Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет имени академика
С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

Отчет по лабораторной работе №6

Дисциплина: «ООП»

Тема «Разработать приложение, формирующее в одном потоке вычислений набор заданий для интегрирования, а во втором потоке – вычисляющее значения интегралов функций.»

Выполнил: Зорин Дмитрий Сергеевич

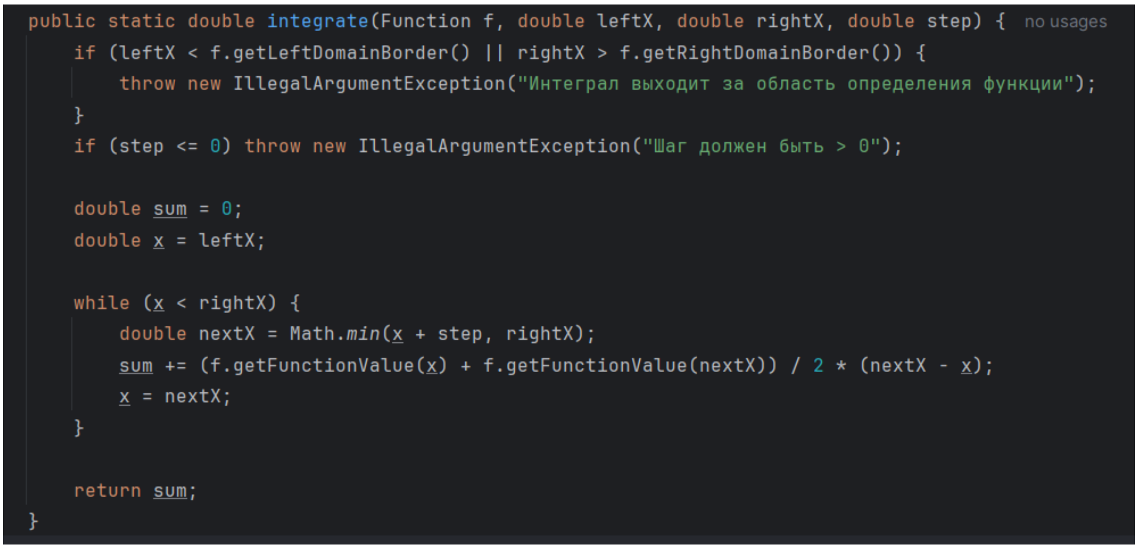
Группа: 6201-120303D

Самара, 2025

Задание на лабораторную работу

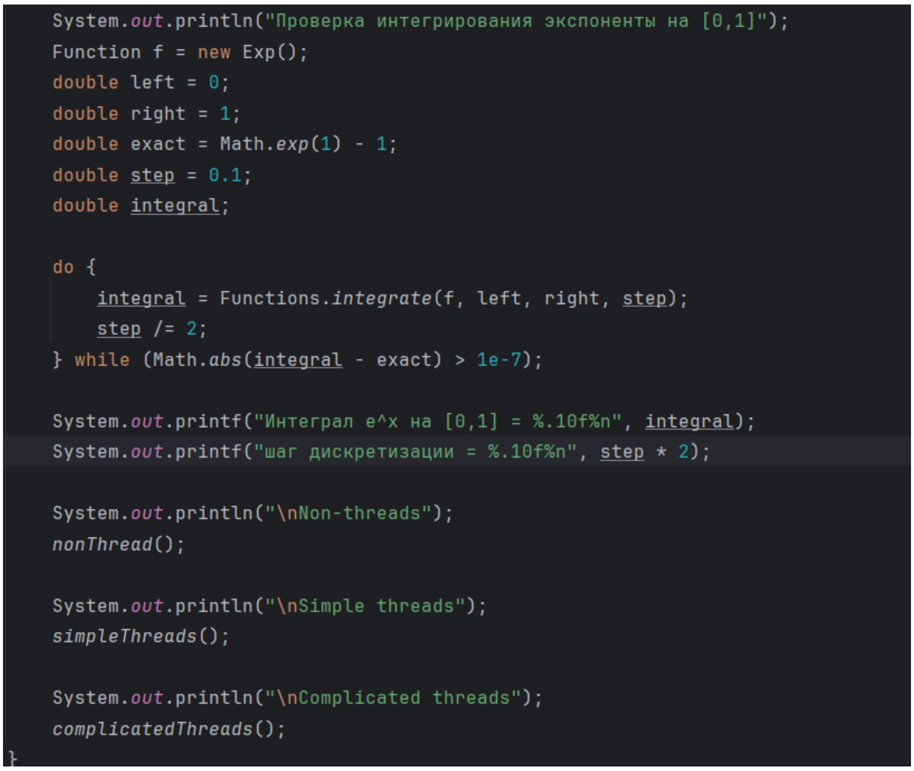
Задание 1

В класс Functions добавил метод для вычисления интеграла функции методом трапеций



```
public static double integrate(Function f, double leftX, double rightX, double step) {  no usages
    if (leftX < f.getLeftDomainBorder() || rightX > f.getRightDomainBorder()) {
        throw new IllegalArgumentException("Интеграл выходит за область определения функции");
    }
    if (step <= 0) throw new IllegalArgumentException("Шаг должен быть > 0");

    double sum = 0;
    double x = leftX;

    while (x < rightX) {
        double nextX = Math.min(x + step, rightX);
        sum += (f.getFunctionValue(x) + f.getFunctionValue(nextX)) / 2 * (nextX - x);
        x = nextX;
    }

    return sum;
}
```

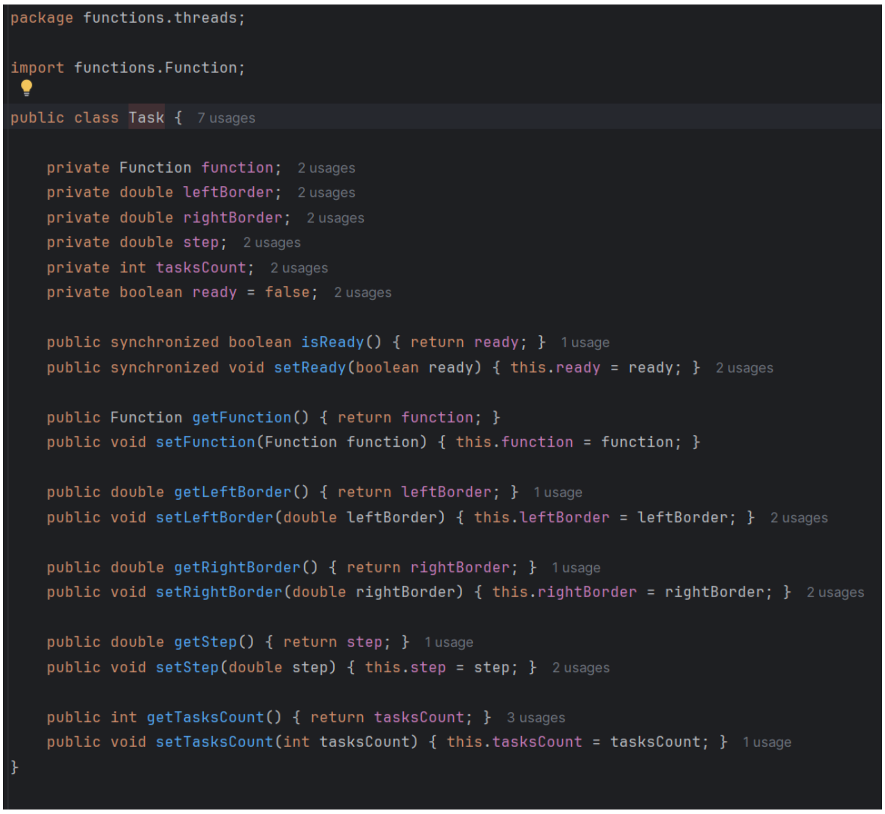
Метод делит интервал и суммирует площади трапеций
В методе Main добавил проверку работы метода интегрирования:



```
    System.out.println("Проверка интегрирования экспоненты на [0,1]");
    Function f = new Exp();
    double left = 0;
    double right = 1;
    double exact = Math.exp(1) - 1;
    double step = 0.1;
    double integral;

    do {
        integral = Functions.integrate(f, left, right, step);
        step /= 2;
    } while (Math.abs(integral - exact) > 1e-7);

    System.out.printf("Интеграл e^x на [0,1] = %.10f%n", integral);
    System.out.printf("шаг дискретизации = %.10f%n", step * 2);

    System.out.println("\nNon-threads");
    nonThread();

    System.out.println("\nSimple threads");
    simpleThreads();

    System.out.println("\nComplicated threads");
    complicatedThreads();
}
```

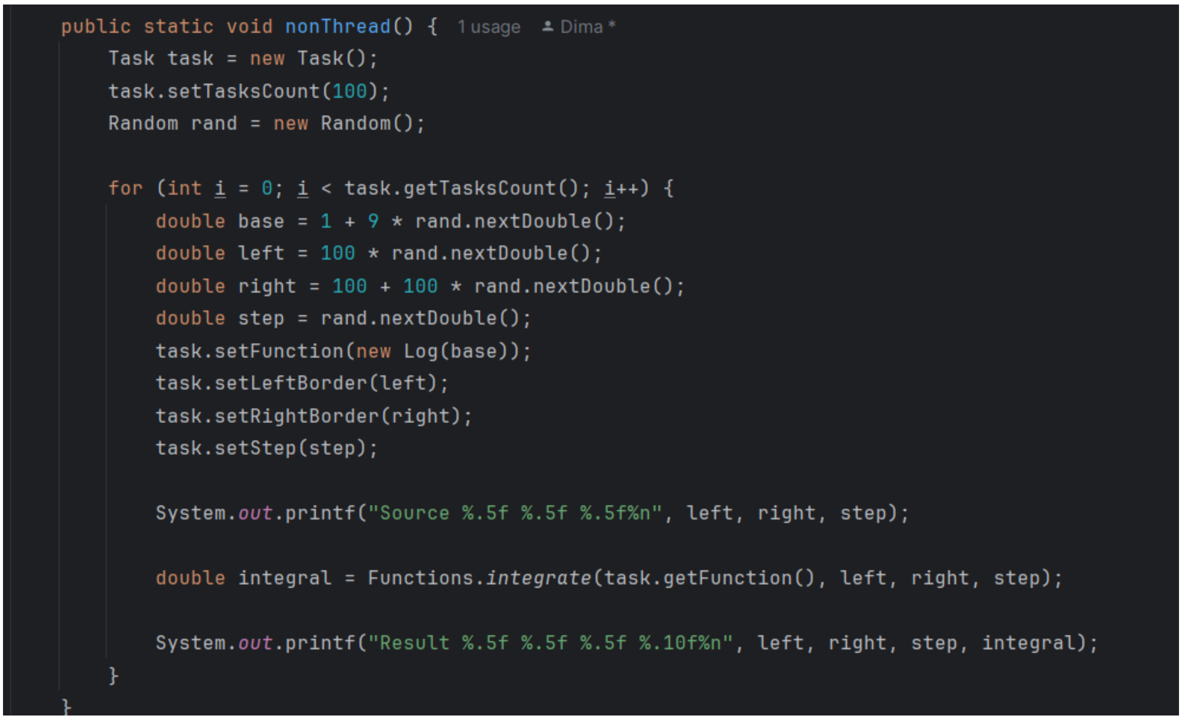
Задание 2

Во 2 задании я создал пакет Threads и описал класс Task:



```
package functions.threads;

import functions.Function;

public class Task {  7 usages

    private Function function;  2 usages
    private double leftBorder;  2 usages
    private double rightBorder;  2 usages
    private double step;  2 usages
    private int tasksCount;  2 usages
    private boolean ready = false;  2 usages

    public synchronized boolean isReady() { return ready; }  1 usage
    public synchronized void setReady(boolean ready) { this.ready = ready; }  2 usages

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

    public double getLeftBorder() { return leftBorder; }  1 usage
    public void setLeftBorder(double leftBorder) { this.leftBorder = leftBorder; }  2 usages

    public double getRightBorder() { return rightBorder; }  1 usage
    public void setRightBorder(double rightBorder) { this.rightBorder = rightBorder; }  2 usages

    public double getStep() { return step; }  1 usage
    public void setStep(double step) { this.step = step; }  2 usages

    public int getTasksCount() { return tasksCount; }  3 usages
    public void setTasksCount(int tasksCount) { this.tasksCount = tasksCount; }  1 usage
}
```

А также метод nonThread():



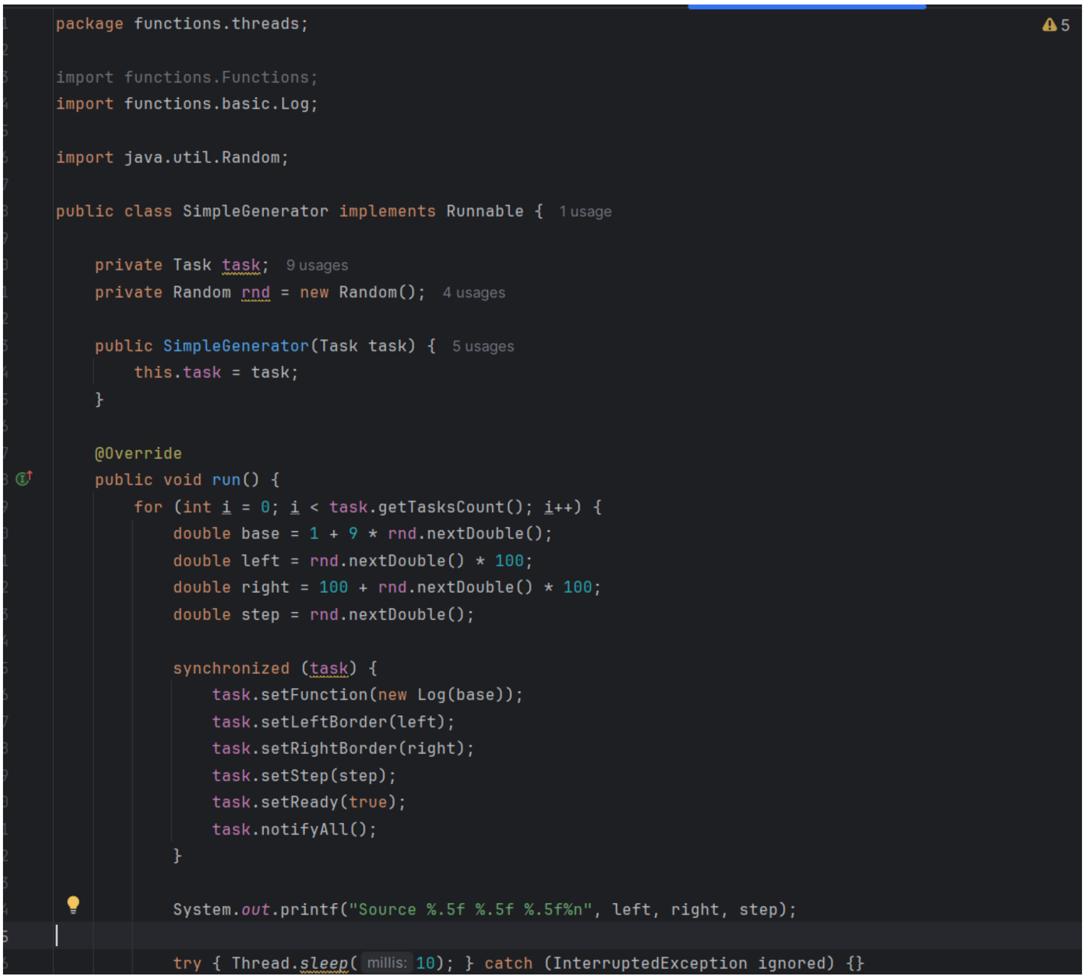
```java
public static void nonThread() {
    Task task = new Task();
    task.setTasksCount(100);
    Random rand = new Random();

    for (int i = 0; i < task.getTasksCount(); i++) {
        double base = 1 + 9 * rand.nextDouble();
        double left = 100 * rand.nextDouble();
        double right = 100 + 100 * rand.nextDouble();
        double step = rand.nextDouble();
        task.setFunction(new Log(base));
        task.setLeftBorder(left);
        task.setRightBorder(right);
        task.setStep(step);

        System.out.printf("Source %.5f %.5f %.5f%n", left, right, step);

        double integral = Functions.integrate(task.getFunction(), left, right, step);

        System.out.printf("Result %.5f %.5f %.5f %.10f%n", left, right, step, integral);
    }
}
```

Метод nonThread() показывает как последовательная обработка через методы объекта Taskс, а класс Functions позволяет интегрировать функции без многопоточности

Задание 3

Необходимо реализовать два потока: генератор задач и интегратор

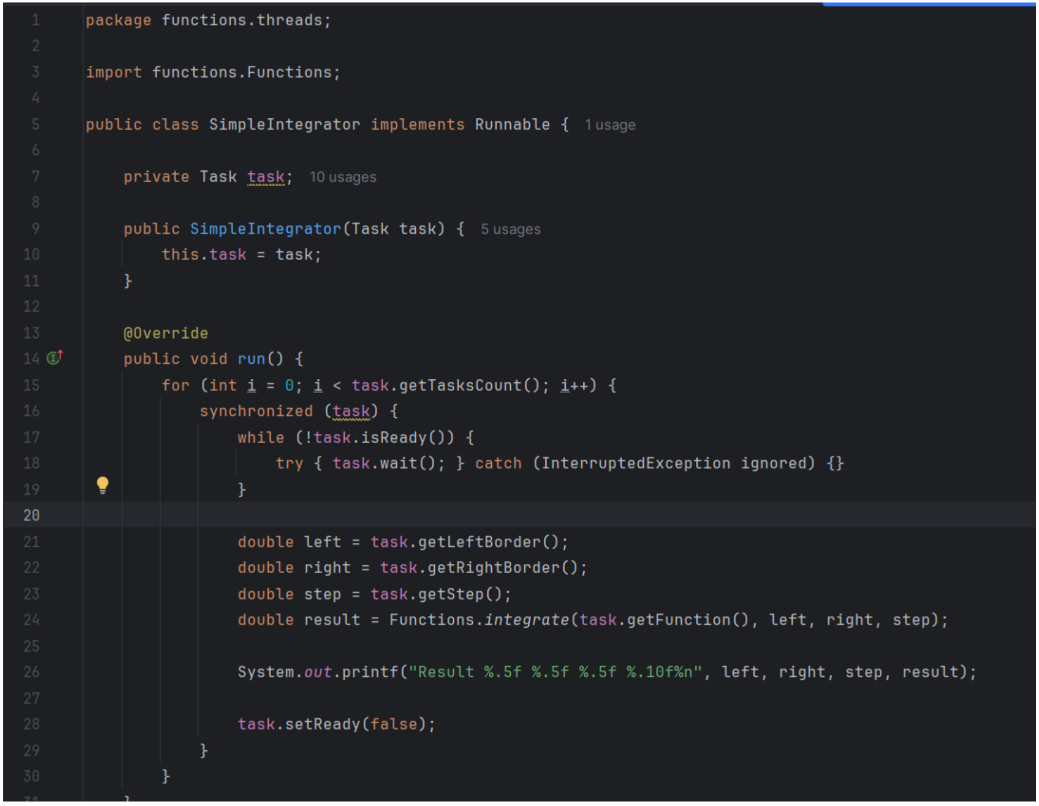
Класс SimpleGenerator:



```
package functions.threads;

import functions.Functions;
import functions.basic.Log;

import java.util.Random;

public class SimpleGenerator implements Runnable {  1 usage

    private Task task;  9 usages
    private Random rnd = new Random();  4 usages

    public SimpleGenerator(Task task) {  5 usages
        this.task = task;
    }

    @Override
    public void run() {
        for (int i = 0; i < task.getTasksCount(); i++) {
            double base = 1 + 9 * rnd.nextDouble();
            double left = rnd.nextDouble() * 100;
            double right = 100 + rnd.nextDouble() * 100;
            double step = rnd.nextDouble();

            synchronized (task) {
                task.setFunction(new Log(base));
                task.setLeftBorder(left);
                task.setRightBorder(right);
                task.setStep(step);
                task.setReady(true);
                task.notifyAll();
            }

            System.out.printf("Source %.5f %.5f %.5f%n", left, right, step);

            try { Thread.sleep( millis: 10); } catch (InterruptedException ignored) {}
```

run() — метод, который вызывается при старте потока

Класс SimpleIntegrator:

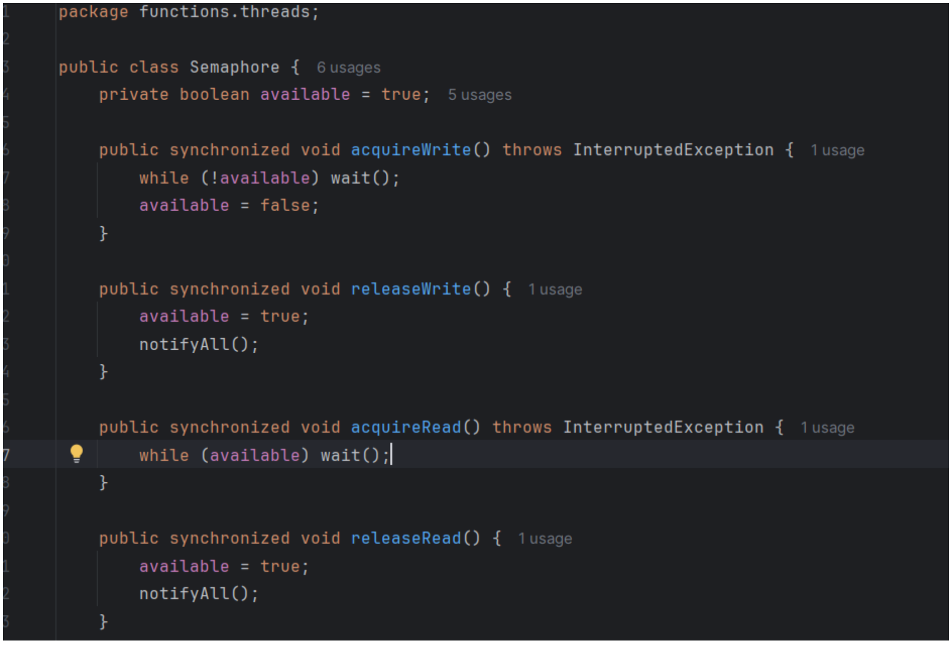


```
package functions.threads;

import functions.Functions;

public class SimpleIntegrator implements Runnable {  1 usage

    private Task task;  10 usages

    public SimpleIntegrator(Task task) {  5 usages
        this.task = task;
    }

    @Override
    public void run() {
        for (int i = 0; i < task.getTasksCount(); i++) {
            synchronized (task) {
                while (!task.isReady()) {
                    try { task.wait(); } catch (InterruptedException ignored) {}
                }

                double left = task.getLeftBorder();
                double right = task.getRightBorder();
                double step = task.getStep();
                double result = Functions.integrate(task.getFunction(), left, right, step);

                System.out.printf("Result %.5f %.5f %.5f %.10f%n", left, right, step, result);

                task.setReady(false);
            }
        }
    }
```

Задание 4

В задании 4 необходимо было использовать семафор для предотвращения пропуска задач интегратором

Класс семафора:

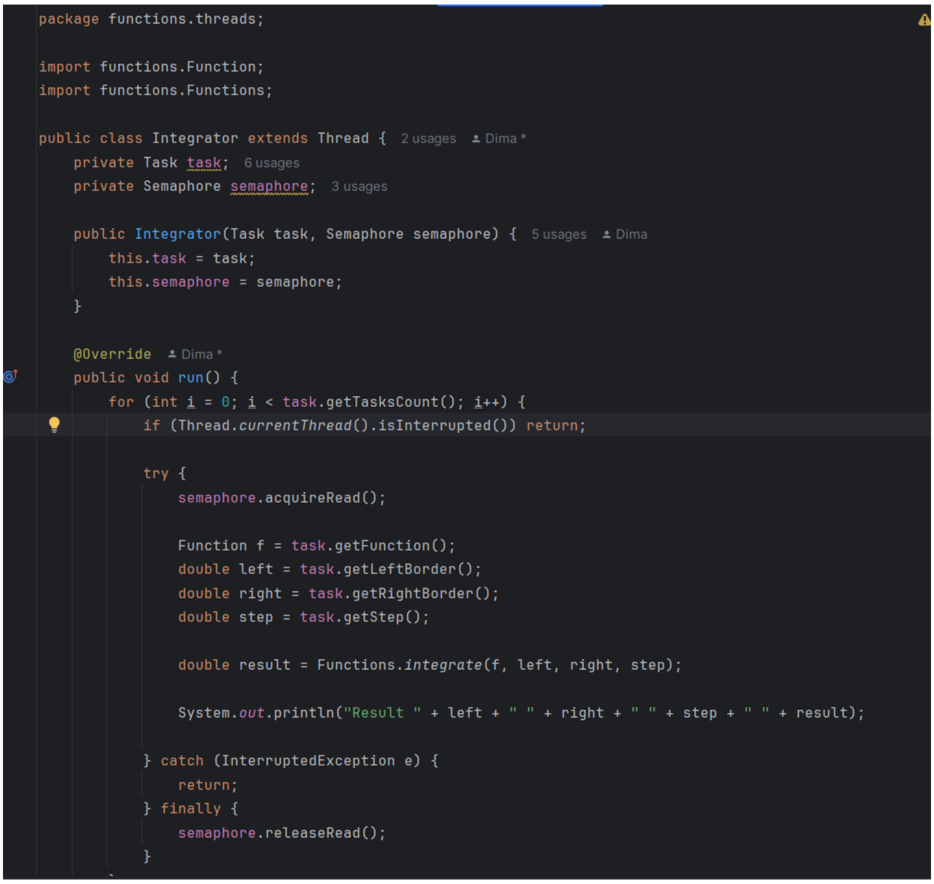


```
package functions.threads;

public class Semaphore {  6 usages
    private boolean available = true;  5 usages

    public synchronized void acquireWrite() throws InterruptedException {  1 usage
        while (!available) wait();
        available = false;
    }

    public synchronized void releaseWrite() {  1 usage
        available = true;
        notifyAll();
    }

    public synchronized void acquireRead() throws InterruptedException {  1 usage
        while (available) wait();
    }

    public synchronized void releaseRead() {  1 usage
        available = true;
        notifyAll();
    }
```

Семафор различает операции чтения и записи в защищаемый объект

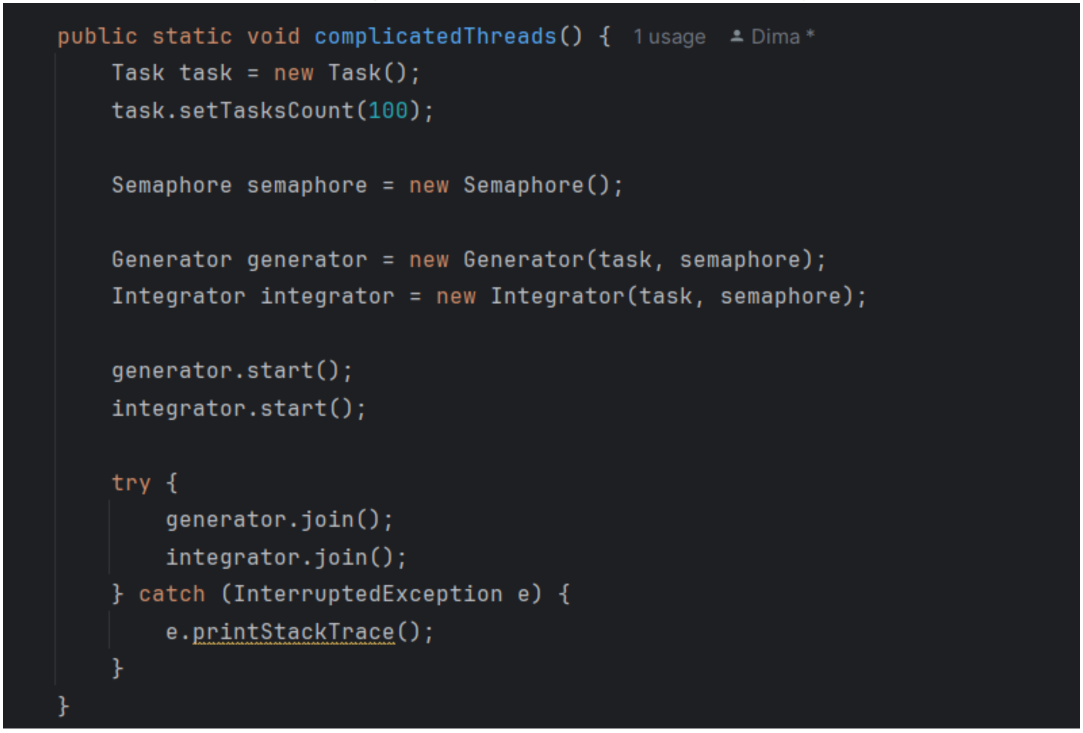
Также необходимо создать 2 класса Generator и Integrator:

Примет 1 из классов:



```
package functions.threads;

import functions.Function;
import functions.Functions;

public class Integrator extends Thread {  2 usages  Dima *
    private Task task;  6 usages
    private Semaphore semaphore;  3 usages

    public Integrator(Task task, Semaphore semaphore) {  5 usages  Dima
        this.task = task;
        this.semaphore = semaphore;
    }

    @Override  Dima *
    public void run() {
        for (int i = 0; i < task.getTasksCount(); i++) {
            if (Thread.currentThread().isInterrupted()) return;

            try {
                semaphore.acquireRead();

                Function f = task.getFunction();
                double left = task.getLeftBorder();
                double right = task.getRightBorder();
                double step = task.getStep();

                double result = Functions.integrate(f, left, right, step);

                System.out.println("Result " + left + " " + right + " " + step + " " + result);

            } catch (InterruptedException e) {
                return;
            } finally {
                semaphore.releaseRead();
            }
```

В Main надо создать метод complicatedThreads(), который создаёт и запускает два потока Generator и Integrator и проверяет их работу



```
public static void complicatedThreads() {  1 usage  Dima *
    Task task = new Task();
    task.setTasksCount(100);

    Semaphore semaphore = new Semaphore();

    Generator generator = new Generator(task, semaphore);
    Integrator integrator = new Integrator(task, semaphore);

    generator.start();
    integrator.start();

    try {
        generator.join();
        integrator.join();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}
```

Пример работы Main:

```
Проверка интегрирования экспоненты на [0,1]
Интеграл e^x на [0,1] = 1,7182819159
шаг дискретизации = 0,0007812500

Non-threads
Source 28,62037 154,59991 0,78573
Result 28,62037 154,59991 0,78573 360,4319796962
Source 54,34101 161,55941 0,55820
Result 54,34101 161,55941 0,55820 375,1251954584
Source 42,61317 155,65451 0,05775
Result 42,61317 155,65451 0,05775 326,5317563687
Source 95,80771 123,24610 0,25939
Result 95,80771 123,24610 0,25939 296,5659790460
Source 88,37920 152,92338 0,49807
Result 88,37920 152,92338 0,49807 196,2783793929
Source 0,44334 137,89109 0,77422
Result 0,44334 137,89109 0,77422 453,4391694595
Source 40,50157 155,86222 0,58322
Result 40,50157 155,86222 0,58322 381,8575066700
Source 42,22811 125,36081 0,07164
Result 42,22811 125,36081 0,07164 358,0916930974
Source 24,23860 107,74926 0,75349
Result 24,23860 107,74926 0,75349 176,5829384081
Source 8,88814 122,83003 0,98205
Result 8,88814 122,83003 0,98205 205,0453328020
Source 6,40071 172,53148 0,86563
Result 6,40071 172,53148 0,86563 1756,5909850216
Source 84,01079 121,97299 0,34021
Result 84,01079 121,97299 0,34021 86,0561052023
Source 56,30172 170,11451 0,66764
Result 56,30172 170,11451 0,66764 756,5758939096
Source 52,55464 153,23092 0,71471
Result 52,55464 153,23092 0,71471 277,9868813760
Source 39,67526 163,61169 0,98699
Result 39,67526 163,61169 0,98699 363,8304104654
Source 5,14501 133,17156 0,54027
Result 5,14501 133,17156 0,54027 224,1157812852
Source 33,44692 127,06801 0,57360
Result 33,44692 127,06801 0,57360 548,6581699236
Source 41,54536 115,95647 0,71935
```

```
Result 41,54536 115,95647 0,71935 140,8874917116
Source 74,25274 147,17456 0,78923
Result 74,25274 147,17456 0,78923 245,8019929914
Source 98,69544 114,93147 0,00952
Result 98,69544 114,93147 0,00952 34,4870491615
Source 78,59457 137,48903 0,58437
Result 78,59457 137,48903 0,58437 205,6351658449
Source 53,70669 185,35610 0,76290
Result 53,70669 185,35610 0,76290 345,3589493364
Source 97,52314 168,67795 0,45430
Result 97,52314 168,67795 0,45430 625,8507930361
Source 77,20554 184,28763 0,83407
Result 77,20554 184,28763 0,83407 2655,5827391860
Source 93,73929 143,29634 0,31559
Result 93,73929 143,29634 0,31559 260,6224642715
Source 37,41521 148,63514 0,72300
Result 37,41521 148,63514 0,72300 347,9658081670
Source 54,00402 168,62024 0,49444
Result 54,00402 168,62024 0,49444 376,5241896309
Source 80,36669 157,39951 0,26926
Result 80,36669 157,39951 0,26926 185,5671294291
Source 74,74803 106,04793 0,26838
Result 74,74803 106,04793 0,26838 67,5145266489
Source 36,62950 169,11474 0,46395
Result 36,62950 169,11474 0,46395 291,7972991471
Source 83,77134 100,58537 0,65258
Result 83,77134 100,58537 0,65258 35,6996935006
Source 29,95283 150,50634 0,19582
Result 29,95283 150,50634 0,19582 309,4232028567
Source 85,92361 118,93140 0,37232
Result 85,92361 118,93140 0,37232 103,5811259195
Source 88,20770 137,38240 0,99789
Result 88,20770 137,38240 0,99789 124,3699383700
Source 45,65584 149,67855 0,21293
Result 45,65584 149,67855 0,21293 237,8860186400
Source 82,02081 193,77291 0,11107
Result 82,02081 193,77291 0,11107 771,3331750390
Source 44,33396 144,14489 0,18863
Result 44,33396 144,14489 0,18863 203,7201954122
Source 41,49060 160,33307 0,43113
```

```
Result 41,49060 160,33307 0,43113 331,2458602674
Source 75,20510 101,35254 0,22172
Result 75,20510 101,35254 0,22172 63,9111914744
Source 26,37630 108,02605 0,15537
Result 26,37630 108,02605 0,15537 218,8983935784
Source 63,90697 125,66872 0,27346
Result 63,90697 125,66872 0,27346 182,0861739942
Source 50,15042 138,10097 0,19560
Result 50,15042 138,10097 0,19560 177,6912586318
Source 27,41117 110,62455 0,11809
Result 27,41117 110,62455 0,11809 218,3519681929
Source 83,30541 194,38168 0,67358
Result 83,30541 194,38168 0,67358 248,4020679175
Source 9,29093 127,42764 0,20138
Result 9,29093 127,42764 0,20138 254,1716361148
Source 86,30517 178,88192 0,06551
Result 86,30517 178,88192 0,06551 500,9455022739
Source 54,20390 121,68379 0,27513
Result 54,20390 121,68379 0,27513 140,4575949803
Source 47,57184 186,76254 0,81188
Result 47,57184 186,76254 0,81188 378,1372086156
Source 9,48070 146,84362 0,74899
Result 9,48070 146,84362 0,74899 765,8740631971
Source 26,89870 101,64638 0,45913
Result 26,89870 101,64638 0,45913 258,4179643835
Source 66,70981 187,88965 0,95107
Result 66,70981 187,88965 0,95107 463,2121778339
Source 29,67970 117,30647 0,64955
Result 29,67970 117,30647 0,64955 186,1059700362
Source 81,20937 153,77942 0,73868
Result 81,20937 153,77942 0,73868 163,0576716482
Source 62,34626 176,74309 0,52573
Result 62,34626 176,74309 0,52573 250,0367270013
Source 32,27664 142,16049 0,15066
Result 32,27664 142,16049 0,15066 219,2891187310
Source 44,40196 144,66541 0,36606
Result 44,40196 144,66541 0,36606 213,4356911723
Source 12,10562 104,87262 0,04263
Result 12,10562 104,87262 0,04263 159,4647244800
Source 77,44579 154,39330 0,51766
```

```
Result 77,44579 154,39330 0,51766 178,9047503199
Source 56,26353 167,75306 0,89668
Result 56,26353 167,75306 0,89668 1297,5544206389
Source 96,57452 195,19494 0,94317
Result 96,57452 195,19494 0,94317 242,0619898953
Source 53,61673 183,26038 0,24682
Result 53,61673 183,26038 0,24682 310,4315578570
Source 54,55484 131,07574 0,94767
Result 54,55484 131,07574 0,94767 196,8273169749
Source 98,59678 119,43253 0,41161
Result 98,59678 119,43253 0,41161 56,9551616908
Source 14,15592 167,03449 0,90236
Result 14,15592 167,03449 0,90236 1588,2352177683
Source 73,69713 113,69481 0,47536
Result 73,69713 113,69481 0,47536 92,8270865871
Source 49,03301 103,80202 0,03124
Result 49,03301 103,80202 0,03124 158,5524608309
Source 75,09126 145,03320 0,70943
Result 75,09126 145,03320 0,70943 221,3060953628
Source 78,55262 147,07819 0,57723
Result 78,55262 147,07819 0,57723 244,6592722707
Source 19,26267 103,04893 0,87483
Result 19,26267 103,04893 0,87483 179,5369688049
Source 86,95009 162,28285 0,91421
Result 86,95009 162,28285 0,91421 163,1231964623
Source 93,53750 157,40972 0,09767
Result 93,53750 157,40972 0,09767 159,3470082850
Source 2,20995 190,83628 0,42123
Result 2,20995 190,83628 0,42123 414,2591548619
Source 38,41041 156,21331 0,25888
Result 38,41041 156,21331 0,25888 232,5129215697
Source 35,05987 168,98479 0,18550
Result 35,05987 168,98479 0,18550 270,0309791577
Source 74,55425 106,77909 0,08948
Result 74,55425 106,77909 0,08948 77,5580081175
Source 3,46871 115,18719 0,85040
Result 3,46871 115,18719 0,85040 191,7524275732
Source 12,40416 117,38324 0,41294
Result 12,40416 117,38324 0,41294 648,5910944944
Source 93,03944 177,07760 0,17497
```

```
Result 93,03944 177,07760 0,17497 223,0238483594
Source 7,20686 103,45314 0,22596
Result 7,20686 103,45314 0,22596 193,8554960613
Source 28,68733 154,02029 0,12135
Result 28,68733 154,02029 0,12135 260,0709657367
Source 47,42554 186,73664 0,55908
Result 47,42554 186,73664 0,55908 290,3722151748
Source 57,99427 136,27228 0,53646
Result 57,99427 136,27228 0,53646 452,1665173284
Source 89,39115 140,96176 0,30860
Result 89,39115 140,96176 0,30860 566,5763259719
Source 71,62970 126,76914 0,74440
Result 71,62970 126,76914 0,74440 144,3789170083
Source 76,09069 162,50254 0,48107
Result 76,09069 162,50254 0,48107 909,9280337014
Source 11,22879 136,54973 0,61599
Result 11,22879 136,54973 0,61599 447,0335760213
Source 2,98296 169,56662 0,54193
Result 2,98296 169,56662 0,54193 474,9377518826
Source 40,11860 169,83551 0,93182
Result 40,11860 169,83551 0,93182 289,3184578739
Source 18,73268 158,42015 0,80954
Result 18,73268 158,42015 0,80954 530,9808325667
Source 77,28265 139,77997 0,10514
Result 77,28265 139,77997 0,10514 127,0279241921
Source 73,75329 170,16567 0,08620
Result 73,75329 170,16567 0,08620 294,2779277915
Source 19,95479 180,90390 0,58127
Result 19,95479 180,90390 0,58127 335,4310207364
Source 54,44939 146,58862 0,96732
Result 54,44939 146,58862 0,96732 307,8491472769
Source 5,23264 149,23204 0,47951
Result 5,23264 149,23204 0,47951 561,4177013531
Source 66,04910 178,58810 0,52038
Result 66,04910 178,58810 0,52038 11698,2913517611
Source 86,10168 126,14678 0,29441
Result 86,10168 126,14678 0,29441 118,5153292501
Source 51,91722 175,70740 0,27496
Result 51,91722 175,70740 0,27496 277,8461328348
Source 22,26014 166,97453 0,72241
```

```
Result 22,26014 166,97453 0,72241 344,8007212672
Source 62,23972 110,28009 0,71523
Result 62,23972 110,28009 0,71523 129,1399524075
Source 11,13567 160,32039 0,93059
Result 11,13567 160,32039 0,93059 571,5983405798

Simple threads
Source 72,97448 195,78189 0,00295
Result 72,97448 195,78189 0,00295 356,5634916248
Source 58,43348 180,47562 0,08444
Result 58,43348 180,47562 0,08444 1538,3411647318
Source 37,06369 145,65599 0,76832
Result 37,06369 145,65599 0,76832 248,2004541979
Source 63,66768 148,16274 0,73036
Result 63,66768 148,16274 0,73036 203,3854354338
Source 99,70739 144,28457 0,71625
Result 99,70739 144,28457 0,71625 207,8713693822
Source 77,31533 198,21752 0,64560
Result 77,31533 198,21752 0,64560 440,3180283665
Source 50,51665 107,68194 0,99263
Result 50,51665 107,68194 0,99263 109,2480510332
Source 39,81754 150,82721 0,48228
Result 39,81754 150,82721 0,48228 223,5421934934
Source 22,50684 158,65060 0,74285
Result 22,50684 158,65060 0,74285 1674,8366065923
Source 20,33390 195,37997 0,19018
Result 20,33390 195,37997 0,19018 372,6375989977
Source 88,66751 185,61698 0,08312
Result 88,66751 185,61698 0,08312 236,1978555402
Source 16,80671 161,51065 0,38686
Result 16,80671 161,51065 0,38686 400,5791886554
Source 7,80605 115,82864 0,80766
Result 7,80605 115,82864 0,80766 222,0071076232
Source 17,99412 137,51593 0,12174
Result 17,99412 137,51593 0,12174 224,1821075624
Source 1,30552 175,17884 0,71081
Result 1,30552 175,17884 0,71081 537,7218065131
Source 21,61296 120,90449 0,03831
Result 21,61296 120,90449 0,03831 209,5462656104
Source 67,43755 131,92371 0,65271
```

```
Result 67,43755 131,92371 0,65271 284,8616100057
Source 27,73175 110,84788 0,18816
Result 27,73175 110,84788 0,18816 316,1861023985
Result 64,87073 146,55811 0,74773 339,9415555963
Source 64,87073 146,55811 0,74773
Source 51,55462 177,29752 0,00089
Result 51,55462 177,29752 0,00089 382,4258776219
Source 22,01514 175,11531 0,30430
Result 22,01514 175,11531 0,30430 374,2873571722
Source 24,73674 166,53368 0,16375
Result 24,73674 166,53368 0,16375 563,0142284726
Source 53,84380 198,15661 0,39531
Result 53,84380 198,15661 0,39531 314,1631000547
Source 79,37178 127,91196 0,41487
Result 79,37178 127,91196 0,41487 112,2393763306
Source 92,83145 172,71609 0,18774
Result 92,83145 172,71609 0,18774 324,3664782941
Source 33,79223 171,57286 0,50409
Result 33,79223 171,57286 0,50409 372,4087034399
Source 65,65142 129,46643 0,41057
Result 65,65142 129,46643 0,41057 136,6031988775
Source 24,32304 191,88015 0,82520
Result 24,32304 191,88015 0,82520 368,9405295627
Source 72,27119 162,37123 0,58985
Result 72,27119 162,37123 0,58985 218,1569389179
Source 52,12918 116,35364 0,47353
Result 52,12918 116,35364 0,47353 133,5892557906
Source 49,08726 183,08957 0,92981
Result 49,08726 183,08957 0,92981 428,4214123901
Source 39,29798 164,86188 0,73282
Result 39,29798 164,86188 0,73282 977,1661070938
Source 22,73993 137,39837 0,96897
Result 22,73993 137,39837 0,96897 308,1167880665
Source 13,29657 128,84058 0,24526
Result 13,29657 128,84058 0,24526 234,1798664597
Source 86,24868 105,89821 0,39005
Result 86,24868 105,89821 0,39005 53,4548376345
Source 63,94522 105,30227 0,36857
Result 63,94522 105,30227 0,36857 271,1082763271
Source 54,10863 136,26823 0,61275
```

```
Result 54,10863 136,26823 0,61275 168,1879657262
Source 42,04728 145,40280 0,47412
Result 42,04728 145,40280 0,47412 317,0472811093
Source 77,98897 179,33567 0,04812
Result 77,98897 179,33567 0,04812 437,1037363091
Source 93,81605 102,33039 0,18985
Result 93,81605 102,33039 0,18985 22,5182336094
Source 29,67365 126,22583 0,14356
Result 29,67365 126,22583 0,14356 211,7604466208
Source 50,23959 188,28283 0,84207
Result 50,23959 188,28283 0,84207 493,3558111401
Source 82,88526 168,85787 0,59736
Result 82,88526 168,85787 0,59736 278,1216758426
Source 15,52376 175,29840 0,59987
Result 15,52376 175,29840 0,59987 325,3780803348
Source 50,63250 131,05228 0,37152
Result 50,63250 131,05228 0,37152 226,9173101843
Source 23,75430 199,70617 0,07624
Result 23,75430 199,70617 0,07624 3365,1653220151
Source 59,72202 142,51550 0,38203
Result 59,72202 142,51550 0,38203 197,9636483595
Source 57,91857 154,96138 0,38958
Result 57,91857 154,96138 0,38958 210,6040948138
Source 69,07516 152,96732 0,99522
Result 69,07516 152,96732 0,99522 200,4661919318
Source 50,42744 182,63136 0,93971
Result 50,42744 182,63136 0,93971 463,0417003015
Source 63,49023 108,67684 0,70934
Result 63,49023 108,67684 0,70934 1178,8660852208
Source 9,27817 196,29778 0,40222
Result 9,27817 196,29778 0,40222 371,5236415312
Source 79,49103 138,67236 0,65010
Result 79,49103 138,67236 0,65010 129,2584615245
Source 38,99866 198,14074 0,69493
Result 38,99866 198,14074 0,69493 1012,3093562858
Source 44,90213 122,89110 0,75917
Result 44,90213 122,89110 0,75917 164,3012875501
Source 50,51392 172,97043 0,92062
Result 50,51392 172,97043 0,92062 279,8613838498
Source 38,79155 139,53418 0,80449
```

```
Result 38,79155 139,53418 0,80449 387,8162392786
Source 79,07521 169,47183 0,53973
Result 79,07521 169,47183 0,53973 249,2960841130
Result 3,42162 113,89252 0,59138 229,1133080599
Source 3,42162 113,89252 0,59138
Source 54,39261 103,42827 0,27703
Result 54,39261 103,42827 0,27703 114,1594156101
Source 64,84836 115,90935 0,00324
Result 64,84836 115,90935 0,00324 133,5619778786
Result 92,11155 142,68932 0,51613 354,5585423312
Source 92,11155 142,68932 0,51613
Source 43,81030 178,74495 0,75086
Result 43,81030 178,74495 0,75086 505,4786510166
Source 0,77394 158,64867 0,18573
Result 0,77394 158,64867 0,18573 290,4958226476
Source 63,54716 135,60911 0,95913
Result 63,54716 135,60911 0,95913 158,6102214482
Source 23,40459 153,99770 0,39162
Result 23,40459 153,99770 0,39162 370,4436342154
Source 86,18909 197,12717 0,97019
Result 86,18909 197,12717 0,97019 290,1262847625
Source 57,86350 101,52029 0,29817
Result 57,86350 101,52029 0,29817 93,6169038125
Source 34,35715 116,50500 0,99181
Result 34,35715 116,50500 0,99181 248,4996320961
Source 25,10457 153,69449 0,01329
Result 25,10457 153,69449 0,01329 266,1469078734
Source 79,61551 114,75879 0,46995
Result 79,61551 114,75879 0,46995 70,7669805986
Source 79,48154 107,86673 0,48088
Result 79,48154 107,86673 0,48088 125,6207429025
Source 4,41615 163,27509 0,31447
Result 4,41615 163,27509 0,31447 775,6003051421
Source 80,24154 136,76414 0,63444
Result 80,24154 136,76414 0,63444 153,8605303319
Source 69,17659 145,96679 0,35672
Result 69,17659 145,96679 0,35672 172,4157383472
Source 68,34718 119,11066 0,31599
Result 68,34718 119,11066 0,31599 465,4521368585
Source 41,13127 148,29077 0,66816
```

```
Result 41,13127 148,29077 0,66816 245,0962267480
Source 17,76746 130,11250 0,71398
Result 17,76746 130,11250 0,71398 214,2064577657
Source 66,59735 145,39263 0,61869
Result 66,59735 145,39263 0,61869 164,6475289522
Source 55,37841 176,57864 0,32436
Result 55,37841 176,57864 0,32436 627,7587637110
Source 93,29526 186,64429 0,80375
Result 93,29526 186,64429 0,80375 229,3857134661
Source 68,61394 186,44195 0,68841
Result 68,61394 186,44195 0,68841 256275,7559312411
Source 74,36499 198,84007 0,21898
Result 74,36499 198,84007 0,21898 1972,3988519546
Source 45,00758 187,44656 0,41149
Result 45,00758 187,44656 0,41149 297,0134896382
Source 10,71845 155,31677 0,31151
Result 10,71845 155,31677 0,31151 372,7597995396
Source 12,76565 134,76715 0,21648
Result 12,76565 134,76715 0,21648 298,5185812544
Source 13,36865 183,26355 0,78657
Result 13,36865 183,26355 0,78657 398,1298956624
Source 76,51264 172,32570 0,03591
Result 76,51264 172,32570 0,03591 209,4015224937
Source 92,37637 106,48699 0,29847
Result 92,37637 106,48699 0,29847 34,5637035815
Source 67,34925 137,57142 0,39652
Result 67,34925 137,57142 0,39652 182,1241644544
Source 4,05479 145,89724 0,40490
Result 4,05479 145,89724 0,40490 1170,1558192390
Result 51,36388 144,48941 0,60040 473,1802560884
Source 51,36388 144,48941 0,60040
Source 13,07291 124,98067 0,41326
Result 13,07291 124,98067 0,41326 231,1469227482
Source 65,73803 104,60373 0,45639
Result 65,73803 104,60373 0,45639 129,3913398756
Source 51,09820 152,52469 0,99815
Result 51,09820 152,52469 0,99815 263,1107974800
Source 76,58639 170,74043 0,56576
Result 76,58639 170,74043 0,56576 198,9855232341
Source 78,61921 179,86047 0,71877
```

```
Result 78,61921 179,86047 0,71877 334,6702110205
Source 77,67214 113,31630 0,41612
Result 77,67214 113,31630 0,41612 110,2681762348
Source 81,58870 154,88906 0,81516
Result 81,58870 154,88906 0,81516 165,7697311028
Source 74,13813 163,91880 0,78727
Result 74,13813 163,91880 0,78727 444,7347519709
```

Complicated threads

```
Source 89.86136510851682 177.6655185161762 0.21692713070360603
Source 37.818506905058044 197.46559913354537 0.5500523933860743
Source 28.963480215165315 101.63583157157498 0.9249296899776042
Source 99.14270835346916 152.656148918343 0.7450231656321324
Source 96.49476539770919 108.10307143235738 0.08728409888867639
Source 42.96675331035627 180.56116484807094 0.030289654236670294
Source 71.94696673860201 147.40948613163584 0.1504201990847044
Source 95.874299388629 148.64523102375043 0.20954289507466428
Source 17.938777715438626 187.56696894637977 0.050833734862088842
Source 65.36651627837945 153.925030518528 0.2437843125386932 6
Source 88.28628789593085 122.0567011853269 0.0751032248916792
Source 37.80377579009478 137.55914251561535 0.06913674080436549
Source 9.663562821731286 195.2129918432864 0.3395967439088188
Source 15.757731561277877 128.15485944425012 0.7237985270043337
Source 24.743900808092256 173.70700033883077 0.8312748288026095
Source 95.164434922565 154.80768891025537 0.8505467910476296
Source 9.712440908861375 117.64590524526442 0.20099593588107678
Source 7.669092068707172 195.8891295943826 0.8911965463951129
Source 4.040734083967967 129.80320895715647 0.7507304327793463
Source 78.55310083486623 154.59838069960966 0.5394213914493164
Source 15.413907860476572 186.4957479078481 0.306089782755501
Source 75.6339893599272 177.29357784586824 0.26834240881425786
Source 87.70102981883889 178.62306326249347 0.9155765993674341
Source 49.528877800942695 155.61820232738572 0.999291257754982
Source 44.85931882187221 116.38475609128217 0.5679624304410372
Source 90.67208761558535 178.4382767476837 0.2281539555137097
Source 48.66281900385011 176.19143649886155 0.3691990312794293
Source 53.477532242710744 124.6059887320112 0.1757013852300533
Source 33.146157870085645 193.28490934105787 0.8702697401544046
Source 23.03650699238442 112.56533710580376 0.7783129285113983
Source 3.654020660727941 3 128.58320078950788 0.4873868873931585
Source 24.741497819372228 147.8154549049332 0.5854993881152992
Source 13.386958603862043 163.2660899819842 0.20594175424392214
Source 2.2266498882477115 134.2472281932552 0.8806314865040219
Source 55.33416395669132 199.61670755071617 0.009613243801564253
Source 14.015501448012213 140.36881662444586 0.5912801573598861
Source 32.48383318518218 171.63489283363816 0.7260412958890508
Source 24.266632120020425 136.435731231479 0.8055749243266384
Source 94.20173851496368 193.93788869720186 0.23047184147517075
Source 88.00782143135217 143.50106420486483 0.9609866205651836
Source 51.66864186984874 102.46402181725963 0.1987788591378451
Source 56.24936595236496 160.40151794736266 0.021940616992667472
Source 10.267167705479041 165.4949898149643 0.8512702471970363
Source 51.36716949701988 159.69484679273998 0.21266687981080046
Source 81.80876949951865 168.51235913576738 0.37868375202941107
Source 88.14623611826352 168.5643430300716 0.12169315185228557
Source 64.49140417072414 101.40393753677014 0.9917789285812965
Source 40.87518706258133 175.70598136620865 0.24438745169137055
Source 3.4785439729098755 101.53561240452282 0.7859315598565948
Source 81.88242769071466 130.2138720943632 0.2799559391782285
Source 69.99015876258764 131.20430312230695 0.8621368913758078
Source 41.9045877638311 123.47960208370922 0.7493029597413746
Source 21.79627359938332 192.98762392440233 0.39582766779858225
Source 67.67786631754674 170.12113993497752 0.7501587722254411
Source 33.00374873411316 111.50592408282579 0.5949134245634744
Source 77.1429007977997 128.95690453351344 0.9158819141695501
Source 6.026896629376788 135.1122802573587 0.738447252705342
Source 37.65378348533503 130.0406115049558 0.8943418508546349
Source 3.6325629509820434 136.02612350057015 0.7373261246256146
Source 17.581857548787095 198.77255119424416 0.9132887204639211
Source 63.95028095965245 112.42606159254956 0.9084313640543005
Source 86.70490904759708 193.10289553677742 0.24861594834478717
Source 86.33817019975781 121.01757845199731 0.9036151170109782
Source 65.77239244015833 194.84267626656037 0.6619221627366053
Source 0.9951315450481668 193.59120631611634 0.9849393548495723
Source 50.63729325244001 121.0980579071927 0.12377889275793752
Source 60.210041535628164 116.0867935763664 0.08757363127260742
Source 61.34527961858477 146.84351744315714 0.3891098328171789
Source 14.138518182719807 173.67951090118152 0.36175752090271174
Source 91.57174262344093 154.6495801942932 0.24108831070221404
Source 58.99789188505083 190.4032058667256 0.13523469890454465
Source 58.797376413466665 152.58069473364841 0.4699590638881421
Source 70.57104527555825 116.82674053114806 0.487563970846729
Source 80.61289130044396 126.27893287943928 0.9031905217343849
Source 60.909665490219126 144.29892291847227 0.8777143091545112
Source 27.543516039846605 116.24908046497782 0.05336394397939126
Source 21.127638987865083 149.2776810036819 0.5163497641362735
Source 27.783890703938486 136.3607758600317 0.2002057100618002
Source 38.17511974680596 196.259076009035 0.42421049382133746
Source 24.234001116039817 174.58332229717695 0.9731687147955036
Source 98.48395551857737 152.8218043482287 0.36799633938572684
Source 9.226733169238098 141.96878105772095 0.7509695151047605
Source 16.37506331434031 195.30280067466393 0.11515898923765411
Source 87.33774046429066 160.8989812565096 0.4444872653031897
Source 50.11840974698334 108.61956082305227 0.6990091361876557
Source 41.668183473427845 108.61208382757681 0.4794306013350529
Source 93.53619112974579 108.75484105903989 0.6355449017431394
Source 3.991006097655714 157.0242885112925 0.04517902555271858
Source 56.03985330819328 141.41930399731717 0.28340750726917496
Source 54.1513592332733 174.8623609330442 0.9262046916783954
Source 68.99157220339589 135.40779218980452 0.78837083113374
Source 99.11180368759067 183.1225395168442 0.4824810595676403
Source 81.28084025854696 114.16482582141535 0.30377079432634957
Source 7.032911745317505 179.87152391081378 0.5952847143035017
Source 32.00673878867952 194.07494218505133 0.4282810125026071
Source 96.7798870596216 175.4050979624132 0.6447963200499018
Source 13.156561542285717 122.99353223291781 0.0524407985046329
Source 8.245848246358989 103.7800539026404 0.8359073086669246
Source 60.17140747245755 137.53376430866672 0.05534194585932961
Source 3.8823802092557202 101.73504304769916 0.43863828096719315
```