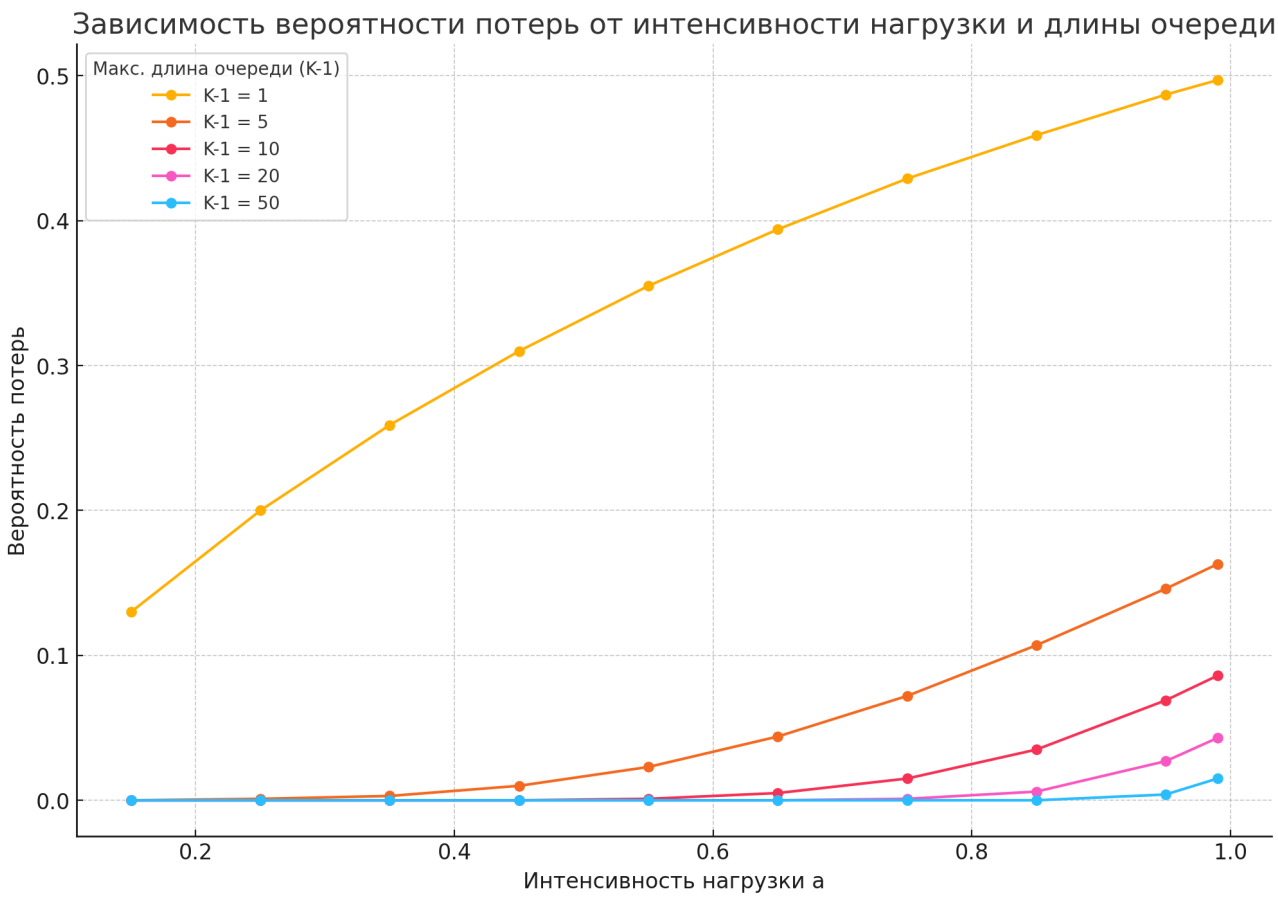
Код 5.1:

```
public class Solevoin {

    public static void main(String[] args) {
        // Значения интенсивности нагрузки a, как в таблице
        double[] aValues = {0.15, 0.25, 0.35, 0.45, 0.55, 0.65, 0.75, 0.85, 0.95, 0.99};

        // Значения максимальной длины очереди (K-1), как в таблице
        int[] kValues = {1, 5, 10, 20, 50};

        // Значение времени обслуживания t
        double t = 1.0; // Можно изменить при необходимости

        // Заголовок таблицы
        System.out.println("   a   | K-1=1 (AM) | K-1=5 (AM) | K-1=10 (AM) | K-1=20 (AM) | K-1=50 (AM)");

System.out.println("-----------------------------------------------------------------------");

        // Цикл по всем значениям a и k для вычисления вероятности потерь
        for (double a : aValues) {
```

```
            System.out.printf("%.2f |", a);
            for (int k : kValues) {
                double p = calculateLossProbability(a,
t, k);
                System.out.printf(" %.3f |", p);
            }
            System.out.println();
        }
    }

    /**
     * Метод для вычисления вероятности потерь.
     *
     * @param a интенсивность нагрузки
     * @param t время обслуживания
     * @param k максимальная длина очереди (K-1)
     * @return вероятность потерь
     */
    public static double
calculateLossProbability(double a, double t, int k) {
        double rho = a * t;
        double numerator = (1 - rho) * Math.pow(rho,
k);
        double denominator = 1 - Math.pow(rho, k + 1);
        return denominator != 0 ? numerator /
denominator : 0;
    }
```

```
---------------------------------------------------------------------
0.15 | 0.130 | 0.000 | 0.000 | 0.000 | 0.000 |
0.25 | 0.200 | 0.001 | 0.000 | 0.000 | 0.000 |
0.35 | 0.259 | 0.003 | 0.000 | 0.000 | 0.000 |
0.45 | 0.310 | 0.010 | 0.000 | 0.000 | 0.000 |
0.55 | 0.355 | 0.023 | 0.001 | 0.000 | 0.000 |
0.65 | 0.394 | 0.044 | 0.005 | 0.000 | 0.000 |
0.75 | 0.429 | 0.072 | 0.015 | 0.001 | 0.000 |
0.85 | 0.459 | 0.107 | 0.035 | 0.006 | 0.000 |
0.95 | 0.487 | 0.146 | 0.069 | 0.027 | 0.004 |
0.99 | 0.497 | 0.163 | 0.086 | 0.043 | 0.015 |
```

A	ИМ	АМ	ИМ	АМ	ИМ	АМ	ИМ	АМ	ИМ	АМ
0.15	0.019	0.130	0	0	0	0	0	0	0	0
0.25	0.047	0.200	0	0.001	0	0	0	0	0	0
0.35	0.083	0.259	0.001	0.003	0	0	0	0	0	0
0.45	0.123	0.310	0.004	0.010	0	0	0	0	0	0
0.55	0.165	0.355	0.013	0.023	0.003	0.001	0	0	0	0
0.65	0.203	0.394	0.027	0.044	0.01	0.005	0.001	0	0	0
0.75	0.242	0.429	0.052	0.072	0.029	0.015	0.004	0.001	0	0
0.85	0.281	0.459	0.085	0.107	0.057	0.035	0.023	0.006	0.003	0
0.95	0.313	0.487	0.115	0.146	0.079	0.069	0.043	0.027	0.011	0.004
0.99	0.327	0.497	0.14	0..163	0.079	0.086	0.043	0.043	0.011	0.015



Код 5.2:

```
public class Solevar {

    private double ca;       // Коэффициент вариации
времени между заявками C_a

    private double cb;       // Коэффициент вариации
времени обслуживания C_b (равен 1)

    private double tau;      // Средний интервал между
заявками (τ)

    private int capacity;    // Максимальная длина
очереди (K)

    private double t = 1.0; // Время обслуживания (t,
всегда равно 1)

    public Solevar(double ca, double tau, int capacity)
{
```

```
        this.ca = ca;
        this.cb = 1.0;       // По условию C_b = 1
        this.tau = tau;
        this.capacity = capacity;
    }

    // Метод для вычисления коэффициента нагрузки ρ
    private double calculateRho() {
        return t / tau; // ρ = t / τ
    }

    // Метод для расчета вероятности потерь по формуле
    public double calculateLossProbability() {
        double rho = calculateRho(); // Рассчитываем ρ

        // Проверка на случай, если rho приближается к 1
        if (Math.abs(rho - 1.0) < 1e-6) {
            // Используем предельное значение вероятности потерь при ρ → 1
            double lossProbability = (Math.pow(ca, 2) + Math.pow(cb, 2)) / (2.0 * (capacity + 1));
            return lossProbability;
        }

        double exponentPart = 2 / (Math.pow(ca, 2) + Math.pow(cb, 2)); // Часть формулы с экспонентой
```

```
        double denominator = 1 - Math.pow(rho,
exponentPart * (capacity + 1)); // Знаменатель
        double numerator = (1 - rho) * Math.pow(rho,
exponentPart * capacity); // Числитель

        if (denominator == 0) {
            System.out.println("Ошибка: знаменатель
равен 0.");
            return Double.NaN; // Возвращаем NaN, если
знаменатель равен 0
        }

        // Финальная формула для вероятности потерь
        double lossProbability = numerator /
denominator;
        return lossProbability;
    }


    public static void main(String[] args) {
        // Массив данных для двух случаев: K = 5 и K = 10
        // Каждый элемент: {C_a для K=5, τ для K=5, K=5,
C_a для K=10, τ для K=10, K=10}
        double[][] data = {
                {0.447, 10.0, 5, 0.408, 6.6, 10},
                {0.577, 3.9, 5, 0.5, 3.6, 10},
                {0.707, 2.8, 5, 0.447, 2.5, 10},
```

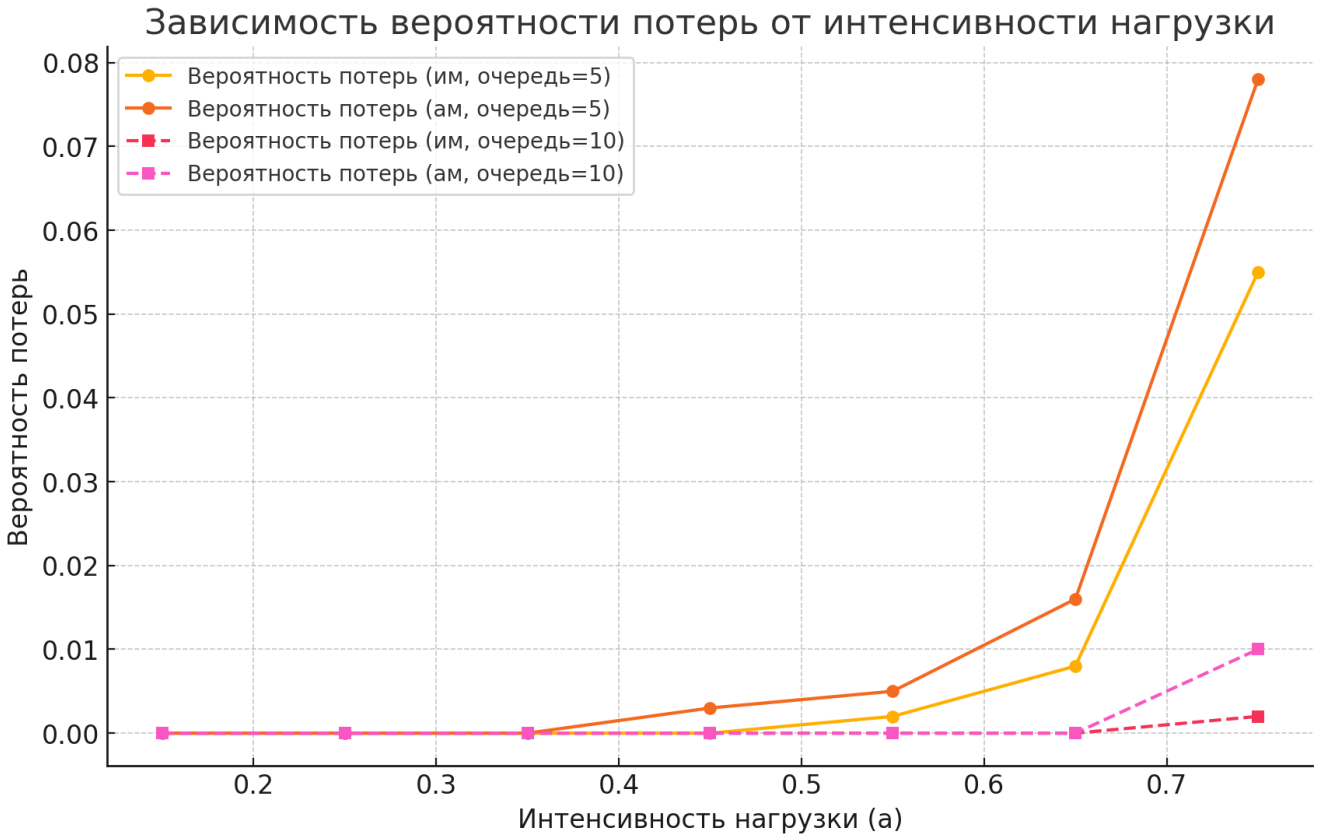
```
                {0.69, 2.1, 5, 0.577, 2.1, 10},
                {0.707, 2.0, 5, 0.577, 1.8, 10},
                {0.577, 1.5, 5, 0.707, 1.6, 10},
                {1.0, 1.3, 5, 1.0, 1.4, 10},
                {0.707, 1.2, 5, 0.707, 1.2, 10},
                {0.608, 1.08, 5, 0.447, 1.0, 10},
                {1.0, 1.0, 5, 0.707, 1.0, 10}
        };

        // Расчет для каждого набора данных
        for (int i = 0; i < data.length; i++) {
            // Для K = 5
            double ca5 = data[i][0];
            double tau5 = data[i][1];
            int capacity5 = (int) data[i][2];

            Solevar calculator5 = new Solevar(ca5,
tau5, capacity5);
            double lossProbability5 =
calculator5.calculateLossProbability();
            System.out.printf("Строка %d для K=5:
Вероятность потерь: %.15f\n", (i + 1), lossProbability5);

            // Для K = 10
            double ca10 = data[i][3];
            double tau10 = data[i][4];
            int capacity10 = (int) data[i][5];
```

```
            Solevar calculator10 = new Solevar(ca10,
tau10, capacity10);
            double lossProbability10 =
calculator10.calculateLossProbability();
            System.out.printf("Строка %d для K=10:
Вероятность потерь: %.15f\n", (i + 1),
lossProbability10);
        }
    }
}
```

N	*a*	Максимальная длина очереди (*K*-1)											
		5						10					
		C_a	C_b	τ	*t*	им	ам	C_a	C_b	τ	*t*	им	ам
1	0,15	0,447	1	10	1	0	0	0,408	1	6,6	1	0	0
2	0,25	0,577	1	3,9	1	0	0	0,5	1	3,6	1	0	0
3	0,35	0,707	1	2,8	1	0	0	0,447	1	2,5	1	0	0
4	0,45	0,69	1	2,1	1	0	0,003	0,577	1	2,1	1	0	0
5	0,55	0,707	1	2	1	0.002	0,005	0,577	1	1,8	1	0	0
6	0,65	0,577	1	1,5	1	0,008	0,016	0,707	1	1,6	1	0	0
7	0,75	1	1	1,3	1	0,055	0,078	0,707	1	1,4	1	0,002	0,010

им – имитационное моделирование, ам – аналитическое моделирование. *t* – среднее время обслуживания,

τ – среднее интервала между заявками.




```
/Library/Java/JavaVirtualMachines/jdk-20.jdk/Contents/Home/
Строка 1 для K=5: Вероятность потерь: 0.000000004164689
Строка 1 для K=10: Вероятность потерь: 0.000000000000008
Строка 2 для K=5: Вероятность потерь: 0.000027353077803
Строка 2 для K=10: Вероятность потерь: 0.000000000907467
Строка 3 для K=5: Вероятность потерь: 0.000671236681104
Строка 3 для K=10: Вероятность потерь: 0.000000139560328
Строка 4 для K=5: Вероятность потерь: 0.003445942340671
Строка 4 для K=10: Вероятность потерь: 0.000007663372160
Строка 5 для K=5: Вероятность потерь: 0.004938557970199
Строка 5 для K=10: Вероятность потерь: 0.000065704476647
Строка 6 для K=5: Вероятность потерь: 0.016339071629417
Строка 6 для K=10: Вероятность потерь: 0.000712193481635
Строка 7 для K=5: Вероятность потерь: 0.078394296658130
Строка 7 для K=10: Вероятность потерь: 0.010127697172143
```