|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| *Федеральное государственное бюджетное образовательное учреждение высшего образования* ***«МИРЭА – Российский технологический университет»***  **РТУ МИРЭА** |

**Институт информационных технологий (ИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**Дисциплина «Программирование на языке Джава»**

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №4**

Выполнил студент группы ИВБО-07-19 Гудаев И.И.

Принял Степанин П.С.

Практические работы выполнены «\_\_\_»\_\_\_\_\_\_\_2020г.

«\_\_\_\_\_\_\_\_\_\_\_\_\_\_» «\_\_\_»\_\_\_\_\_\_\_2020г.

Отметка о выполнении

**Москва – 2020 г.**

## **Задание**

1.Прочитать UML диаграмму: на диаграмме представлены Абстрактный суперкласс Shape и его подклассы Circle, Rectangle and Square.  
2.Перепишите суперкласс Shape и его подклассы так как это представлено на UML диаграмме Circle, Rectangle and Square

## **Ход Работы**

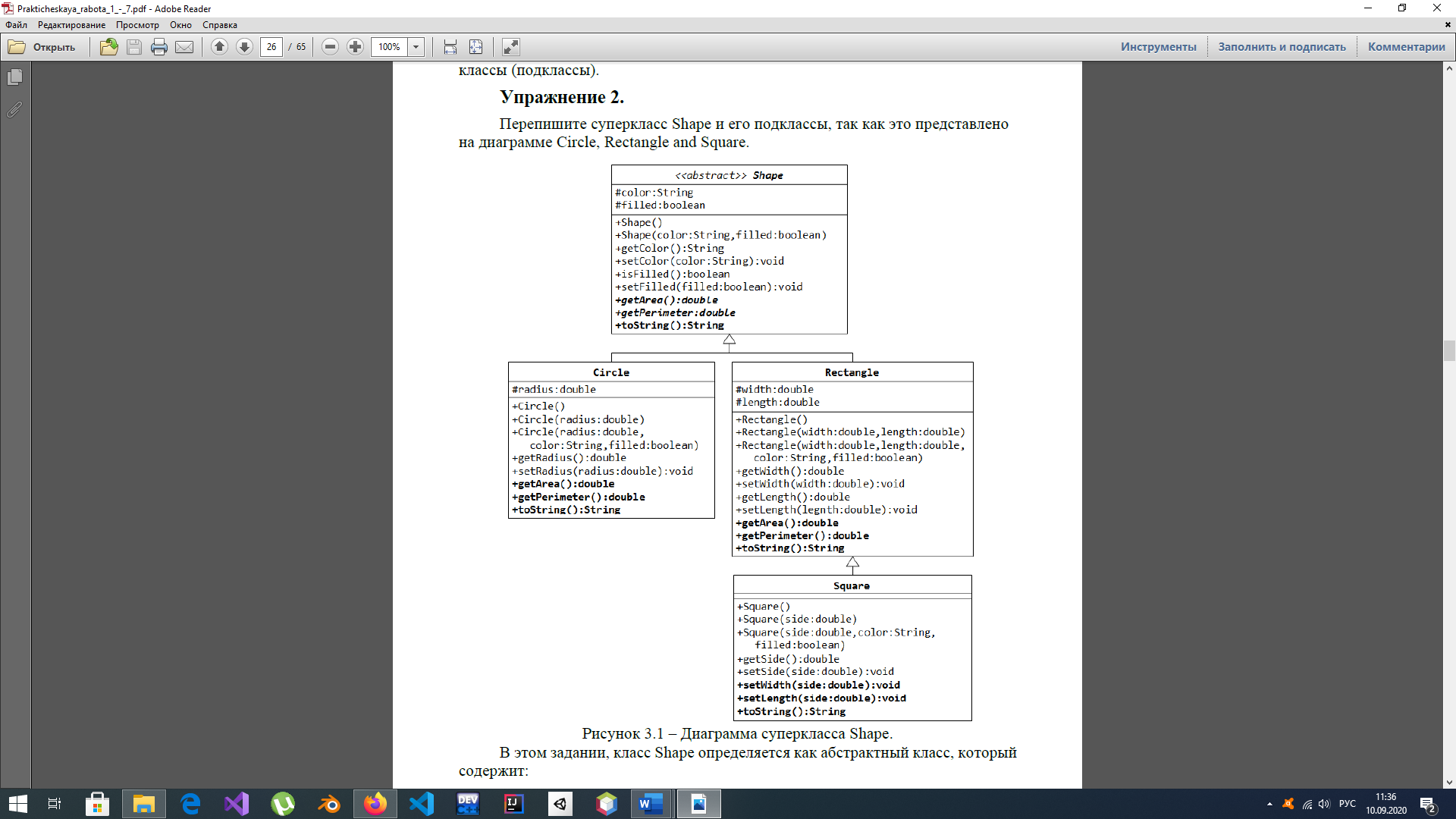


Рисунок 1. UML диаграмма

По UML диаграмме видно, что требуется создать базовый абстрактный класс Shape с полями, указанными в 1 колонке, и методами из 2 колонки. От него должны наследоваться два класса – Circle и Rectangle, которые должны так же обладать собственными методами и полями и реализовать несколько абстрактных методов суперкласса. Затем от класса Rectangle наследуется класс Square. Программная реализация представлена ниже на листингах 1-5

В ходе выполнения работы были получены следующие исходные коды:

**package** com.company;

**public** **abstract** **class** Shape {

**protected** **String** color;

**protected** boolean filled;

Shape(**String** color, boolean filled)

{

**this**.color = color;

**this**.filled = filled;

}

**public** **String** getColor() {

**return** color;

}

**public** void setColor(**String** color) {

**this**.color = color;

}

**public** boolean isFilled() {

**return** filled;

}

**public** void setFilled(boolean filled) {

**this**.filled = filled;

}

**public** **abstract** double getArea();

**public** **abstract** double getPerimeter();

**@Override**

**public** **String** toString() {

**return** "Shape{" +

"color='" + color + '\'' +

", filled=" + filled +

'}';

}

}

Листинг 1. Задание 2 (класс Shape)

**package** com.company;

**public** **class** Circle **extends** Shape {

**protected** double radius;

Circle(double radius)

{

**super**("Black", **false**);

}

Circle(double radius, **String** color, boolean filled)

{

**super**(color, filled);

**this**.radius = radius;

}

**public** double getRadius() {

**return** radius;

}

**public** void setRadius(double radius) {

**this**.radius = radius;

}

Листинг 2.1. Задание 2 (класс Circle)

**@Override**

**public** double getArea() {

**return** 3.14 \* radius \* radius;

}

**@Override**

**public** double getPerimeter() {

**return** 2 \* 3.14 \* radius;

}

**@Override**

**public** **String** toString() {

**return** "Circle{" +

"radius=" + radius +

", color='" + color + '\'' +

", filled=" + filled +

'}';

}

}

Листинг 2.2. Задание 2 (класс Circle)

**package** com.company;

**public** **class** Rectangle **extends** Shape{

**protected** double width, length;

Rectangle(double width, double length, **String** color, boolean filled)

{

**super**(color, filled);

**this**.length = length;

**this**.width = width;

}

Rectangle(double width, double length)

{

**super**("Black", **false**);

**this**.length = length;

**this**.width = width;

}

**public** double getWidth() {

**return** width;

}

**public** void setWidth(double width) {

**this**.width = width;

}

**public** double getLength() {

**return** length;

}

**public** void setLength(double length) {

**this**.length = length;

}

**@Override**

**public** double getArea() {

**return** **this**.width \* **this**.length;

}

**@Override**

**public** double getPerimeter() {

**return** **this**.width + **this**.length;

}

Листинг 3.1. Задание 2 (класс Rectangle)

**@Override**

**public** **String** toString() {

**return** "Rectangle{" +

"width=" + width +

", length=" + length +

", color='" + color + '\'' +

", filled=" + filled +

'}';

}

}

Листинг 3.2. Задание 2 (класс Rectangle)

**package** com.company;

**public** **class** Square **extends** Rectangle {

Square(double side)

{

**super**(side, side, "Black", **false**);

}

Square(double side, **String** color, boolean filled)

{

**super**(side, side, color, filled);

}

**public** double getSide()

{

**return** length;

}

**public** void setSide(double side)

{

**this**.length = side;

**this**.width = side;

}

**@Override**

**public** **String** toString() {

**return** **super**.toString();

}

}

Листинг 4. Задание 2 (класс Square)

**package** com.company;

**public** **class** Main {

**public** **static** void main(**String**[] args) {

// write your code here

Circle cr = **new** Circle(2);

Rectangle rc = **new** Rectangle(1, 2, "Red", **true**);

Square sq = **new** Square(3);

**System**.out.println(cr.toString());

**System**.out.println(sq.toString());

**System**.out.println(rc.toString());

}

}

Листинг 5. Задание 2 (класс Main)

## **Вывод**

Мы научились читать UML диаграммы: узнали, как обозначаются классы, где и как указываются поля и методы классов на диаграмме, как отображается наследственная связь между дочерним и родительским классами, а так же написали программу по заданной UML диаграмме