Отчет по практической работе №5 на языке программирования **Kotlin**

Состав отчета:

-Пояснительная записка
-Документация
-Ссылка на github

Пояснительная записка:

Во время выполнения данной практической возникали трудности, но используя имеющийся опыт и интернет, они были решены.

Документация

Данная документация содержит описания классов и их функционала, созданных в рамках практической работы №5. Ниже приведено описание каждого из созданных классов, включая их конструкторы, св-ва, методы.

Класс **Point**

Класс "Point" описывает точку в двумерном пространстве.

Свойства:
- **x**: Координата x точки.
- **y**: Координата y точки.

Конструкторы:
1. `Point(_x: Double, _y: Double)`: Создает объект точки с указанными координатами [x] и [y].
2. `Point()`: Создает точку с координатами (0.0, 0.0).
3. `Point(_x: Double)`: Создает точку с заданной координатой [x] и координатой y по умолчанию (0.0).

Методы:
- `info()`: Переопределенный метод для вывода информации о точке. Выводит координаты точки в формате "(x, y)".

Пример использования:
```
val point1 = Point(2.0, 3.0)
val point2 = Point(1.0)
point1.info() // Выведет "Точка с координатами (2.0, 3.0)"
point2.info() // Выведет "Точка с координатами (1.0, 0.0)"
```

Класс **ColoredPoint**

Класс "ColoredPoint" представляет собой точку в двумерном пространстве с добавленным цветом.

Свойства:
- **color**: Цвет точки, представленный объектом типа "Color".
- **point**: Точка с координатами в двумерном пространстве.

Конструкторы:
1. `ColoredPoint()`: Создает объект "ColoredPoint" без указания координат и цвета. Координаты и цвет будут установлены в значения по умолчанию (0.0, 0.0, null).
2. `ColoredPoint(_x: Double, _y: Double, _color: Color)`: Создает объект "ColoredPoint" с указанными координатами [x] и [y] и цветом [_color].
3. `ColoredPoint(_x: Double, _color: Color)`: Создает объект "ColoredPoint" с указанной координатой [x] и цветом [_color]. Координата [y] устанавливается в значение по умолчанию (0.0).

Методы:
- `info()`: Переопределенный метод для вывода информации о точке вместе с её цветом. Выводит координаты точки в формате "(x, y)", а также информацию о цвете.

Пример использования:
val redPoint = ColoredPoint(2.0, 3.0, Color.RED)
val bluePoint = ColoredPoint(1.0, Color.BLUE)
redPoint.info() // Выведет "Точка с координатами (2.0, 3.0), имеет цвет RED"
bluePoint.info() // Выведет "Точка с координатами (1.0, 0.0), имеет цвет BLUE"

Класс **Line**

Класс "Line" представляет собой линию в двумерном пространстве, определенную двумя точками.

Свойства:
- **point1**: Первая точка линии, представленная объектом типа "Point".
- **point2**: Вторая точка линии, представленная объектом типа "Point".

Конструкторы:
1. `Line()`: Создает объект "Line" без указания точек. Обе точки будут установлены в значения по умолчанию (0.0, 0.0).
2. `Line(_point1: Point, _point2: Point)`: Создает объект "Line", определенную двумя указанными точками [_point1] и [_point2].

Методы:

- `info()`: Переопределенный метод для вывода информации о линии. Выводит координаты двух точек, определяющих линию, в формате "((x1, y1), (x2, y2))".

Пример использования:
val pointA = Point(1.0, 2.0)
val pointB = Point(3.0, 4.0)
val lineAB = Line(pointA, pointB)
lineAB.info() // Выведет "Прямая с координатами ((1.0, 2.0), (3.0, 4.0))"

Класс **ColoredLine**

Класс "ColoredLine" представляет собой линию в двумерном пространстве с добавленным цветом, определенную двумя точками.

Свойства:
- **line**: Линия, определенная двумя точками, представленная объектом типа "Line".
- **color**: Цвет линии, представленный объектом типа "Color".

Конструкторы:
1. `ColoredLine()`: Создает объект "ColoredLine" без указания точек и цвета. Обе точки и цвет будут установлены в значения по умолчанию (0.0, 0.0, null).
2. `ColoredLine(_point1: Point, _point2: Point, _color: Color)`: Создает объект "ColoredLine", определенную двумя указанными точками [_point1] и [_point2], а также указанным цветом [_color].

Методы:
- `info()`: Переопределенный метод для вывода информации о линии вместе с её цветом. Выводит координаты двух точек, определяющих линию, в формате "((x1, y1), (x2, y2))", а также информацию о цвете.

Пример использования:
val pointA = Point(1.0, 2.0)
val pointB = Point(3.0, 4.0)
val coloredLineAB = ColoredLine(pointA, pointB, Color.RED)
coloredLineAB.info() // Выведет "Прямая с координатами ((1.0, 2.0), (3.0, 4.0)), имеет цвет RED"

Класс **Polygon**

Класс "Polygon" представляет многоугольник в двумерном пространстве, определенный набором точек. Этот класс расширяет класс "Line" и добавляет функциональность для перемещения многоугольника по осям ox и oy.

Свойства:
- **points**: Массив точек, определяющих многоугольник.

Конструктор:
- `Polygon(vararg _points: Point)`: Создает объект "Polygon" с заданным набором точек [_points].

Методы:
- `moveOx(dx: Double)`: Перемещает многоугольник на указанное расстояние по оси ox.
- `moveOy(dy: Double)`: Перемещает многоугольник на указанное расстояние по оси oy.
- `move(dx: Double, dy: Double)`: Перемещает многоугольник на указанные расстояния по осям ox и oy.

Пример использования:
```
val point1 = Point(1.0, 2.0)
val point2 = Point(3.0, 4.0)
val point3 = Point(5.0, 6.0)
val polygon = Polygon(point1, point2, point3)

polygon.info() // Выведет "Многоугольник с координатами точек: (1.0,2.0) (3.0,4.0) (5.0,6.0)"

polygon.moveOx(2.0) // Перемещает многоугольник на 2.0 по оси ox
polygon.info() // Выведет обновленные координаты точек многоугольника
```

Ссылка на github:

Ссылка: https://github.com/Skr0ls/Practice5