# Отчет по лабораторной работе по предмету Математические основы защиты информации и информационной безопасности

## Лабораторная работа №5. Вероятностные алгоритмы проверки чисел на простоту

Никита Андреевич Топонен

# Содержание

# Список иллюстраций

# Список таблиц

# 1 Цель работы

Цель работы — изучить вероятностные алгоритмы проверки чисел на простоту.

# 2 Задание

- Реализовать алгоритмы:
    1. Алгоритм, реализующий тест Ферма;
    2. Алгоритм вычисления символа Якоби;
    3. Алгоритм, реализующий тест Соловэя-Штрассена;
    4. Алгоритм, реализующий тест Миллера-Рабина.

# 3 Теоретическое введение

Тестом простоты (или проверкой простоты) называется алгоритм, который, приняв на входе число $N$, позволяет либо не подтвердить предположение о том, является ли это число составным, либо точно утверждать его простоту. Во втором случае он называется истинным тестом простоты. Таким образом, тест простоты представляет собой только гипотезу о том, что если алгоритм не подтвердил предположение о составности числа $N$, то это число может являться простым с определённой вероятностью. Это определение подразумевает меньшую уверенность в соответствии результата проверки истинному положению вещей, нежели истинное испытание на простоту, которое даёт математически подтверждённый результат.

Существуют различные вероятностные тесты простоты числа. К этой категории относятся:

1. Тест Ферма.
2. Тест Миллера — Рабина.
3. Тест Соловэя — Штрассена.
4. Тест Бейли — Померанца — Селфриджа — Уогстаффа.
5. Квадратичный тест Фробениуса.

В данной лабораторной рассмотрим первые 3. Также нам понадобится алгоритм вычисления символа Якоби.

# 4 Выполнение лабораторной работы

## 4.1 Вспомогательные функции

Для данной лабораторной работы определил вспомогательные функции получения случайного числа из интервала, а также для вычисления числа $a^b(modc)$:

```
private int getRandomInt(int min, int max) {
    Random random = new Random();
    return random.nextInt(max - min) + min;
}

public int modPow(int a, int b, int c) {
    int res = 1;
    for (int i = 0; i < b; i++) {
        res *= a;
        res %= c;
    }
    return res % c;
}
```

## 4.2 Тест Ферма

В рамках данной лабораторной работы я реализовал тест Ферма на языке Java. Ниже приведен код:

```
private String fermat(int n) {
    if (n % 2 == 0 && n < 5) {
        throw new RuntimeException("n must uneven and be greater or equal than 5"
    }

    int a = getRandomInt(2, n - 2);
    int r = modPow(a, n - 1, n);

    if (r == 1) {
        return String.format("%s is probably prime", n);
    } else {
        return String.format("%s is probably composite", n);
    }
}
```

## 4.3 Вычисления символа Якоби

В рамках данной лабораторной работы я реализовал вычисление символа Якоби на языке Java. Ниже приведен код:

```
private int jacobi(int n, int a) {
    if (n % 2 == 0 || n < 3 || a < 0 || a >= n) {
        throw new RuntimeException("n must be uneven and more than 2, " +
            "and a must be positive and less than n");
    }

    int g = 1;
    int a1;
    int s = 0;
```

```
do {
    if (a == 0) {
        return 0;
    }
    if (a == 1) {
        return g;
    }

    int k = 0;
    a1 = a;
    while (a1 % 2 == 0) {
        a1 = a1 / 2;
        k++;
    }

    if (k % 2 == 0) {
        s = 1;
    } else {
        if ((n - 1) % 8 == 0 || (n + 1) % 8 == 0) {
            s = 1;
        } else if ((n - 3) % 8 == 0 || (n + 3) % 8 == 0) {
            s = -1;
        }
    }

    if (a1 != 1) {
        if ((n - 3) % 4 == 0 && (a1 - 3) % 4 == 0) {
            s = -s;
        }
```

```
            a = modPow(n, 1, a1);
            n = a1;
            g = g * s;
        }
    } while (a1 != 1);

    return g * s;
}
```

## 4.4 Тест Соловэя — Штрассена

В рамках данной лабораторной работы я реализовал тест Соловэя — Штрассена на языке Java. Ниже приведен код:

```
private String solovejShtrassen(int n) {
    if (n % 2 == 0 || n < 5) {
        throw new RuntimeException("n must be uneven and more than 4");
    }

    int a = getRandomInt(2, n - 2);
    int r = modPow(a, (n-1)/2, n);

    if (r != 1 && r != (n - 1)) {
        return String.format("%s is composite", n);
    }

    int s = jacobi(n, a);
    if ((r - s) % n == 0) {
        return String.format("%s is composite", n);
```

```
    } else {
        return String.format("%s is probably prime", n);
    }
}
```

## 4.5 Тест Миллера — Рабина

В рамках данной лабораторной работы я реализовал тест Миллера — Рабина на языке Java. Ниже приведен код:

```
private String millerRabin(int n) {
    if (n % 2 == 0 || n < 5) {
        throw new RuntimeException("n must be uneven and more than 4");
    }

    int s = 0;
    int r = 0;
    int nEven = n - 1;

    while (nEven % 2 == 0) {
        nEven = nEven / 2;
        s++;
    }
    r = nEven;

    int a = getRandomInt(2, n-2);
    int y = modPow(a, r, n);

    if (y != 1 && y != (n - 1)) {
        for (int i = 1; i <= (s - 1) && y != (n - 1) && y != 1; i++) {
```

```
            y = modPow(y, 2, n);
        }
        if (y == 1 || y != (n - 1)) {
            return String.format("%s is composite", n);
        }
    }

    return String.format("%s is probably prime", n);
}
```

# 5 Выводы

В рамках данной лабораторной работы я изучил и реализовал на языке Java вероятностные алгоритмы проверки чисел на простоту.

# Список литературы