«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет программной инженерии и компьютерной техники

Отчет

По лабораторной работе №3

Вариант 290005

Студент:

*Ильин Н. С.*

*Р3110 поток 2.9*

Преподаватель:

*Наумова Н. А.*

Санкт-Петербург, 2023 г.

**Оглавление**

[Задание: 3](#_Toc145408419)

[Выполнение работы: 3](#_Toc145408420)

[Выводы: 13](#_Toc145408421)

# **Задание:**

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

По правую руку от Короля находился Белый Кролик, с трубой в одной лапке и пергаментным свитком в другой. А в самом центре судебного зала стоял стол, а на нем красовалось большое блюдо с пирожками, н вид у них был такой аппетитный, что у Алисы прямо слюнки потекли. "Хорошо бы, суд уже кончился и позвали к столу!" - подумала она.

**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.

# **Выполнение работы:**

Исходный код:

#### src/Main.java

  1 │ **import** characters**.Alice;**  
  2 │ **import** characters**.King;**  
  3 │ **import** characters**.WhiteRabbit;**  
  4 │ **import** characters**.**movement**.Direction;**  
  5 │ **import** food**.Pie;**  
  6 │ **import** props**.\*;**  
  7 │ **import** furniture**.Table;**  
  8 │ **import** rooms**.Courtroom;**  
  9 │   
 10 │   
 11 │ **public** **class** **Main** **{**  
 12 │  **public** **static** **void** main**(String[]** args**)** **{**  
 13 │  **System.**out**.**println**(new** **String(**"123"**)** **==** **new** **String(**"123"**));**  
 14 │  **System.**out**.**println**(new** **String(**"123"**)** **==** "123"**);**  
 15 │  **System.**out**.**println**((**"12"**+**"3"**).**intern**()** **==** "123"**);**  
 16 │  **System.**out**.**println**((**"12" **+** "3"**)** **==** "123"**);**  
 17 │  **Alice** alice **=** **new** **Alice(**"Алиса"**);**  
 18 │  **King** king **=** **new** **King(**"Король"**);**  
 19 │  **WhiteRabbit** whiteRabbit **=** **new** **WhiteRabbit(**"Белый Кролик"**);**  
 20 │  **Courtroom** courtroom **=** **new** **Courtroom(**"Судебный зал"**);**  
 21 │  **Table** table **=** **new** **Table(**"Стол"**);**  
 22 │  **Dish** dish **=** **new** **Dish(**"Большое блюдо с пирожками"**,** 10**);**  
 23 │  dish**.**addFood**(new** **Pie(**"Аппетитный пирожок"**,** "Курица"**));**  
 24 │  dish**.**addFood**(new** **Pie(**"Аппетитный пирожок"**,** "Рыба"**));**  
 25 │  dish**.**addFood**(new** **Pie(**"Аппетитный пирожок"**,** "Картошка"**));**  
 26 │   
 27 │   
 28 │  whiteRabbit**.**standBy**(**king**,** **Direction.**BY\_RIGHT\_HAND**);**  
 29 │  whiteRabbit**.**hold**(new** **Trumpet(**"Труба"**));**  
 30 │  whiteRabbit**.**hold**(new** **Scroll(**"Пергаментный свиток"**));**  
 31 │  courtroom**.**setCenterObject**(**table**);**  
 32 │  table**.**setTopObject**(**dish**);**  
 33 │  dish**.**lookLike**(**"аппетитно"**);**  
 34 │  alice**.**salivateAt**(**dish**);**  
 35 │  alice**.**think**(**"Хорошо бы, суд уже кончился и позвали к столу!"**);**  
 36 │  ***//alice.walkTo(Direction.CENTER);***  
 37 │  **}**  
 38 │ **}**

#### src/rooms/Courtroom.java

  1 │ **package** rooms**;**  
  2 │   
  3 │ **import** furniture**.\*;**  
  4 │   
  5 │ **public** **class** **Courtroom** **extends** **Room** **{**  
  6 │  **public** **Courtroom(String** name**)** **{**  
  7 │  **super(**name**);**  
  8 │  **}**  
  9 │   
 10 │  **public** **void** setCenterObject**(Table** object**){**  
 11 │  **System.**out**.**println**(**"В центре " **+** getName**()** **+** " стоит " **+** object**.**getName**());**  
 12 │  **}**  
 13 │   
 14 │   
 15 │ **}**

#### src/rooms/Room.java

  1 │ **package** rooms**;**  
  2 │   
  3 │ **import** interfaces**.Nameable;**  
  4 │   
  5 │ **public** **abstract** **class** **Room** **implements** **Nameable** **{**  
  6 │  **private** **String** name**;**  
  7 │  **public** **Room(String** name**){**  
  8 │  **this.**name **=** name**;**  
  9 │  **}**  
 10 │  **public** **String** getName**(){**  
 11 │  **return** **this.**name**;**  
 12 │  **}**  
 13 │   
 14 │  @Override  
 15 │  **public** **boolean** equals**(Object** obj**)** **{**  
 16 │  **if** **(this** **==** obj**)** **return** true**;**  
 17 │  **if** **(**obj **==** null **||** getClass**()** **!=** obj**.**getClass**())** **return** false**;**  
 18 │  **Room** room **=** **(Room)** obj**;**  
 19 │  **return** name**.**equals**(**room**.**name**);**  
 20 │  **}**  
 21 │   
 22 │  @Override  
 23 │  **public** **int** hashCode**()** **{**  
 24 │  **return** name**.**hashCode**();**  
 25 │  **}**  
 26 │  @Override  
 27 │  **public** **String** toString**()** **{**  
 28 │  **return** "Room{" **+**  
 29 │  "name='" **+** name **+** '\'' **+**  
 30 │  '}'**;**  
 31 │  **}**  
 32 │ **}**  
 33 │

#### src/props/Dish.java

  1 │ **package** props**;**  
  2 │   
  3 │   
  4 │ **import** food**.Food;**  
  5 │   
  6 │ **import** java**.**util**.Arrays;**  
  7 │   
  8 │ **public** **class** **Dish** **extends** **Prop** **{**  
  9 │  **private** **Food[]** foodItems**;**  
 10 │  **private** **int** itemCount**;**  
 11 │   
 12 │  **public** **Dish(String** name**,** **int** maxItemCount**)** **{**  
 13 │  **super(**name**);**  
 14 │  **this.**foodItems **=** **new** **Food[**maxItemCount**];**  
 15 │  **this.**itemCount **=** 0**;**  
 16 │  **}**  
 17 │   
 18 │  **public** **void** addFood**(Food** foodItem**)** **{**  
 19 │  **if** **(**itemCount **<** foodItems**.**length**)** **{**  
 20 │  foodItems**[**itemCount**]** **=** foodItem**;**  
 21 │  itemCount**++;**  
 22 │  **}** **else** **{**  
 23 │  **System.**out**.**println**(**"На блюде больше нет места."**);**  
 24 │  **}**  
 25 │  **}**  
 26 │   
 27 │  **public** **Food[]** getFoodItems**()** **{**  
 28 │  **return** foodItems**;**  
 29 │  **}**  
 30 │ **}**

#### src/props/Prop.java

  1 │ **package** props**;**  
  2 │   
  3 │ **import** food**.\*;**  
  4 │ **import** interfaces**.Nameable;**  
  5 │   
  6 │ **public** **abstract** **class** **Prop** **implements** **Nameable** **{**  
  7 │  **protected** **String** name**;**  
  8 │   
  9 │  **public** **Prop(String** name**)** **{**  
 10 │  **this.**name **=** name**;**  
 11 │  **}**  
 12 │   
 13 │  **public** **String** getName**()** **{**  
 14 │  **return** **this.**name**;**  
 15 │  **}**  
 16 │  **public** **void** lookLike**(String** description**)** **{**  
 17 │  **System.**out**.**println**(**getName**()** **+** " выглядит " **+** description**);**  
 18 │  **}**  
 19 │  **public** **boolean** equals**(Object** obj**)** **{**  
 20 │  **if** **(this** **==** obj**)** **return** true**;**  
 21 │  **if** **(**obj **==** null **||** getClass**()** **!=** obj**.**getClass**())** **return** false**;**  
 22 │  **Prop** prop **=** **(Prop)** obj**;**  
 23 │  **return** name**.**equals**(**prop**.**name**);**  
 24 │  **}**  
 25 │   
 26 │  **public** **int** hashCode**()** **{**  
 27 │  **return** name**.**hashCode**();**  
 28 │  **}**  
 29 │   
 30 │  **public** **String** toString**()** **{**  
 31 │  **return** "Prop{" **+**  
 32 │  "name='" **+** name **+** '\'' **+**  
 33 │  '}'**;**  
 34 │  **}**  
 35 │ **}**  
 36 │

#### src/props/Scroll.java

  1 │ **package** props**;**  
  2 │   
  3 │ **public** **class** **Scroll** **extends** **Prop** **{**  
  4 │  **private** **String** text **=** "..."**;**  
  5 │  **public** **Scroll(String** name**){**  
  6 │  **super(**name**);**  
  7 │  **}**  
  8 │   
  9 │  **public** **String** getText**()** **{**  
 10 │  **return** text**;**  
 11 │  **}**  
 12 │   
 13 │  **public** **void** setText**(String** text**)** **{**  
 14 │  **this.**text **=** text**;**  
 15 │  **}**  
 16 │ **}**  
 17 │

#### src/props/Trumpet.java

  1 │ **package** props**;**  
  2 │   
  3 │ **public** **class** **Trumpet** **extends** **Prop** **{**  
  4 │  **public** **Trumpet(String** name**){**  
  5 │  **super(**name**);**  
  6 │  **}**  
  7 │   
  8 │  **public** **void** makeSound**(){**  
  9 │  **System.**out**.**println**(**getName**()** **+** " издаёт звук"**);**  
 10 │  **}**  
 11 │ **}**  
 12 │

#### src/interfaces/Nameable.java

 1 │ **package** interfaces**;**  
 2 │   
 3 │ **public** **interface** **Nameable** **{**  
 4 │  **String** getName**();**  
 5 │ **}**

#### src/interfaces/Thinkable.java

 1 │ **package** interfaces**;**  
 2 │   
 3 │ **public** **interface** **Thinkable** **{**  
 4 │  **void** think**(String** thought**);**  
 5 │ **}**

#### src/furniture/Furniture.java

  1 │ **package** furniture**;**  
  2 │   
  3 │ **import** interfaces**.Nameable;**  
  4 │   
  5 │ **public** **abstract** **class** **Furniture** **implements** **Nameable** **{**  
  6 │  **protected** **String** name**;**  
  7 │   
  8 │  **public** **Furniture(String** name**)** **{**  
  9 │  **this.**name **=** name**;**  
 10 │  **}**  
 11 │   
 12 │  **public** **String** getName**()** **{**  
 13 │  **return** **this.**name**;**  
 14 │  **}**  
 15 │   
 16 │  **public** **boolean** equals**(Object** obj**)** **{**  
 17 │  **if** **(this** **==** obj**)** **return** true**;**  
 18 │  **if** **(**obj **==** null **||** getClass**()** **!=** obj**.**getClass**())** **return** false**;**  
 19 │  **Furniture** furniture **=** **(Furniture)** obj**;**  
 20 │  **return** name**.**equals**(**furniture**.**name**);**  
 21 │  **}**  
 22 │   
 23 │  **public** **int** hashCode**()** **{**  
 24 │   
 25 │  **return** name**.**hashCode**();**  
 26 │  **}**  
 27 │   
 28 │  **public** **String** toString**()** **{**  
 29 │  **return** "Furniture{" **+**  
 30 │  "name='" **+** name **+** '\'' **+**  
 31 │  '}'**;**  
 32 │  **}**  
 33 │ **}**

#### src/furniture/Table.java

  1 │ **package** furniture**;**  
  2 │   
  3 │ **import** props**.\*;**  
  4 │ **public** **class** **Table** **extends** **Furniture** **{**  
  5 │  **public** **Table(String** name**){**  
  6 │  **super(**name**);**  
  7 │  **}**  
  8 │  **private** **Prop** topObject**;**  
  9 │  **private** **Prop** bottomObject**;**  
 10 │   
 11 │  **public** **void** setTopObject**(Prop** object**){**  
 12 │  **this.**topObject **=** object**;**  
 13 │  **System.**out**.**println**(**"На " **+** getName**()** **+** " стоит " **+** object**.**getName**());**  
 14 │  **}**  
 15 │  **public** **void** setBottomObject**(Prop** object**){**  
 16 │  **this.**bottomObject **=** object**;**  
 17 │  **System.**out**.**println**(**"Под " **+** getName**()** **+** " стоит " **+** object**.**getName**());**  
 18 │  **}**  
 19 │ **}**

#### src/food/Food.java

  1 │ **package** food**;**  
  2 │   
  3 │ **import** interfaces**.Nameable;**  
  4 │ **public** **abstract** **class** **Food** **implements** **Nameable** **{**  
  5 │  **protected** **String** name**;**  
  6 │  **public** **Food(String** name**)** **{**  
  7 │  **this.**name **=** name**;**  
  8 │  **}**  
  9 │  **public** **String** getName**()** **{**  
 10 │  **return** **this.**name**;**  
 11 │  **}**  
 12 │  **public** **int** hashCode**()** **{**  
 13 │  **return** name**.**hashCode**();**  
 14 │  **}**  
 15 │  **public** **boolean** equals**(Object** obj**)** **{**  
 16 │  **if** **(this** **==** obj**)** **return** true**;**  
 17 │  **if** **(**obj **==** null **||** getClass**()** **!=** obj**.**getClass**())** **return** false**;**  
 18 │  **Food** food **=** **(Food)** obj**;**  
 19 │  **return** name**.**equals**(**food**.**name**);**  
 20 │  **}**  
 21 │  **public** **String** toString**()** **{**  
 22 │  **return** "Food{" **+**  
 23 │  "name='" **+** name **+** '\'' **+**  
 24 │  '}'**;**  
 25 │  **}**  
 26 │ **}**  
 27 │

#### src/food/Pie.java

  1 │ **package** food**;**  
  2 │   
  3 │ **import** interfaces**.Nameable;**  
  4 │   
  5 │ **import** javax**.**swing**.\*;**  
  6 │   
  7 │ **public** **class** **Pie** **extends** **Food** **{**  
  8 │  **private** **String** filling**;**  
  9 │  **public** **Pie(String** name**,** **String** filling**){**  
 10 │  **super(**name**);**  
 11 │  **this.**filling **=** filling**;**  
 12 │  **}**  
 13 │   
 14 │  **public** **String** getFilling**()** **{**  
 15 │  **return** **this.**filling**;**  
 16 │  **}**  
 17 │ **}**

#### src/characters/Alice.java

 1 │ **package** characters**;**  
 2 │   
 3 │ **public** **class** **Alice** **extends** **Character** **{**  
 4 │  **public** **Alice(String** name**){**  
 5 │  **super(**name**);**  
 6 │  **}**  
 7 │ **}**

#### src/characters/Character.java

  1 │ **package** characters**;**  
  2 │   
  3 │ **import** characters**.**movement**.Direction;**  
  4 │ **import** interfaces**.Nameable;**  
  5 │ **import** interfaces**.Thinkable;**  
  6 │ **import** props**.Prop;**  
  7 │   
  8 │ **public** **abstract** **class** **Character** **implements** **Nameable,** **Thinkable** **{**  
  9 │  **protected** **String** name**;**  
 10 │  **private** **Prop** objectInHand**;**  
 11 │   
 12 │  **public** **Character(String** name**)** **{**  
 13 │  **this.**name **=** name**;**  
 14 │  **}**  
 15 │   
 16 │  **public** **String** getName**()** **{**  
 17 │  **return** **this.**name**;**  
 18 │  **}**  
 19 │   
 20 │  **public** **void** standBy**(Character** character**,** **Direction** direction**)** **{**  
 21 │  **System.**out**.**println**(**getName**()** **+** " находится " **+** direction**.**getDirection**()** **+** " " **+** character**.**getName**());**  
 22 │  **}**  
 23 │   
 24 │  **public** **void** salivateAt**(Prop** prop**)** **{**  
 25 │  **System.**out**.**println**(**"У " **+** getName**()** **+** " текут слюнки от " **+** prop**.**getName**());**  
 26 │  **}**  
 27 │   
 28 │  **public** **void** think**(String** thought**)** **{**  
 29 │  **System.**out**.**println**(**"\"" **+** thought **+** "\" - думает " **+** getName**());**  
 30 │  **}**  
 31 │   
 32 │  **public** **void** walkTo**(Direction** direction**)** **{**  
 33 │  **System.**out**.**println**(**getName**()** **+** " идет " **+** direction**.**getDirection**());**  
 34 │  **}**  
 35 │   
 36 │  @Override  
 37 │  **public** **boolean** equals**(Object** obj**)** **{**  
 38 │  **if** **(this** **==** obj**)** **return** true**;**  
 39 │  **if** **(**obj **==** null **||** getClass**()** **!=** obj**.**getClass**())** **return** false**;**  
 40 │  **Character** character **=** **(Character)** obj**;**  
 41 │  **return** name**.**equals**(**character**.**name**);**  
 42 │  **}**  
 43 │   
 44 │  @Override  
 45 │  **public** **int** hashCode**()** **{**  
 46 │  **return** name**.**hashCode**();**  
 47 │  **}**  
 48 │   
 49 │  @Override  
 50 │  **public** **String** toString**()** **{**  
 51 │  **return** "Character{" **+**  
 52 │  "name='" **+** name **+** '\'' **+**  
 53 │  '}'**;**  
 54 │  **}**  
 55 │   
 56 │  **public** **void** hold**(Prop** object**)** **{**  
 57 │  **this.**objectInHand **=** object**;**  
 58 │  **System.**out**.**println**(**getName**()** **+** " держит " **+** **this.**objectInHand**.**getName**());**  
 59 │  **}**  
 60 │ **}**  
 61 │   
 62 │

#### src/characters/King.java

  1 │ **package** characters**;**  
  2 │   
  3 │ **public** **class** **King** **extends** **Character** **{**  
  4 │  **public** **King(String** name**){**  
  5 │  **super(**name**);**  
  6 │  **}**  
  7 │  **public** **void** command**(String** command**){**  
  8 │  **System.**out**.**println**(**getName**()** **+** " приказывает: " **+** command**);**  
  9 │  **}**  
 10 │ **}**

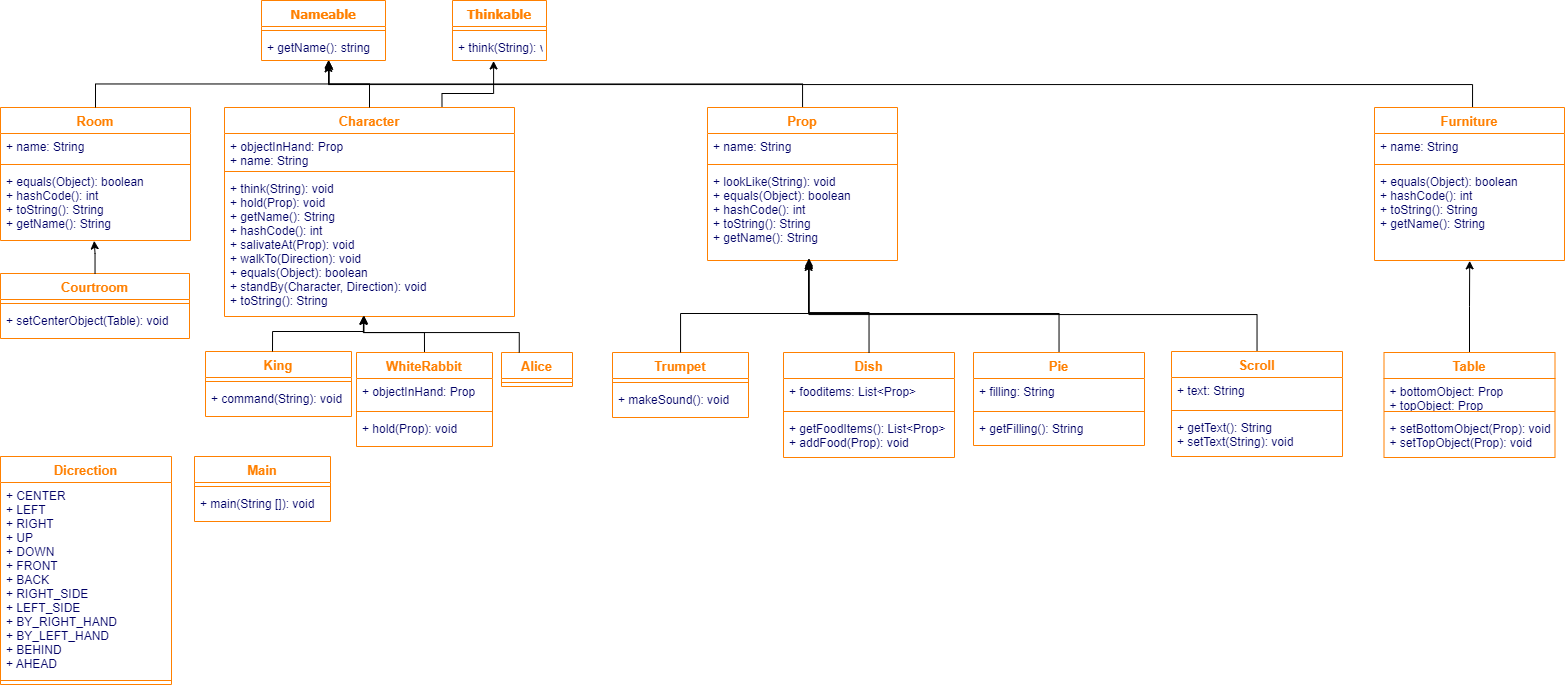
#### src/characters/WhiteRabbit.java

  1 │ **package** characters**;**  
  2 │   
  3 │ **import** props**.Prop;**  
  4 │   
  5 │ **public** **class** **WhiteRabbit** **extends** **Character** **{**  
  6 │  **private** **Prop** objectInHand**;**  
  7 │  **public** **WhiteRabbit(String** name**){**  
  8 │  **super(**name**);**  
  9 │  **}**  
 10 │  @Override  
 11 │  **public** **void** hold**(Prop** object**){**  
 12 │  **this.**objectInHand **=** object**;**  
 13 │  **System.**out**.**println**(**getName**()** **+** " держит в лапке " **+** **this.**objectInHand**.**getName**());**  
 14 │  **}**  
 15 │ **}**

#### src/characters/movement/Direction.java

  1 │ **package** characters**.**movement**;**  
  2 │   
  3 │ **public** **enum** **Direction** **{**  
  4 │  CENTER**(**"в центр"**),**  
  5 │  LEFT**(**"налево"**),**  
  6 │  RIGHT**(**"направо"**),**  
  7 │  UP**(**"вверх"**),**  
  8 │  DOWN**(**"вниз"**),**  
  9 │  FRONT**(**"вперед"**),**  
 10 │  BACK**(**"назад"**),**  
 11 │  RIGHT\_SIDE**(**"с правой стороны от"**),**  
 12 │  LEFT\_SIDE**(**"с левой стороны от"**),**  
 13 │  BY\_RIGHT\_HAND**(**"по правую руку от"**),**  
 14 │  BY\_LEFT\_HAND**(**"по левую руку от"**),**  
 15 │  BEHIND**(**"сзади"**),**  
 16 │  AHEAD**(**"спереди"**);**  
 17 │   
 18 │  **private** **String** direction**;**  
 19 │   
 20 │  **Direction(String** direction**)** **{**  
 21 │  **this.**direction **=** direction**;**  
 22 │  **}**  
 23 │   
 24 │  **public** **String** getDirection**()** **{**  
 25 │  **return** direction**;**  
 26 │  **}**  
 27 │   
 28 │ **}**

Диаграмма классов:



Вывод программы:

Белый Кролик находится по правую руку от Король

Белый Кролик держит в лапке Труба

Белый Кролик держит в лапке Пергаментный свиток

В центре Судебный зал стоит Стол

На Стол стоит Большое блюдо с пирожками

Большое блюдо с пирожками выглядит аппетитно

У Алиса текут слюнки от Большое блюдо с пирожками

"Хорошо бы, суд уже кончился и позвали к столу!" - думает Алиса

# **Выводы**:

В результате проделанной работы, я попрактиковался с применением принципов ООП, самостоятельно спроектировал объектную модель приложения, вручную изобразил диаграмму классов.