this + " sneaked to Storage room");
        super.takeClothes(storageRoom);
        storageRoom.takeCloth();
        System.out.println(this + " took clothes");
    }



    @Override
    protected void complain() {

    }
}
```

Ingridient.java

```
public enum Ingridient {
    STRAWBERRY("Strawberry"), SWEETS("Sweets"), MARMALADE("Marmalade"),
    MUSHROOM("Mushroom"), APPLE("Apple"),
    PEAR("Pear"), ONION("Onion"), BEEF("Beef");
    private String name;

    Ingridient(String ingridientName) {
```

```
        this.name = ingridientName;
    }

    public String ingridientName() {
        return name;
    }
}
```

Cast.java

```
public class Cast {
    public String description = "гипс";
    public Cast(){
        description = "Гипс";
    }
}
```

Bandage.java

```
public class Bandage {
    public String description = "бинт";

    public Bandage() {
        description = "Бинт";
    }
}
```

Meal.java

```
import java.util.HashSet;
import java.util.Objects;

public abstract class Meal {
    private final Ingridient ingridient;
    private final String name;
    private static final HashSet<Ingridient> possibleIngridients = new HashSet<>();

    public Meal(String n, Ingridient ing) {
        name = n;
        ingridient = ing;
    }

    public String getName() {
        return name;
    }

    public Ingridient getIngridient() {
        return ingridient;
    }

    protected void setPossibleIngridients(HashSet<Ingridient> ingridients){
        for (Ingridient ing: ingridients){
            possibleIngridients.add(ing);
        }
    }

    public static HashSet<Ingridient> getPossibleIngridients() {
        return possibleIngridients;
    }

    @Override
    public boolean equals(Object o) {
```

```
        if (this == o) return true;
        if (!(o instanceof Meal)) return false;
        Meal meal = (Meal) o;
        return ingridient == meal.ingridient && Objects.equals(name, meal.name);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name);
    }

    @Override
    public String toString() {
        return getIngridient().ingridientName() + " " +getName();
    }
}
```

Cutlet.java

```
import java.util.HashSet;

public class Cutlet extends Meal {
    public Cutlet(Ingridient ing) {
        super("Cutlet", ing);
        HashSet<Ingridient> ingridients = new HashSet<>();
        ingridients.add(Ingridient.BEEF);
        setPossibleIngridients(ingridients);
    }
}
```

Soup.java

```
import java.util.HashSet;

public class Soup extends Meal {
    public Soup(Ingridient ing) {
        super("Soup", ing);
        HashSet<Ingridient> ingridients = new HashSet<>();
        ingridients.add(Ingridient.SWEETS);
        setPossibleIngridients(ingridients);
    }
}
```

Porridge.java

```
import java.util.HashSet;

public class Porridge extends Meal {
    public Porridge(Ingridient ing) {
        super("Porridge", ing);
        HashSet<Ingridient> ingridients = new HashSet<>();
        ingridients.add(Ingridient.MARMALADE);
        setPossibleIngridients(ingridients);
    }
}
```

Mash.java

```
import java.util.HashSet;

public class Mash extends Meal {
    public Mash(Ingridient ing) {
        super("Mash", ing);
```

```
        HashSet<Ingridient> ingridients = new HashSet<>();
        ingridients.add(Ingridient.APPLE);
        ingridients.add(Ingridient.PEAR);
        setPossibleIngridients(ingridients);
    }
}
```

MedicalStaff.java

```
interface MedicalStaff {
    Bandage putBandage();

    Cast putCast();

    void move(Location newLocation, Hospital hosp);
}
```

HospitalStaff.java

```
interface HospitalStaff {
    void washFloor();
}
```

Результат выполнения программы:

```
Pilulkin ran to city
Nurse ran after Pilulkin
Medunica ran after Pilulkin
Cook ran after Pilulkin
Vorchun sneaked to Storage room
Vorchun took clothes
Vorchun ran to city
Medunica returned to Hospital
Nurse returned to Hospital
Cook returned to Hospital
Medunica: Vorchun and Pilulkin ran away!
Nurse: two clothes missing!

Pulka: Cook me Sweets Soup
Pulka gets Sweets Soup
Pulka: Cook me Marmalade Porridge
Pulka gets Marmalade Porridge
Pulka: Cook me Strawberry Cutlet
This meal doesn't exist
Pulka gets Strawberry Cutlet
Pulka: Cook me Apple Mash
Pulka gets Apple Mash
I asked for Pear Kvas
Pulka: Cook me Pear Kvas
Pulka gets Pear Kvas
This Kvas stinks of onion
What a terrible patient! He'd rather get well soon
Nurse: I'm incredibly tired!

Pulka find my dog Bul'ka
Nurse ran to city
Nurse: I'm tired of walking. I hope he's forgotten about it
Nurse returned to Hospital

Medunica: Pulka's character get worse day by day, only hospital discharge can help him
Pulka woke up
Pulka: I feel no pain!
```

Pulka tried to run, but he twisted his leg and fell!
Pulka was carried to bed
Tumour has appeared
Pulka's temperature has risen
Medunica was sitting and healing Pulka all night
Pulka's tumour decreased
Pulka's temperature has risen

Вывод: Во время выполнения данной лабораторной работы я научился применять вложенные классы и исключения.