Лабораторная n6

Задание 1

Добавьте в класс Functions метод, возвращающий значение интеграла функции, вычисленное с помощью численного метода.

```
public static double integrate(Function function, double leftLimit, double rightLimit, double step) {  3 usages
    // Проверка входных параметров
    if (step <= 0) {
        throw new IllegalArgumentException("Шаг интегрирования должен быть положительным: " + step);
    }

    if (Double.isNaN(leftLimit) || Double.isNaN(rightLimit)) {
        throw new IllegalArgumentException("Границы интегрирования не могут быть NaN");
    }

    if (leftLimit >= rightLimit) {
        throw new IllegalArgumentException(
                String.format("Левая граница (%f) должна быть меньше правой (%f)", leftLimit, rightLimit)
        );
    }

    // Проверка области определения
    if (leftLimit < function.getLeftDomainBorder() || rightLimit > function.getRightDomainBorder()) {
        throw new IllegalArgumentException(
                String.format("Интервал интегрирования [%f, %f] выходит за границы области определения [%f, %f]",
                        leftLimit, rightLimit,
                        function.getLeftDomainBorder(), function.getRightDomainBorder())
        );
    }

    double integral = 0.0;
    double currentX = leftLimit;
    double nextX;
```

Задание 2

Создайте пакет threads, в котором будут размещены классы, связанные с потоками.

```java
package threads;

import functions.Function;

public class Task {  7 usages
    private Function function;  3 usages
    private double leftBound;  4 usages
    private double rightBound;  4 usages
    private double step;  4 usages
    private int taskCount;  3 usages
    private int generatedCount;  5 usages
    private int processedCount;  5 usages

    public Task() {  1 usage
        this.generatedCount = 0;
        this.processedCount = 0;
    }

    public Task(Function function, double leftBound, double rightBound, double step) {
        this.function = function;
        this.leftBound = leftBound;
        this.rightBound = rightBound;
        this.step = step;
        this.generatedCount = 0;
        this.processedCount = 0;
    }

    // Геттеры и сеттеры
    public Function getFunction() {
        return function;
    }

    public void setFunction(Function function) {
        this.function = function;
    }

    public double getLeftBound() {  1 usage
        return leftBound;
    }

    public void setLeftBound(double leftBound) {  1 usage
```

Задание 3.

В пакете threads создайте два следующих класса. При их реализации воспользуйтесь фрагментами последовательной версии программы из предыдущего задания.

```java
package threads;

import functions.basic.Log;
import java.util.Random;

public class SimpleGenerator implements Runnable {  2 usages
    private final Task task;  15 usages
    private final Random random;  5 usages

    public SimpleGenerator(Task task) {  1 usage
        this.task = task;
        this.random = new Random();
    }

    @Override
    public void run() {
        try {
            int taskCount = task.getTaskCount();

            for (int i = 0; i < taskCount; i++) {
                synchronized (task) {
                    // Генерация параметров
                    double base = 1 + random.nextDouble() * 9;
                    if (Math.abs(base - 1.0) < 1e-10) base = 1.1;

                    double leftBound = random.nextDouble() * 99.9 + 0.1;
                    double rightBound = 100 + random.nextDouble() * 100;

                    double step = random.nextDouble();
                    if (step < 1e-10) step = 0.01;

                    // Установка параметров
                    task.setFunction(new Log(base));
                    task.setLeftBound(leftBound);
                    task.setRightBound(rightBound);
                    task.setStep(step);
```

Задание 4.

```
=== ПРОГРАММА С СЕМАФОРАМИ ===

=== ВЕРСИЯ С СЕМАФОРАМИ ===
Потоки будут прерваны через 50 мс

Количество заданий: 100
Запуск потоков...

Генератор [25]: Задание 1/100 - Source 63,7406 152,7839 0,648289
Интегратор [26]: Результат 1/100 - Result 63,7406 152,7839 0,648289 239,7332813566
Генератор [25]: Задание 2/100 - Source 28,7570 151,5813 0,757239

Основной поток: Прерываю рабочие потоки через 50 мс...
Интегратор [26]: Прерван - sleep interrupted
Генератор [25]: Прерван - sleep interrupted

=== РЕЗУЛЬТАТЫ ===
Сгенерировано: 2
Обработано: 0

=== ПРОГРАММА ЗАВЕРШЕНА ===

Process finished with exit code 0
```