```kotlin
package math.differential

import javafx.geometry.Point2D
import org.mariuszgromada.math.mxparser.Expression

/**
 * Содержит методы для решения обыкновенных дифференциальных уравнений.
 * @property EULER решение ОДУ методом Эйлера.
 * @author Артемий Кульбако.
 * @version 1.1
 */
internal enum class OrdinaryDifferentialSolver {

    EULER {
        override fun solve(f: Expression, startPoint: Point2D, h: Double, interval: ClosedFloatingPointRange<Double>) =
            generateSequence(startPoint) {
                Point2D(
                    it.x + h,
                    it.y + h * f.apply {
                        this.setArgumentValue("x", it.x)
                        this.setArgumentValue("y", it.y) }.calculate().let {
                            y -> if (y.isNaN()) throw Exception("Невозможно решить введённое ОДУ") else y }
                ) }.takeWhile { it.x <= interval.endInclusive }.toList()

        override fun toString() = "Метод Эйлера"
    };

    /**
     * Решает обыкновенное дифференциальное уравнение. [f] - функция, для которой нужно вернуть
     * @return точки ОДУ на промежутке [interval] с шагом [h], где [startPoint] - x0 и y0.
     */
    abstract fun solve(f: Expression, startPoint: Point2D, h: Double, interval: ClosedFloatingPointRange<Double>): List<Point2D>
}
```