

Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники
Направление подготовки 09.03.04 Программная инженерия
Дисциплина «Программирование»

По лабораторной работе №3
Вариант 2705

Студент
Александров А. А.
Р3106

Преподаватель
Письмак А.Е.

Санкт-Петербург 2023

Содержание

Задание	4
Текст задания	4
Программа должна удовлетворять следующим требованиям:	4
Порядок выполнения работы:	4
UML диаграмма	5
Код программы	6
Main.java	6
LittleMen.java	7
Dunno.java	9
Pilulkin.java	9
Интерфейсы - Heartbeat.java, Sniff.java, Sniffing.java, TryingToWake.java	10
Абстрактный класс - DunnoStatus.java	10
Enum класс - Status.java	11
Результат выполнения программы	12
Заключение	13

Задание

Текст задания

Увидев, что Незнайка снова закрыл глаза, доктор Пилюлькин принялся трясти его за плечо. Заметив, что лицо Незнайки заливают какая-то странная бледность, Пилюлькин снова схватил его за руку. Пульс не прощупывался. Пилюлькин прижался ухом к груди Незнайки. Биения сердца не слышалось. Он снова дал понюхать Незнайке нашатырного спирта, но это не произвело никакого действия.

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.

UML диаграмма



Код программы

Main.java

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

        System.out.println("Введите количество съеденных груш Незнайкой: ");
        Scanner scanner = new Scanner(System.in);
        int amountOfPears = scanner.nextInt();

        Dunno dunno = new Dunno("Незнайка", "Пострадавший", "Ел груши", amountOfPears);
        Pilulkin pilulkin = new Pilulkin("Пилюлькин", "Доктор", "Не ел груши");

        if (dunno.getAmtOfEatenPears() < 0) {
            System.out.println("Ну нет! Ты не можешь съесть отрицательное количество груш");
        }

        if (dunno.getAmtOfEatenPears() == 0) {
            dunno.setStatus(Status.HUNGRY, "Незнайка");
        }

        if (dunno.getAmtOfEatenPears() < 5 && dunno.getAmtOfEatenPears() > 0) {
            dunno.setStatus(Status.NORMAL, "Незнайка");
        }

        if (dunno.getAmtOfEatenPears() > 4 && dunno.getAmtOfEatenPears() < 8) {
            dunno.setStatus(Status.CLOSEYES, "Незнайка");
            pilulkin.Shaking("Пилюлькин", "Незнайку");
            dunno.setStatus(Status.WAKEUP, "Незнайка");
        }

        if (dunno.getAmtOfEatenPears() > 7 && dunno.getAmtOfEatenPears() < 12) {
            dunno.setStatus(Status.CLOSEYES, "Незнайка");
            pilulkin.Shaking("Пилюлькин", "Незнайку");
            dunno.setStatus(Status.WHITEFACE, "Незнайка");
            pilulkin.GrabHand("Пилюлькин", "Незнайку");
            pilulkin.CheckingPulse();
            dunno.setStatus(Status.WEAKPULSE, "У Незнайки");
            pilulkin.getInfoAboutPulse();
            pilulkin.ListenHeartbeat("Пилюлькин", "Незнайки");
        }
    }
}
```

```

        dunno.setStatus(Status.NOHEARTBEAT, "У Незнайки");
        pilulkin.getInfoAboutHeartbeat();
        pilulkin.GiveToSniff("Пилюлькин", "Незнайке");
        dunno.Sniffing("Незнайка");
        dunno.setStatus(Status.WAKEUP, "Незнайка");
    }

    if (dunno.getAmtOfEatenPears() > 11) {
        dunno.setStatus(Status.CLOSEYES, "Незнайка");
        pilulkin.Shaking("Пилюлькин", "Незнайку");
        dunno.setStatus(Status.WHITEFACE, "Незнайка");
        pilulkin.GrabHand("Пилюлькин", "Незнайку");
        pilulkin.CheckingPulse();
        dunno.setStatus(Status.WEAKPULSE, "У Незнайки");
        pilulkin.getInfoAboutPulse();
        pilulkin.ListenHeartbeat("Пилюлькин", "Незнайки");
        dunno.setStatus(Status.NOHEARTBEAT, "У Незнайки");
        pilulkin.getInfoAboutHeartbeat();
        pilulkin.GiveToSniff("Пилюлькин", "Незнайке");
        dunno.Sniffing("Незнайка");
        dunno.setStatus(Status.DEATH, "Незнайка");
    }
}
}
}

```

LittleMen.java

```

import java.util.Objects;

public class LittleMen extends DunnoStatus {
    private String name;
    private String role;
    private String ans;
    private int amtOfEatenPears;

    public LittleMen(String name, String role, String ans, int amtOfEatenPears) {
        this.name = name;
        this.role = role;
        this.ans = ans;
        this.amtOfEatenPears = amtOfEatenPears;
    }

    public LittleMen(String name, String role, String ans) {

```

```

        this.name = name;
        this.role = role;
        this.ans = ans;
    }

    @Override
    public String toString(){
        return "LittleMan{" +
            "Name '" + name + '\'' +
            ", Role '" + role + '\'' +
            ", Answer '" + ans + '\'' +
            ", Amount of eaten pears = " + amtOfEatenPears +
            "}";
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        LittleMen shorty = (LittleMen) obj;
        return this.ans == shorty.ans;
    }

    @Override
    public int hashCode(){
        System.out.print(Objects.hashCode(ans));
        return Objects.hashCode(ans);
    }

    @Override
    public void setStatus(Status DunnoStat, String name){
        System.out.println(name + " " + DunnoStat.getStringStatus());
    }

    @Override
    public int getAmtOfEatenPears(){
        return amtOfEatenPears;
    }
}

```

Dunno.java

```
public class Dunno extends LittleMen implements Sniffing{
    Dunno(String name, String role, String ans, int amtOfEatenPears){
        super(name, role, ans, amtOfEatenPears);
    }

    @Override
    public void Sniffing(String name){
        System.out.println(name + " вдыхает нашатырь... и...");
    }
}
```

Pilulkin.java

```
public class Pilulkin extends LittleMen implements Heartbeat, Sniff, TryingToWake{
    Pilulkin(String name, String role, String ans){
        super(name, role, ans);
    }

    @Override
    public void Shaking(String name1, String name2){
        System.out.println(name1 + " трясет " + name2 + " за плечо");
    }

    @Override
    public void GrabHand(String name1, String name2){
        System.out.print(name + " хватает " + name2 + " за руку ");
    }

    @Override
    public void CheckingPulse(){
        System.out.println("и проверяет пульс");
    }

    @Override
    public void ListenHeartbeat(String name1, String name2){
        System.out.println(name1 + " слушает сердцебиение " + name2);
    }

    @Override
    public void getInfoAboutPulse(){
```



```

        System.out.println("Черт побери! Нет пульса!!");
    }

    @Override
    public void getInfoAboutHeartbeat() {
        System.out.println("Боже мой?!?!? И сердцебиения не слышно?!?!?!?!");
    }

    @Override
    public void GiveToSniff(String name1, String name2) {
        System.out.println(name1 + " решает дать " + name2 + " понюхать нашатырь ");
    }
}

```

Интерфейсы - Heartbeat.java, Sniff.java, Sniffing.java, TryingToWake.java

```

public interface Heartbeat {
    void CheckingPulse();
    void ListenHeartbeat(String name1, String name2);
    void getInfoAboutPulse();
    void getInfoAboutHeartbeat();
}

public interface Sniff{
    void GiveToSniff(String name1, String name2);
}

public interface Sniffing {
    void Sniffing(String name);
}

public interface TryingToWake {
    void Shaking(String name);
    void GrabHand(String name);
}

```

Абстрактный класс - DunnoStatus.java

```

abstract class DunnoStatus {
    abstract void setStatus(Status DunnoStat, String name);
    abstract int getAmtOfEatenPears();
}

```

Enum класс - Status.java

```
public enum Status {  
    CLOSEYES("отключился"),  
    WHITEFACE("побледнел"),  
    WEAKPULSE("слабый пульс"),  
    NOHEARTBEAT("нет сердцебиения"),  
    DEATH("вмер Ж(, нашатырь не помог =("),  
    WAKEUP("очнулся!!!!"),  
    NORMAL("наелся и чувствует себя превосходно"),  
    HUNGRY("ничего не съел :(");  
  
    private String s;  
  
    Status(String s){  
        this.s = s;  
    }  
  
    public String getStringStatus(){  
        return s;  
    }  
}
```

Результат выполнения программы

Введите количество съеденных груш Незнайкой:

12

Незнайка отключился

Пилюлькин трясет Незнайку за плечо

Незнайка побледнел

Пилюлькин хватает Незнайку за руку и проверяет пульс

У Незнайки слабый пульс

Черт побери! Нет пульса!!

Пилюлькин слушает сердцебиение Незнайки

У Незнайки нет сердцебиения

Боже мой?!?!? И сердцебиения не слышно?!?!?!?!?

Пилюлькин решает дать Незнайке понюхать нашатырь

Незнайка вдыхает нашатырь... и...

Незнайка вмер Ж(, нашатырь не помог =(

Заключение

По итогу выполнения Лабораторной работы №3 я расширил свои знания по ООП в Java, научившись применять абстрактные и enum классы, интерфейсы.