Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |
| Кафедра вычислительной техники |
| наименование кафедры |

|  |
| --- |
| **Отчет** |
| по лабораторной работе №4 по дисциплине «Разработка классов с использованием механизмов наследования, полиморфизма и инкапсуляции» |
| наименование темы  Вариант №29 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | АСУб-19-1 |  | C:\Users\Kinzel\AppData\Local\Microsoft\Windows\INetCache\Content.Word\1.png |  | Янхаев Д.В. |
|  |  | шифр |  | подпись |  | И.О. Фамилия |
| Проверил |  |  |  |  |  | Маланова Т.В. |
|  |  |  |  | подпись |  | И.О. Фамилия |
| Работа защищена с оценкой | | | |  | | |

Иркутск 2020 г.

**Содержание**

[1 Постановка задачи 3](#_Toc59591562)

[2 Описание структуры пользовательского меню 4](#_Toc59591563)

[3 Проектирование классов 5](#_Toc59591564)

[4 Таблица тестов 6](#_Toc59591565)

[5 Результаты тестирования 8](#_Toc59591566)

[6 Листинг исходного кода 18](#_Toc59591567)

[Список использованных источников 23](#_Toc59591568)

1 Постановка задачи

В соответствии с индивидуальным заданием описать иерархию классов, для каждого класса описать поля и соответствующие методы доступа к ним. В зависимости от задания некоторые из этих классов(как минимум один) являются абстрактными и служат для выделения общих данных и поведения для других классов. Абстрактный класс должен содержать как минимум один абстрактный метод, реализация которого у его наследников должна различаться. Помимо этого в общую часть задания входит разработка класса группирующего объекты описанных в соответствии с заданием классов. Для первого варианта индивидуального задания это может быть класс «кафедра». Для выполнения задания необходимо создать некоторое количество объектов, добавить их в группу используя предусмотренные методы класса-«группы» и для каждого из них вызвать унаследованный метод.

Вариант: Минерал, самоцвет, камень, сыпучая порода.

2 Описание структуры пользовательского меню

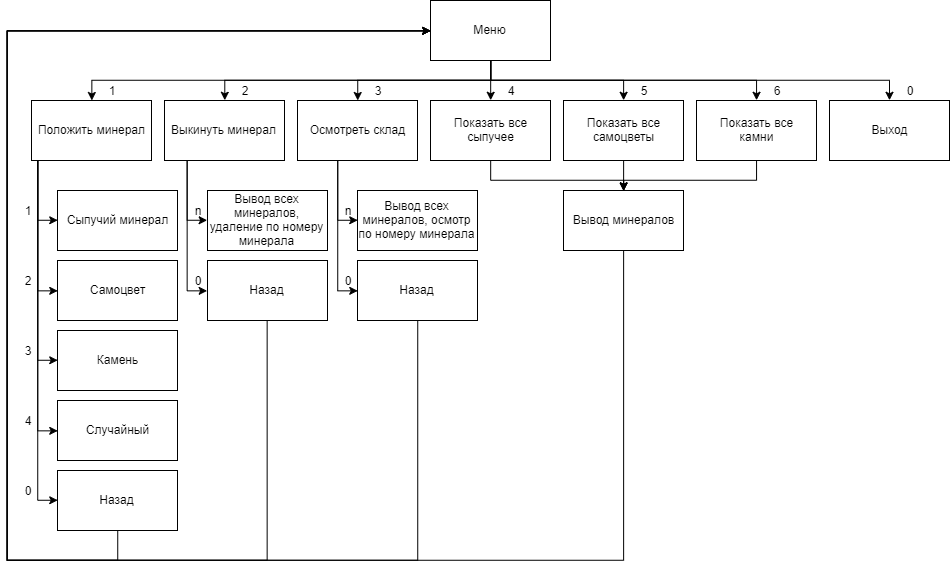


Рис.1 – Структура пользовательского меню

3 Проектирование классов

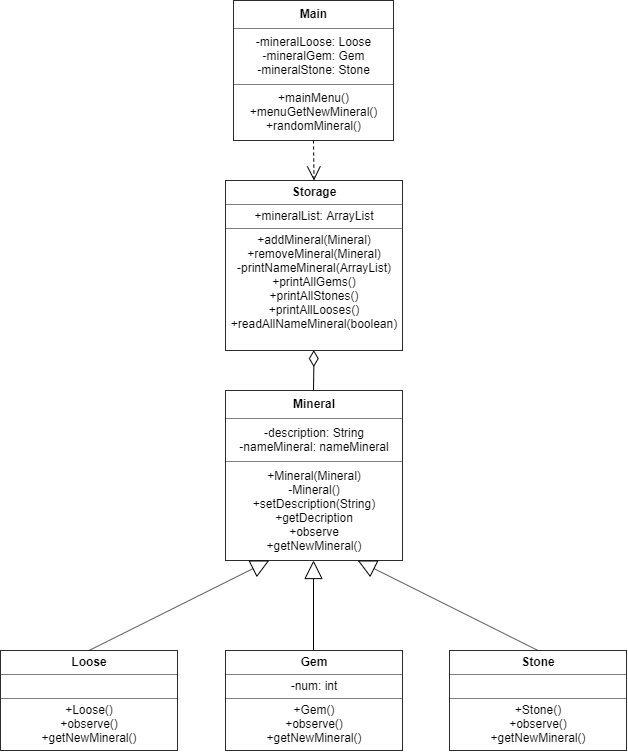


Рис. 2 – Проектирование классов

4 Таблица тестов

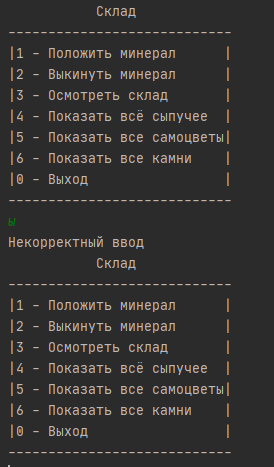
Таблица 1 – Тесты

|  |  |  |
| --- | --- | --- |
| № | Описание теста | Результат работы программы |
| 1 | Ввод в главном меню «Склад» недопустимого символа | Вывод сообщения об ошибке, повтор ввода |
| 2 | Ввод в главном меню «1» | Переход в меню «Положить минерал» |
| 3 | Ввод в меню «Положить минерал» «1» | Добавляет «Сыпучий минерал», возврат на «Склад» |
| 4 | Ввод в меню «Положить минерал» «2» | Добавляет «Самоцвет»,  возврат на «Склад» |
| 5 | Ввод в меню «Положить минерал» «3» | Добавляет «Камень»,  возврат на «Склад» |
| 6 | Ввод в меню «Положить минерал» «4» | Добавляет случайный минерал |
| 7 | Ввод в меню «Положить минерал» «0» | Возврат на «Склад» |
| 8 | Ввод в меню «Положить минерал» недопустимого символа | Вывод сообщения об ошибке, повтор ввода |
| 9 | Ввод в главном меню «2» | Вывод всех минералов, выбор удаления минерала по номеру |
| 10 | Ввод в меню «Выкинуть минерал» «1..n» | Удаление минерала |
| 11 | Ввод в меню «Выкинуть минерал» «0» | Возврат на «Склад» |
| 12 | Ввод в меню «Положить минерал» недопустимого символа | Вывод сообщения об ошибке, повтор ввода |
| 13 | Ввод в главном меню «3» | Вывод всех минералов, выбор осмотра минерала по номеру |
| 14 | Ввод в меню «Осмотреть минерал» «1..n» | Вывод всех минералов, выбор осмотра минерала по номеру |
| 15 | Ввод в меню «Осмотреть минерал» «0» | Возврат на «Склад» |

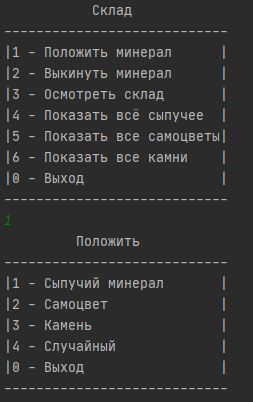
Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| № | Описание теста | Результат работы программы |
| 16 | Ввод в меню «Осмотреть минерал» недопустимого символа | Вывод сообщения об ошибке, повтор ввода |
| 17 | Ввод в главном меню «4» | Вывод всех сыпучих минералов, возврат на «Склад» |
| 18 | Ввод в главном меню «5» | Вывод всех самоцветов, возврат на «Склад» |
| 19 | Ввод в главном меню «6» | Вывод всех камней, возврат на «Склад» |
| 20 | Ввод в главном меню «0» | Выход |

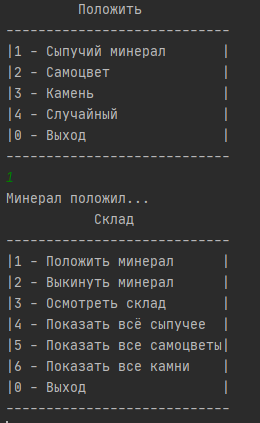
5 Результаты тестирования



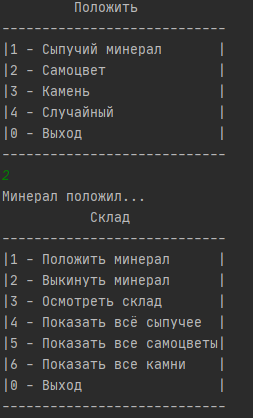
Тест 1



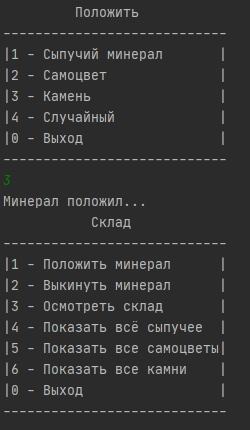
Тест2



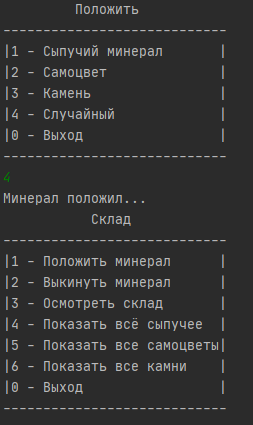
Тест 3



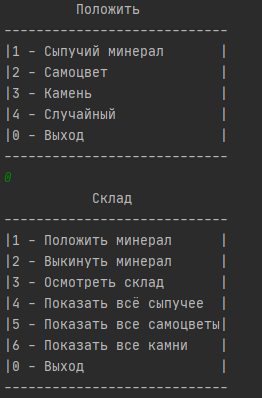
Тест 4



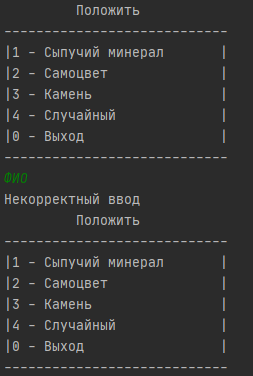
Тест 5



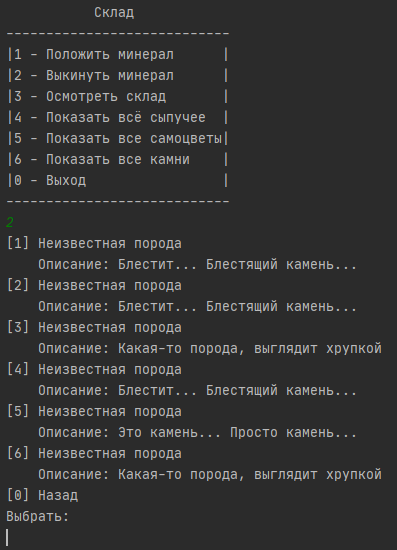
Тест 6



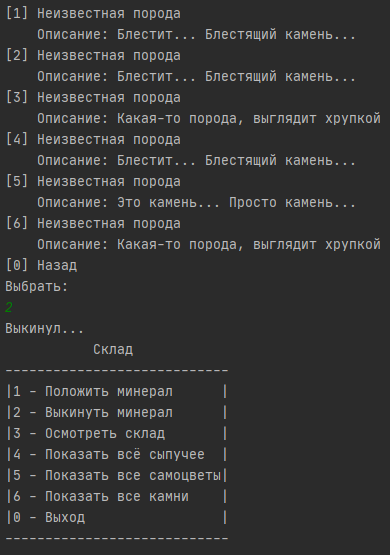
Тест 7



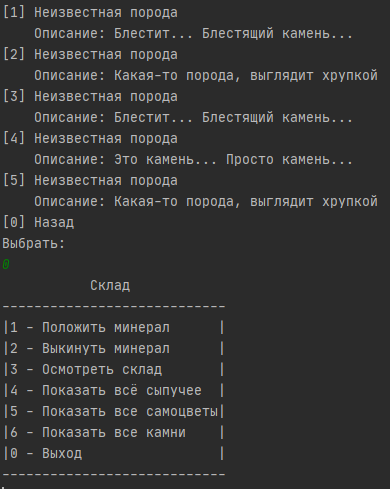
Тест 8



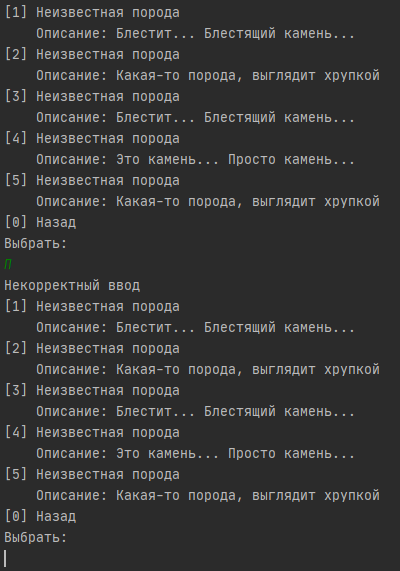
Тест 9



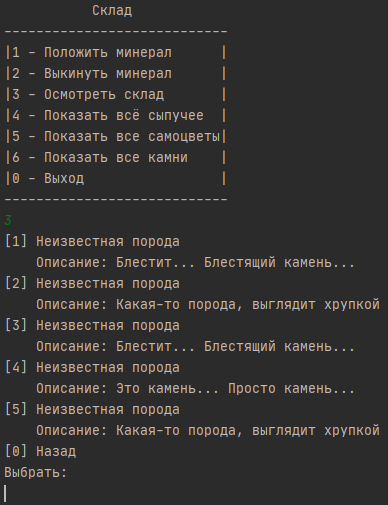
Тест 10



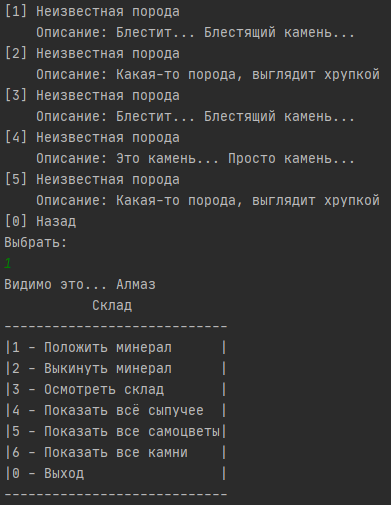
Тест 11



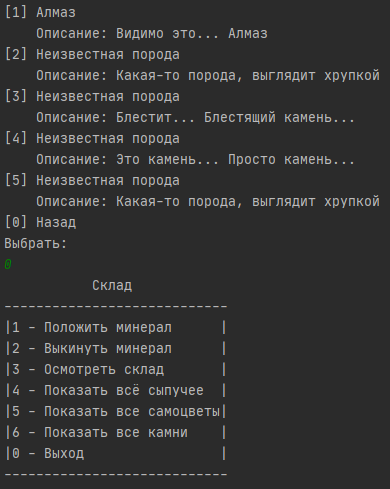
Тест 12



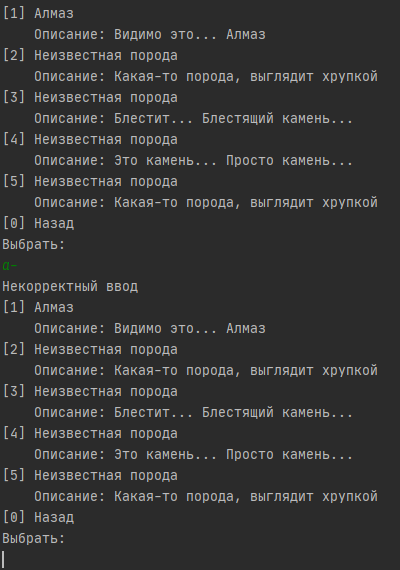
Тест 13



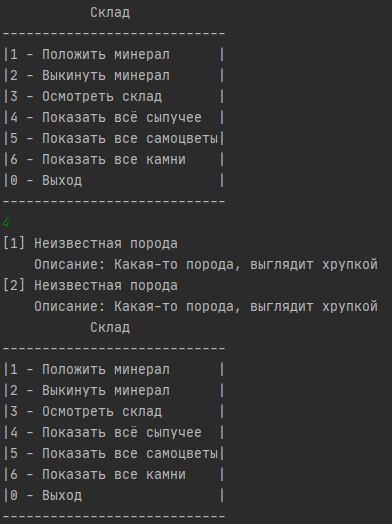
Тест 14



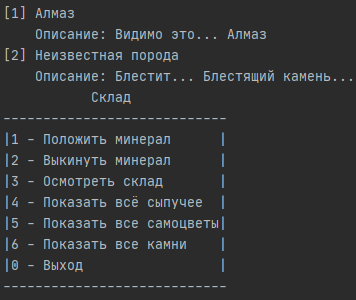
Тест 15



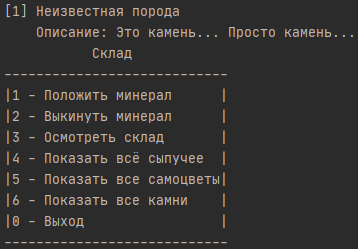
Тест 16



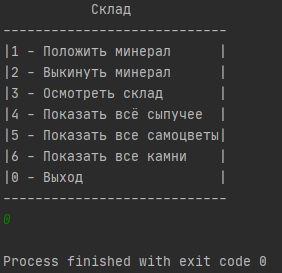
Тест 17



Тест 18



Тест 19



Тест 20

6 Листинг исходного кода

Класс Main

public class Main {  
 private static Scanner *in* = new Scanner(System.*in*);  
 private static Loose *mineralLoose* = new Loose();  
 private static Gem *mineralGem* = new Gem();  
 private static Stone *mineralStone* = new Stone();  
  
 public static void main(String[] args){  
 *mainMenu*();  
 }  
  
 public static void mainMenu() {  
 while (true){  
  
 int numSelectMainMenu = -1;  
  
 System.*out*.println(" Склад ");  
 System.*out*.println("----------------------------");  
 System.*out*.println("|1 - Положить минерал |");  
 System.*out*.println("|2 - Выкинуть минерал |");  
 System.*out*.println("|3 - Осмотреть склад |");  
 System.*out*.println("|4 - Показать всё сыпучее |");  
 System.*out*.println("|5 - Показать все самоцветы|");  
 System.*out*.println("|6 - Показать все камни |");  
 System.*out*.println("|0 - Выход |");  
 System.*out*.println("----------------------------");  
  
 try {  
 numSelectMainMenu = Integer.*parseInt*(*in*.nextLine());  
 if (numSelectMainMenu < 0 || numSelectMainMenu >= 7) throw new IllegalArgumentException();  
 }  
 catch (IllegalArgumentException e) {  
 System.*out*.println("Некорректный ввод");  
 }  
  
 switch (numSelectMainMenu){  
 case 1:  
 *menuGetNewMineral*();  
 break;  
 case 2:  
 Storage.*readAllNameMineral*(true);  
 break;  
 case 3:  
 Storage.*readAllNameMineral*(false);  
 break;  
 case 4:  
 Storage.*printAllLooses*();  
 break;  
 case 5:  
 Storage.*printAllGems*();  
 break;  
 case 6:  
 Storage.*printAllStones*();  
 break;  
 case 0:  
 break;  
 }  
 if(numSelectMainMenu==0) break;  
 }  
 }  
  
  
  
 public static void menuGetNewMineral(){  
 int numSelectMainMenu = -1;  
 while(true) {  
 System.*out*.println(" Положить ");  
 System.*out*.println("----------------------------");  
 System.*out*.println("|1 - Сыпучий минерал |");  
 System.*out*.println("|2 - Самоцвет |");  
 System.*out*.println("|3 - Камень |");  
 System.*out*.println("|4 - Случайный |");  
 System.*out*.println("|0 - Назад |");  
 System.*out*.println("----------------------------");  
  
 try {  
 numSelectMainMenu = Integer.*parseInt*(*in*.nextLine());  
 if (numSelectMainMenu < 0 || numSelectMainMenu > 4) throw new IllegalArgumentException();  
 } catch (IllegalArgumentException e) {  
 System.*out*.println("Некорректный ввод");  
 }  
  
 switch (numSelectMainMenu) {  
 case 1:  
 Storage.*addMineral*(*mineralLoose*.getNewMineral());  
 break;  
 case 2:  
 Storage.*addMineral*(*mineralGem*.getNewMineral());  
 break;  
 case 3:  
 Storage.*addMineral*(*mineralStone*.getNewMineral());  
 break;  
 case 4:  
 Storage.*addMineral*(*randomMineral*());  
 break;  
 case 0:  
 break;  
 }  
 if (numSelectMainMenu>=0 && numSelectMainMenu<=4){  
 if(numSelectMainMenu!=0)System.*out*.println("Минерал положил...");  
 break;  
 }  
 }  
 }  
  
 public static Mineral randomMineral(){  
 return switch ((int) (Math.*random*() \* 3)) {  
 case 0 -> *mineralLoose*.getNewMineral();  
 case 1 -> *mineralGem*.getNewMineral();  
 case 2 -> *mineralStone*.getNewMineral();  
 default -> null;  
 };  
 }  
}

Класс Storage

public class Storage {  
 private static Scanner *in* = new Scanner(System.*in*);  
 public static ArrayList<Mineral> *mineralList* = new ArrayList<>();  
  
 public static void addMineral(Mineral mineral){  
 *mineralList*.add(mineral);  
 }  
  
 public static void removeMineral(Mineral mineral){  
 *mineralList*.remove(mineral);  
 }  
  
 private static void printNameMineral(ArrayList<Mineral> mineralList){  
 for (int i = 0; i < mineralList.size(); i++) {  
 System.*out*.println("[" + (i + 1) + "]\t" + mineralList.get(i).nameMineral);  
 System.*out*.println("\tОписание: " + mineralList.get(i).description);  
 }  
 }  
  
 public static void printAllGems(){  
 ArrayList<Mineral> filterGems = new ArrayList<>();  
 for(Mineral mineral: *mineralList*){  
 if(mineral instanceof Gem) filterGems.add(mineral);  
 }  
 *printNameMineral*(filterGems);  
 }  
  
 public static void printAllStones(){  
 ArrayList<Mineral> filterGems = new ArrayList<>();  
 for(Mineral mineral: *mineralList*){  
 if(mineral instanceof Stone) filterGems.add(mineral);  
 }  
 *printNameMineral*(filterGems);  
 }  
  
 public static void printAllLooses(){  
 ArrayList<Mineral> filterGems = new ArrayList<>();  
 for(Mineral mineral: *mineralList*){  
 if(mineral instanceof Loose) filterGems.add(mineral);  
 }  
 *printNameMineral*(filterGems);  
 }  
  
 public static void readAllNameMineral(boolean flag) {  
 int numSelectMenu = -1;  
 while(true){  
 *printNameMineral*(*mineralList*);  
 System.*out*.println("[0]\tНазад");  
 System.*out*.println("Выбрать: ");  
 try {  
 numSelectMenu = Integer.*parseInt*(*in*.nextLine());  
 if (numSelectMenu < 0 || numSelectMenu > *mineralList*.size()) throw new IllegalArgumentException();  
 } catch (IllegalArgumentException e) {  
 System.*out*.println("Некорректный ввод");  
 }  
 if(numSelectMenu>=0 && numSelectMenu <= *mineralList*.size()) break;  
 }  
 if(numSelectMenu != 0){  
 if(flag){  
 *removeMineral*(*mineralList*.get(numSelectMenu-1));  
 System.*out*.println("Выкинул...");  
 }  
 else{  
 *mineralList*.get(numSelectMenu-1).observe();  
 }  
 }  
 }  
}

Абстрактный класс Mineral

public abstract class Mineral {  
 protected String description;  
 protected String nameMineral = "Неизвестная порода";  
  
  
 public Mineral(String description){  
 this.description = description;  
 }  
  
 protected Mineral() {  
 }  
  
 public void setDescription(String description){this.description = description;}  
 public String getDescription(){return description;}  
  
 public abstract void observe();  
 public abstract Mineral getNewMineral();  
}

Класс Loose

public class Loose extends Mineral{  
  
 public Loose(){ description = "Какая-то порода, выглядит хрупкой"; }  
  
  
 @Override  
 public void observe() {  
 System.*out*.println("Рассыпалось...");  
 Storage.*removeMineral*(this);  
 }  
  
 @Override  
 public Mineral getNewMineral() { return new Loose();  
 }  
}

Класс Gem

public class Gem extends Mineral{  
  
 private int num;  
  
 public Gem(){  
 num = (int) (Math.*random*()\*(6));  
 description = "Блестит... Блестящий камень...";  
 }  
   
 @Override  
 public void observe() {  
 switch (num){  
 case 0:  
 nameMineral = "Топаз";  
 description = "Видимо это... Топаз";  
 break;  
 case 1:  
 nameMineral = "Изумруд";  
 description = "Видимо это... Изумруд";  
 break;  
 case 2:  
 nameMineral = "Рубин";  
 description = "Видимо это... Рубин";  
 break;  
 case 3:  
 nameMineral = "Аметист";  
 description = "Видимо это... Аметист";  
 break;  
 case 4:  
 nameMineral = "Опал";  
 description = "Видимо это... Опал";  
 break;  
 case 5:  
 nameMineral = "Алмаз";  
 description = "Видимо это... Алмаз";  
 break;  
 default:  
 System.*out*.println("Тут ничего нет");  
 break;  
 }  
 System.*out*.println(description);  
 }  
  
 @Override  
 public Mineral getNewMineral() {  
 return new Gem();  
 }  
}

Класс Stone

public class Stone extends Mineral{  
  
 public Stone(){  
 description = "Это камень... Просто камень...";  
 }  
  
 @Override  
 public void observe() {  
 nameMineral = "Камень";  
 description = "Это всё ещё камень...";  
 System.*out*.println(description);  
 }  
  
 @Override  
 public Mineral getNewMineral() {  
 return new Stone();  
 }  
}

Список использованных источников

1. Методические указания по выполнению лабораторных работ / сост.: В.Л. Аршинский. – Иркутск : Изд-во ИРНИТУ, 2017. – 24 c.
2. API Specification for the Java 7 SE. [официальный сайт] URL: <http://docs.oracle.com/javase/7/docs/api/> (дата обращения:08.09.2020)
3. The Java Tutorials. SE [электронный ресурс], URL:<http://docs.oracle.com/javase/tutorial/index.html> (дата обращения:08.09.2020)