Лабораторная работа №5

Реализация вероятностных алгоритмов проверки чисел на простоту. Markdown

Исламова С.М.

Информация

Докладчик

- Исламова Сания Маратовна
- студент уч. группы НПИмд-01-24
- Российский университет дружбы народов
- 1132249576@pfur.ru
- https://github.com/SaniyaIslamova26

Вводная часть

Актуальность

- Реализация (разработка) вероятностных алгоритмов проверки чисел на простоту на языке Julia, чтобы понять принципы работы алгоритмов, git, Markdown.

Объект и предмет исследования

- Вероятностные алгоритмы проверки чисел на простоту
- Тест Ферма
- Тест Соловэя-Штрассена
- Тест Миллера-Рабина
- Алгоритм вычисления символа Якоби
- Веб-сервис GitHub
- Язык разметки Markdown

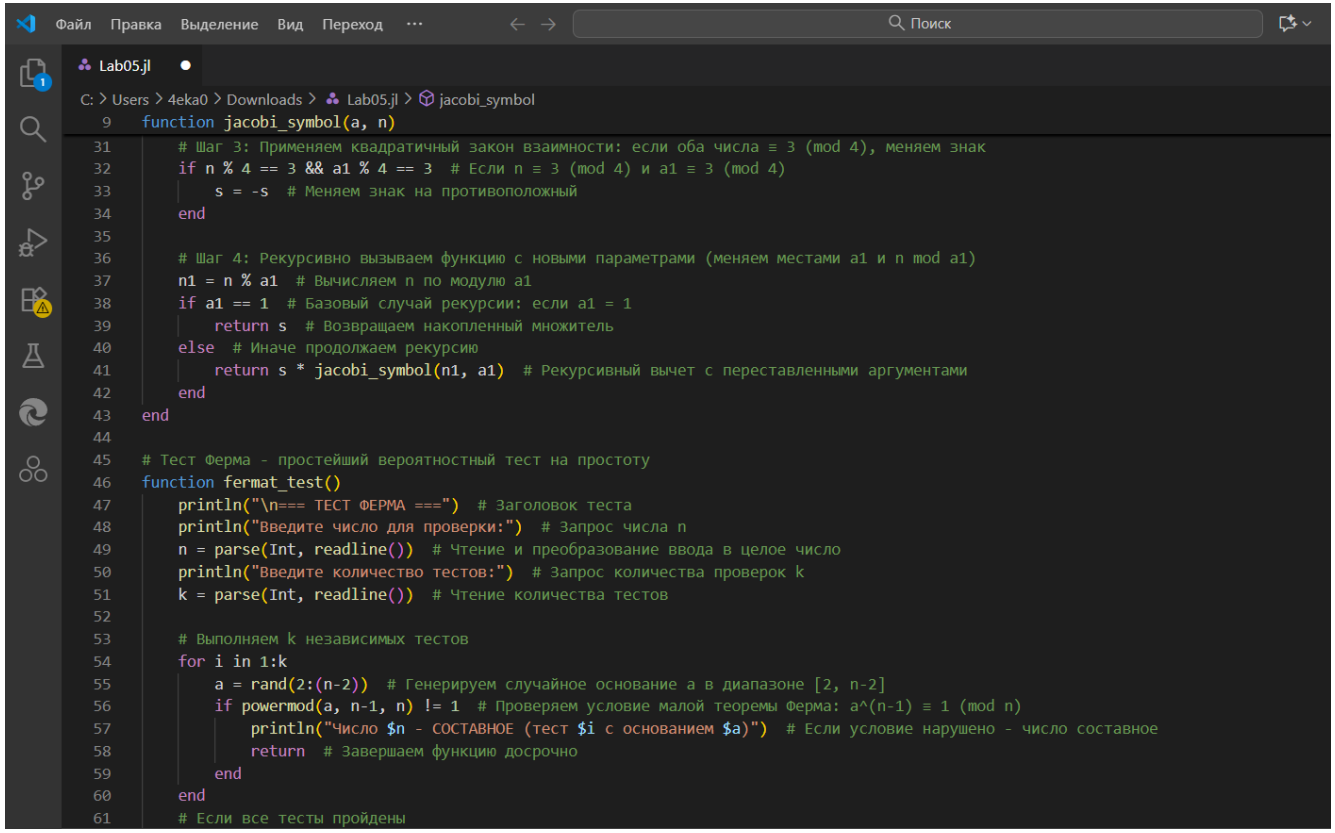
Цели и задачи

- Реализовать вероятностные алгоритмы проверки чисел на простоту. Работа с Markdown.

Процесс выполнения работы

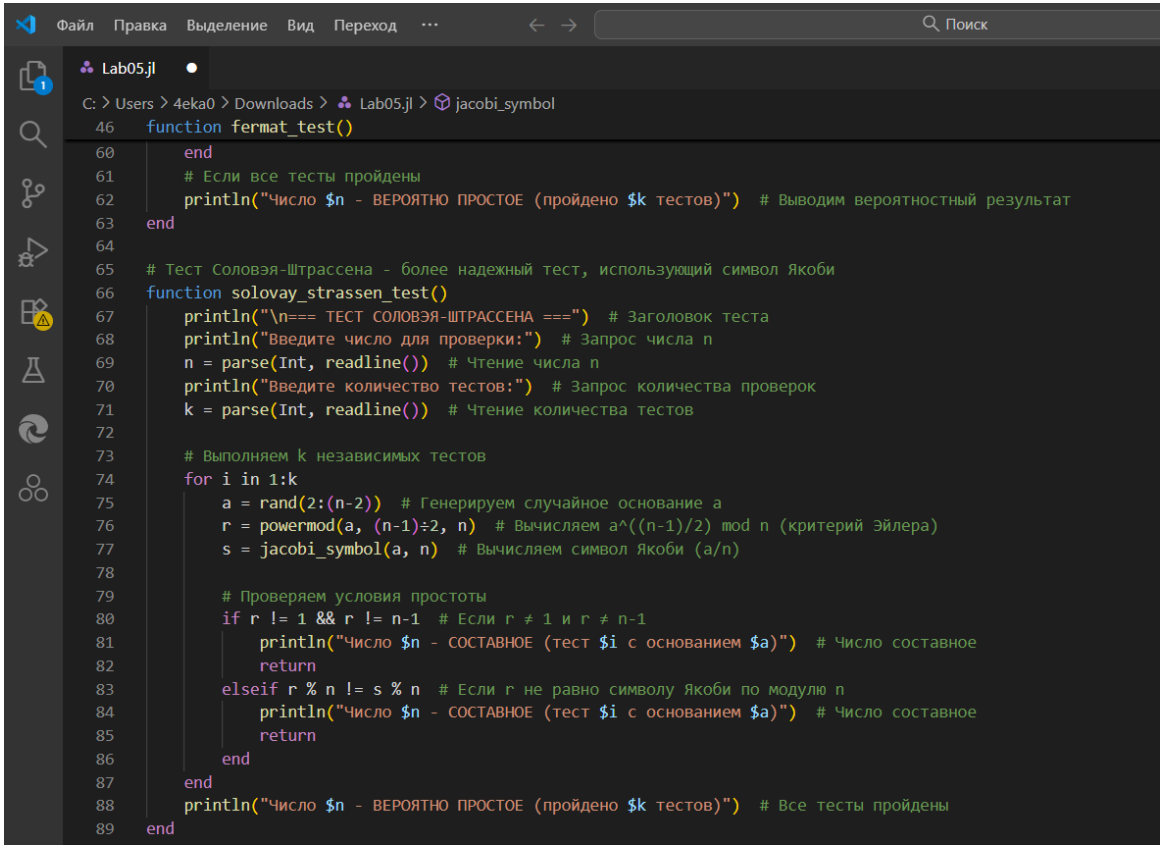
Реализовать на языке программирования Julia вероятностные алгоритмы проверки чисел на простоту

Тест Ферма



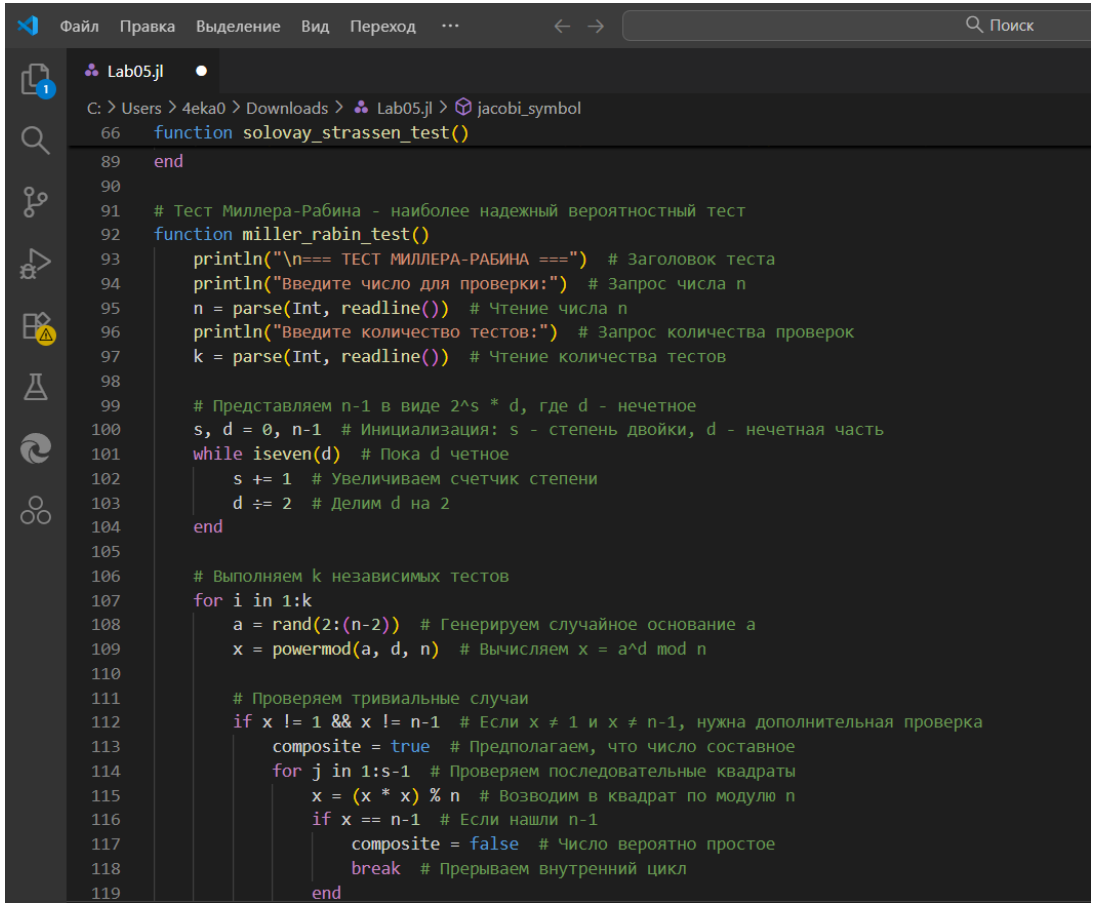

```julia
function jacobi_symbol(a, n)
    # Шаг 3: Применяем квадратичный закон взаимности: если оба числа ≡ 3 (mod 4), меняем знак
    if n % 4 == 3 && a1 % 4 == 3  # Если n ≡ 3 (mod 4) и a1 ≡ 3 (mod 4)
        s = -s  # Меняем знак на противоположный
    end

    # Шаг 4: Рекурсивно вызываем функцию с новыми параметрами (меняем местами a1 и n mod a1)
    n1 = n % a1  # Вычисляем n по модулю a1
    if a1 == 1  # Базовый случай рекурсии: если a1 = 1
        return s  # Возвращаем накопленный множитель
    else  # Иначе продолжаем рекурсию
        return s * jacobi_symbol(n1, a1)  # Рекурсивный вычет с переставленными аргументами
    end
end

# Тест Ферма - простейший вероятностный тест на простоту
function fermat_test()
    println("\n=== ТЕСТ ФЕРМА ===")  # Заголовок теста
    println("Введите число для проверки:")  # Запрос числа n
    n = parse(Int, readline())  # Чтение и преобразование ввода в целое число
    println("Введите количество тестов:")  # Запрос количества проверок k
    k = parse(Int, readline())  # Чтение количества тестов

    # Выполняем k независимых тестов
    for i in 1:k
        a = rand(2:(n-2))  # Генерируем случайное основание a в диапазоне [2, n-2]
        if powermod(a, n-1, n) != 1  # Проверяем условие малой теоремы Ферма: a^(n-1) ≡ 1 (mod n)
            println("Число $n - СОСТАВНОЕ (тест $i с основанием $a)")  # Если условие нарушено - число составное
            return  # Завершаем функцию досрочно
        end
    end
    # Если все тесты пройдены
```

Тест Соловэя-Штрассена



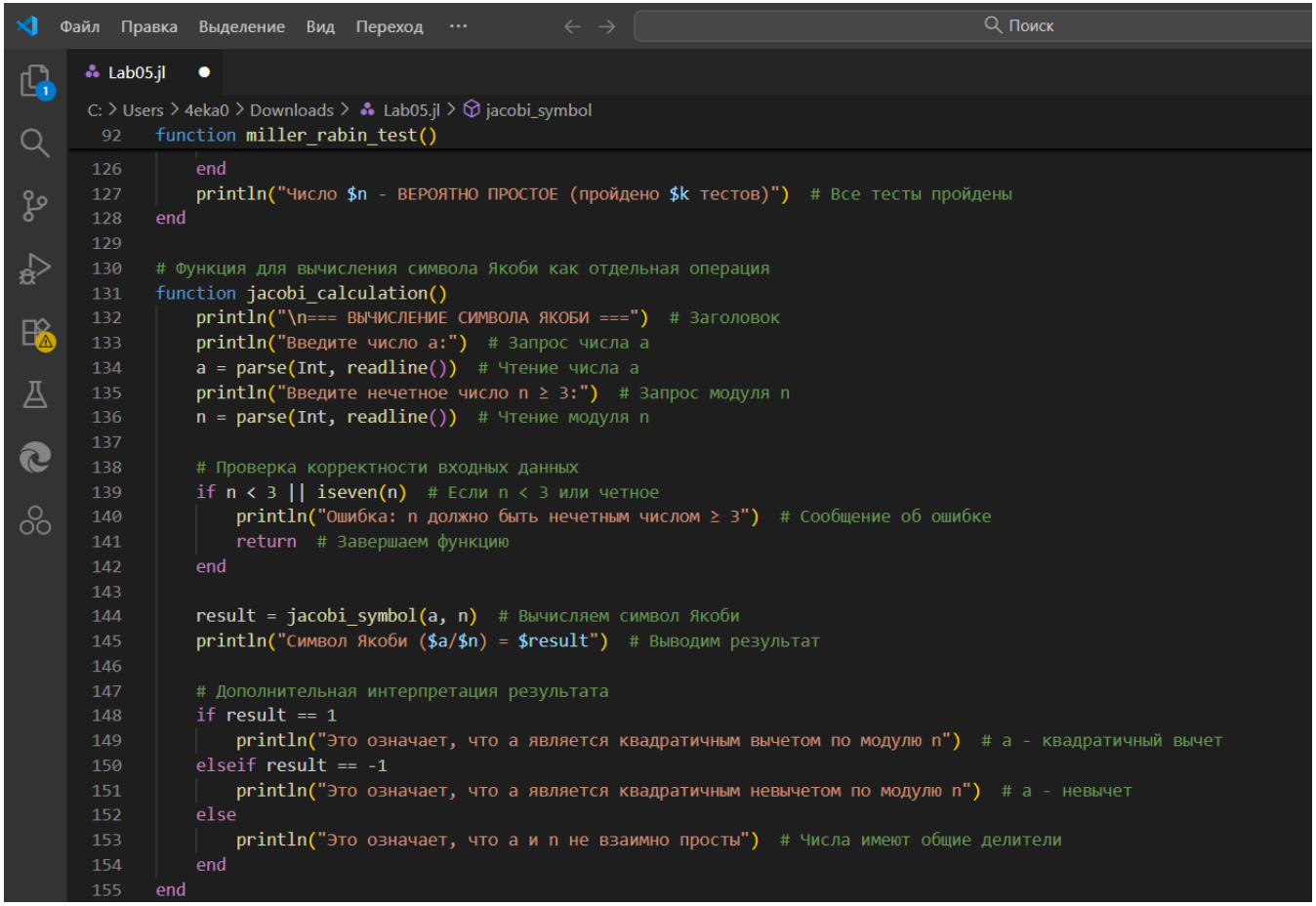

```
46  function fermat_test()
60      end
61      # Если все тесты пройдены
62      println("Число $n - ВЕРОЯТНО ПРОСТОЕ (пройдено $k тестов)")  # Выводим вероятностный результат
63  end
64
65  # Тест Соловэя-Штрассена - более надежный тест, использующий символ Якоби
66  function solovay_strassen_test()
67      println("\n=== ТЕСТ СОЛОВЭЯ-ШТРАССЕНА ===")  # Заголовок теста
68      println("Введите число для проверки:")  # Запрос числа n
69      n = parse(Int, readline())  # Чтение числа n
70      println("Введите количество тестов:")  # Запрос количества проверок
71      k = parse(Int, readline())  # Чтение количества тестов
72
73      # Выполняем k независимых тестов
74      for i in 1:k
75          a = rand(2:(n-2))  # Генерируем случайное основание a
76          r = powermod(a, (n-1)÷2, n)  # Вычисляем a^((n-1)/2) mod n (критерий Эйлера)
77          s = jacobi_symbol(a, n)  # Вычисляем символ Якоби (a/n)
78
79          # Проверяем условия простоты
80          if r != 1 && r != n-1  # Если r ≠ 1 и r ≠ n-1
81              println("Число $n - СОСТАВНОЕ (тест $i с основанием $a)")  # Число составное
82              return
83          elseif r % n != s % n  # Если r не равно символу Якоби по модулю n
84              println("Число $n - СОСТАВНОЕ (тест $i с основанием $a)")  # Число составное
85              return
86          end
87      end
88      println("Число $n - ВЕРОЯТНО ПРОСТОЕ (пройдено $k тестов)")  # Все тесты пройдены
89  end
```

Тест Миллера-Рабина



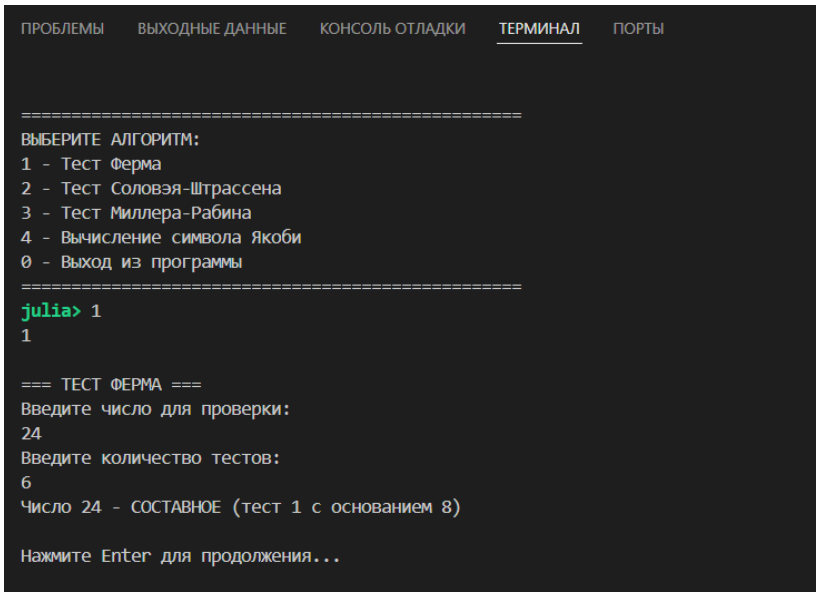

```
66  function solovay_strassen_test()
89  end
90
91  # Тест Миллера-Рабина - наиболее надежный вероятностный тест
92  function miller_rabin_test()
93      println("\n=== ТЕСТ МИЛЛЕРА-РАБИНА ===")  # Заголовок теста
94      println("Введите число для проверки:")  # Запрос числа n
95      n = parse(Int, readline())  # Чтение числа n
96      println("Введите количество тестов:")  # Запрос количества проверок
97      k = parse(Int, readline())  # Чтение количества тестов
98
99      # Представляем n-1 в виде 2^s * d, где d - нечетное
100     s, d = 0, n-1  # Инициализация: s - степень двойки, d - нечетная часть
101     while iseven(d)  # Пока d четное
102         s += 1  # Увеличиваем счетчик степени
103         d ÷= 2  # Делим d на 2
104     end
105
106     # Выполняем k независимых тестов
107     for i in 1:k
108         a = rand(2:(n-2))  # Генерируем случайное основание a
109         x = powermod(a, d, n)  # Вычисляем x = a^d mod n
110
111         # Проверяем тривиальные случаи
112         if x != 1 && x != n-1  # Если x ≠ 1 и x ≠ n-1, нужна дополнительная проверка
113             composite = true  # Предполагаем, что число составное
114             for j in 1:s-1  # Проверяем последовательные квадраты
115                 x = (x * x) % n  # Возводим в квадрат по модулю n
116                 if x == n-1  # Если нашли n-1
117                     composite = false  # Число вероятно простое
118                     break  # Прерываем внутренний цикл
119                 end
```

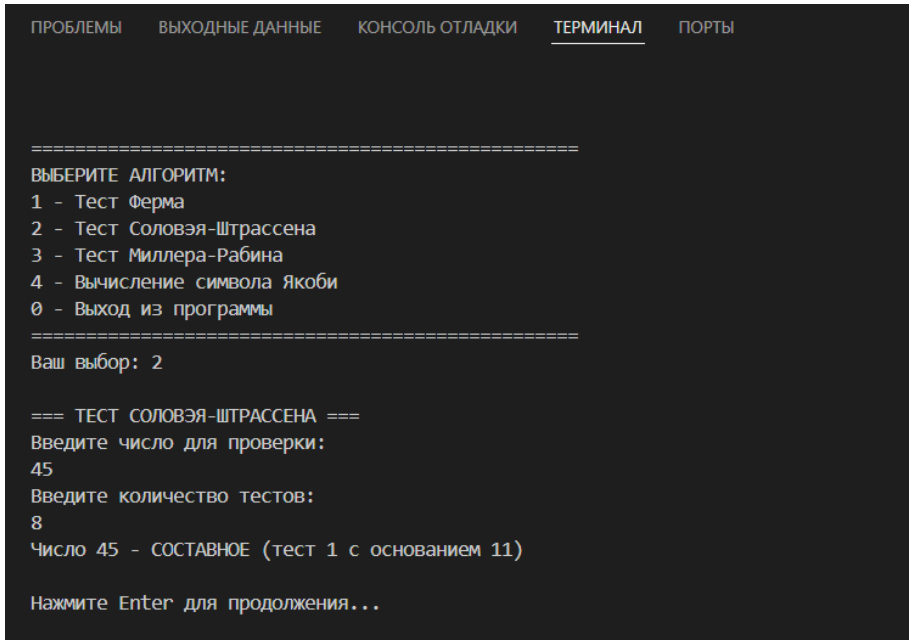
Алгоритм вычисления символа Якоби



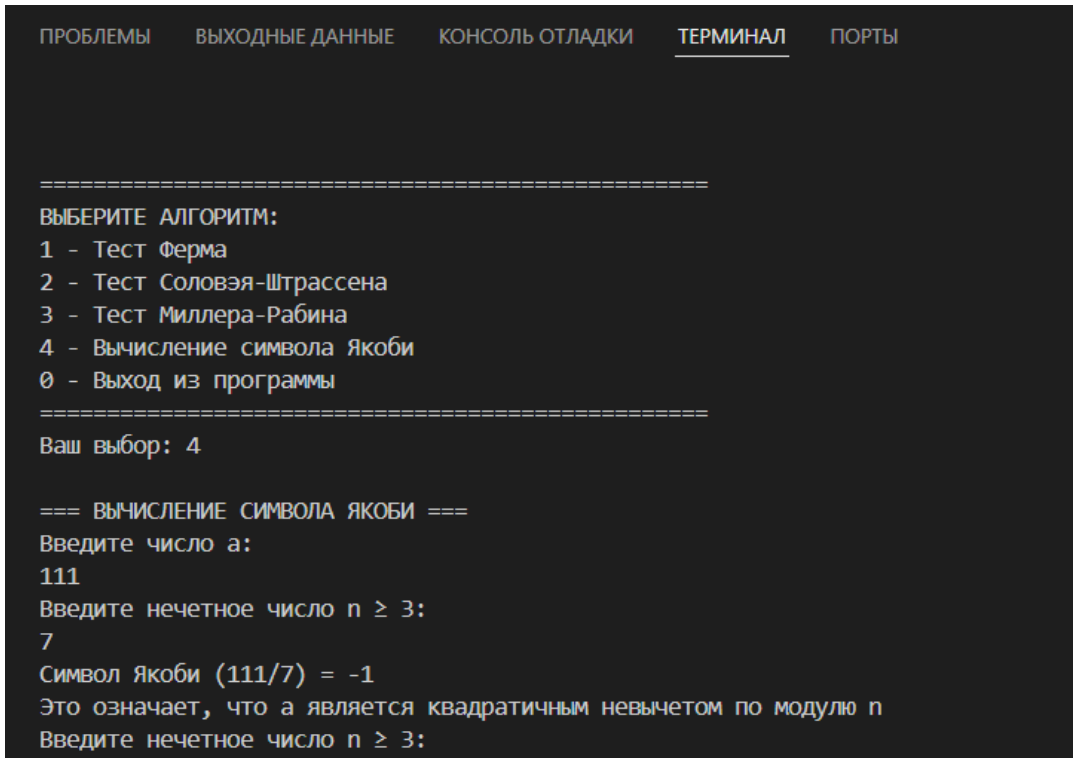

```
92  function miller_rabin_test()
126     end
127     println("Число $n - ВЕРОЯТНО ПРОСТОЕ (пройдено $k тестов)")  # Все тесты пройдены
128 end
129
130 # Функция для вычисления символа Якоби как отдельная операция
131 function jacobi_calculation()
132     println("\n=== ВЫЧИСЛЕНИЕ СИМВОЛА ЯКОБИ ===")  # Заголовок
133     println("Введите число a:")  # Запрос числа a
134     a = parse(Int, readline())  # Чтение числа a
135     println("Введите нечетное число n ≥ 3:")  # Запрос модуля n
136     n = parse(Int, readline())  # Чтение модуля n
137
138     # Проверка корректности входных данных
139     if n < 3 || iseven(n)  # Если n < 3 или четное
140         println("Ошибка: n должно быть нечетным числом ≥ 3")  # Сообщение об ошибке
141         return  # Завершаем функцию
142     end
143
144     result = jacobi_symbol(a, n)  # Вычисляем символ Якоби
145     println("Символ Якоби ($a/$n) = $result")  # Выводим результат
146
147     # Дополнительная интерпретация результата
148     if result == 1
149         println("Это означает, что a является квадратичным вычетом по модулю n")  # a - квадратичный вычет
150     elseif result == -1
151         println("Это означает, что a является квадратичным невычетом по модулю n")  # a - невычет
152     else
153         println("Это означает, что a и n не взаимно просты")  # Числа имеют общие делители
154     end
155 end
```

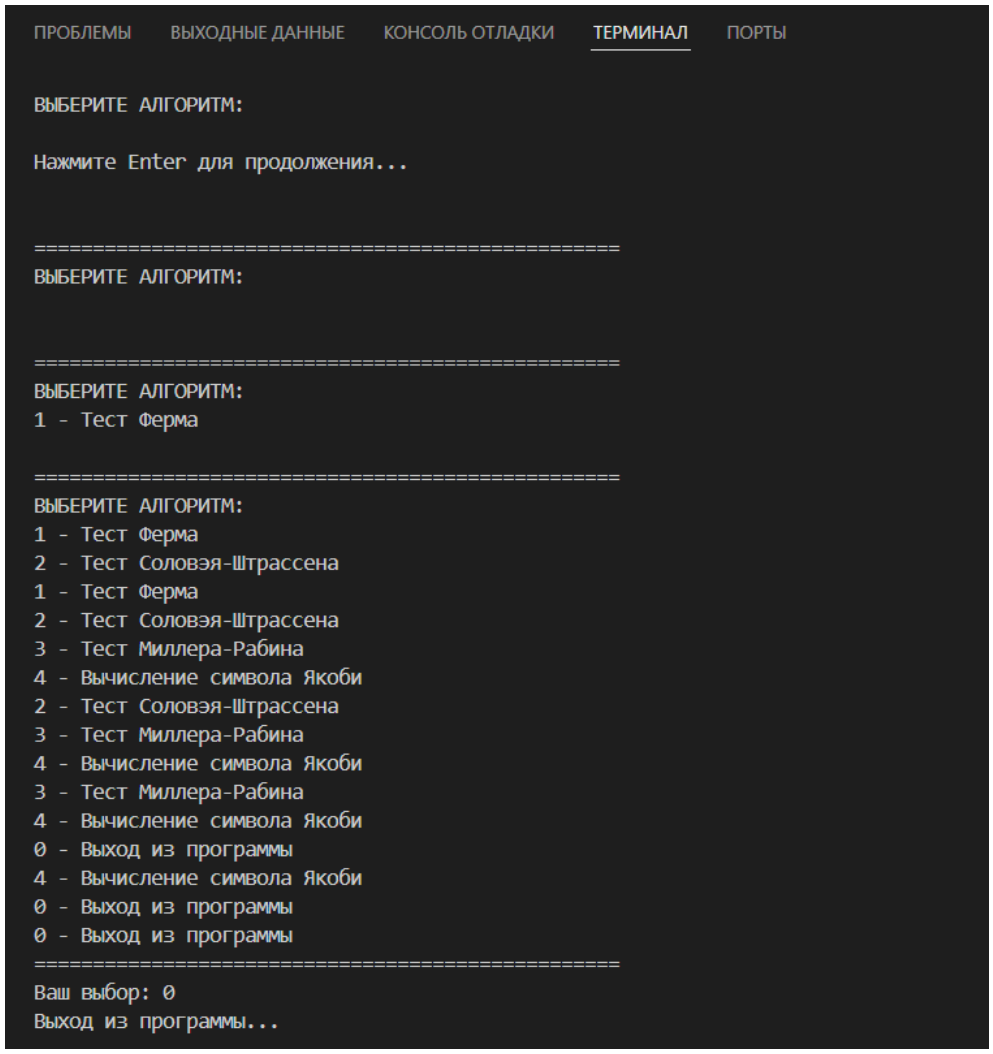
Результаты работы программы через терминал


ПРОБЛЕМЫ
ВЫХОДНЫЕ ДАННЫЕ
КОНСОЛЬ ОТЛАДКИ
ТЕРМИНАЛ
ПОРТЫ
==========
ВЫБЕРИТЕ АЛГОРИТМ:
1 - Тест Ферма
2 - Тест Соловэя-Штрассена
3 - Тест Миллера-Рабина
4 - Вычисление символа Якоби
0 - Выход из программы
==========
julia> 1
1
=== ТЕСТ ФЕРМА ===
Введите число для проверки:
24
Введите количество тестов:
6
Число 24 - СОСТАВНОЕ (тест 1 с основанием 8)
Нажмите Enter для продолжения...


ПРОБЛЕМЫ
ВЫХОДНЫЕ ДАННЫЕ
КОНСОЛЬ ОТЛАДКИ
ТЕРМИНАЛ
ПОРТЫ
==========
ВЫБЕРИТЕ АЛГОРИТМ:
1 - Тест Ферма
2 - Тест Соловэя-Штрассена
3 - Тест Миллера-Рабина
4 - Вычисление символа Якоби
0 - Выход из программы
==========
Ваш выбор: 2
=== ТЕСТ СОЛОВЭЯ-ШТРАССЕНА ===
Введите число для проверки:
45
Введите количество тестов:
8
Число 45 - СОСТАВНОЕ (тест 1 с основанием 11)
Нажмите Enter для продолжения...

```
================================================
ВЫБЕРИТЕ АЛГОРИТМ:
1 - Тест Ферма
2 - Тест Соловэя-Штрассена
3 - Тест Миллера-Рабина
4 - Вычисление символа Якоби
0 - Выход из программы
================================================
Ваш выбор: 3

=== ТЕСТ МИЛЛЕРА-РАБИНА ===
Введите число для проверки:
78
Введите количество тестов:
9
Число 78 - СОСТАВНОЕ (тест 1 с основанием 3)

Нажмите Enter для продолжения...
```


ПРОБЛЕМЫ
ВЫХОДНЫЕ ДАННЫЕ
КОНСОЛЬ ОТЛАДКИ
ТЕРМИНАЛ
ПОРТЫ
==========
ВЫБЕРИТЕ АЛГОРИТМ:
1 - Тест Ферма
2 - Тест Соловэя-Штрассена
3 - Тест Миллера-Рабина
4 - Вычисление символа Якоби
0 - Выход из программы
==========
Ваш выбор: 4
=== ВЫЧИСЛЕНИЕ СИМВОЛА ЯКОБИ ===
Введите число a:
111
Введите нечетное число n ≥ 3:
7
Символ Якоби (111/7) = -1
Это означает, что a является квадратичным невычетом по модулю n
Введите нечетное число n ≥ 3:


ПРОБЛЕМЫ
ВЫХОДНЫЕ ДАННЫЕ
КОНСОЛЬ ОТЛАДКИ
ТЕРМИНАЛ
ПОРТЫ
ВЫБЕРИТЕ АЛГОРИТМ:
Нажмите Enter для продолжения...
==========
ВЫБЕРИТЕ АЛГОРИТМ:
==========
ВЫБЕРИТЕ АЛГОРИТМ:
1 - Тест Ферма
==========
ВЫБЕРИТЕ АЛГОРИТМ:
1 - Тест Ферма
2 - Тест Соловэя-Штрассена
1 - Тест Ферма
2 - Тест Соловэя-Штрассена
3 - Тест Миллера-Рабина
4 - Вычисление символа Якоби
2 - Тест Соловэя-Штрассена
3 - Тест Миллера-Рабина
4 - Вычисление символа Якоби
3 - Тест Миллера-Рабина
4 - Вычисление символа Якоби
0 - Выход из программы
4 - Вычисление символа Якоби
0 - Выход из программы
0 - Выход из программы
==========
Ваш выбор: 0
Выход из программы...

Результаты

- Выполнены все необходимые действия для реализации задач лабораторной работы №5: успешно реализованы все вероятностные алгоритмы проверки чисел на простоту.

Вывод

Реализованывероятностные алгоритмы проверки чисел на простоту на языке Julia.