

Математические основы защиты информации и информационной безопасности

Отчет по лабораторной работе № 5

Кейела Патачона НПИМд-02-21

Содержание

1	Цель работы	4
2	Теоретические сведения	5
2.1	Тест Ферма	5
2.2	Символ Якоби	5
2.3	Алгоритм , реализующий тест Соловея - Штрассена	6
2.4	Алгоритм , реализующий тест Миллера - Рабина	6
3	Выполнение работы	8
3.1	Реализация алгоритмов на языке Python	8
3.2	Контрольный пример	11
4	Выводы	13
	Список литературы	14

List of Figures

3.1	Тест Ферма	11
3.2	Символ Якоби	11
3.3	Алгоритм , реализующий тест Соловея - Штрассена	12
3.4	Алгоритм , реализующий тест Миллера - Рабина	12

1 Цель работы

Изучить вероятностные алгоритмы проверки чисел на простоту.

2 Теоретические сведения

2.1 Тест Ферма

Вход. Нечетное целое число $n \geq 5$.

Выход. “Число n , вероятно, простое” или “Число n составное”.

1. Выбрать случайное целое число a , $2 \leq a \leq n-2$.
2. Вычислить $r = a^{(n-1)} \pmod n$.
3. Если $r = 0$ результат : “Число n , вероятно, простое”. В противном случае результат: “Число n составное”.

2.2 Символ Якоби

Вход. Нечетное целое число $n \geq 3$, целое число a , $0 \leq a < n$.

Выход. Символ Якоби.

1. $g=1$
2. если $a = 0$ результат: 0
3. если $a = 1$ результат: g
4. представить a в виде $a = 2^k a_1$, где a_1 нечетное.
5. при четном k положить $s=1$, при нечетном положить $s=1$, если $n \equiv 1 \pmod 8$; положить $s=-1$, если $n \equiv 3 \pmod 8$

6. при a_1 результат: gs
7. если $n \equiv 3 \pmod{4}$ and $a_1 \equiv 3 \pmod{4}$, то $s = -s$
8. положить $a = n \bmod(a_1)$ $n = a_1$ $g = gs$ и вернуться на шаг 2

2.3 Алгоритм , реализующий тест Соловея - Штрассена

Вход. Нечетное целое число $n \geq 5$.

Выход. “Число n , вероятно, простое” или “Число n составное”.

1. Выбрать случайное целое число a , $2 \leq a \leq n-2$.
2. Вычислить $r = a^{(n+1)/2} \pmod{n}$
3. Если r не равен 1 и $n-1$ результат: “Число n составное”.
4. Вычислить символ Якоби $s = (a/n)$
5. Если $r \neq s \pmod{n}$ результат: “Число n составное”, иначе “Число n , вероятно, простое”.

2.4 Алгоритм , реализующий тест Миллера - Рабина

Вход. Нечетное целое число $n \geq 5$.

Выход. “Число n , вероятно, простое” или “Число n составное”.

1. представить $n-1$ в виде $n-1 = 2^s r$, где r нечетное
2. выбрать случайное целое число a , $2 \leq a \leq n-2$
3. вычислить $y = a^r \pmod{n}$
4. при $y \neq 1$ и $n-1$ выполнить следующее
 - 4.1. положить $j = 1$

4.2. если $j \leq s-1$ и y не равен $n-1$,то

4.2.1. положить $y = y^2 \pmod n$

4.2.2. при $y = 1$ результат: "Число n составное"

4.2.3. положить $j = j+1$

4.3. при y не равном $n-1$ результат: "Число n составное"

5. Результат: "Число n , вероятно, простое"

3 Выполнение работы

3.1 Реализация алгоритмов на языке Python

Тест Ферма

```
import random

n = int(input('Введите нечетное целое число n>=5: '))
a = random.randint(2, n - 2)
r = (a ** (n - 1)) % n
if r == 1:
    print(f'Число {n} ,вероятно, простое')
else:
    print(f'Число {n} - составное')
```

Символ Якоби

```
def jacobian_symbol(a, n):
    g = 1

    while True:
        if a == 0:
            return 0
        elif a == 1:
            return g
```



```

else:
    k, a1 = 0, a
    while a1 % 2 == 0:
        k += 1
        a1 //= 2
    if k % 2 == 0:
        s = 1
    else:
        if abs(n % 8) == 1:
            s = 1
        else:
            s = -1
    if a1 == 1:
        return g * s
    if n % 4 == 3 and a1 % 4 == 3:
        s *= -1
    a = n % a1
    n = a1
    g = g * s

if __name__ == "__main__":
    n = int(input("Enter an odd number n>= 3: "))
    a = int(input("Enter an integer 0<= a <= n: "))
    print(jacobian_symbol(a, n))

```

Алгоритм, реализующий тест Соловея - Штрассена

```

import random
from task_2 import jacobian_symbol

```

```

n = int(input('Введите нечетное целое число n>=5: '))
a = random.randint(2, n - 2)
r = a ** ((n - 1) / 2) % n
if r != 1 and r != n - 1:
    print(f'Число {n} - составное')
else:
    s = jacobian_symbol(a, n)
    if r % n == s:
        print(f'Число {n} составное')
    else:
        print(f'Число {n} ,вероятно, простое')

```

Алгоритм, реализующий тест Миллера - Рабина

```

import random

n = int(input('Введите нечетное целое число n>=5: '))
s, r = 0, n - 1
while r % 2 == 0:
    s += 1
    r //= 2
a = random.randint(2, n - 2)
y = (a ** r) % n
if y != 1 and y != n - 1:
    j = 1
    if j <= s - 1 and y != n - 1:
        y = (y ** 2) % n
        if y==1:
            print(f'Число {n} составное')

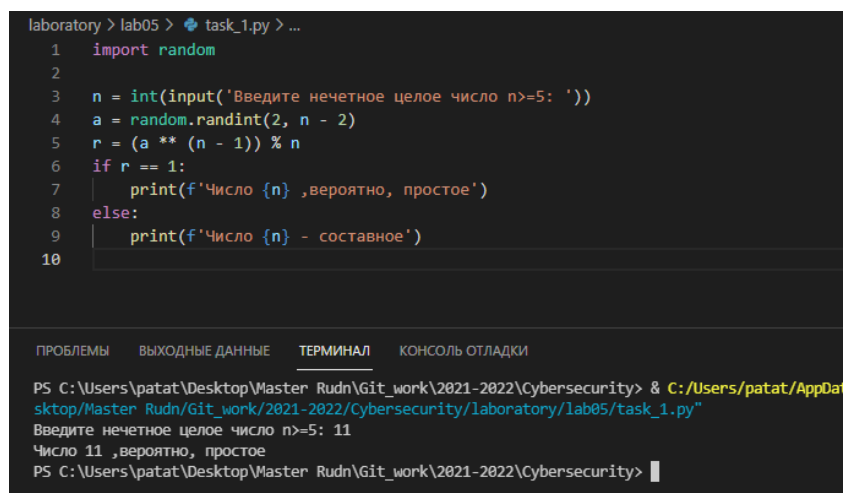
```

```

        j += 1
    if y != n - 1:
        print(f'Число {n} составное')
    else:
        print(f'Число {n} ,вероятно, простое')

```

3.2 Контрольный пример



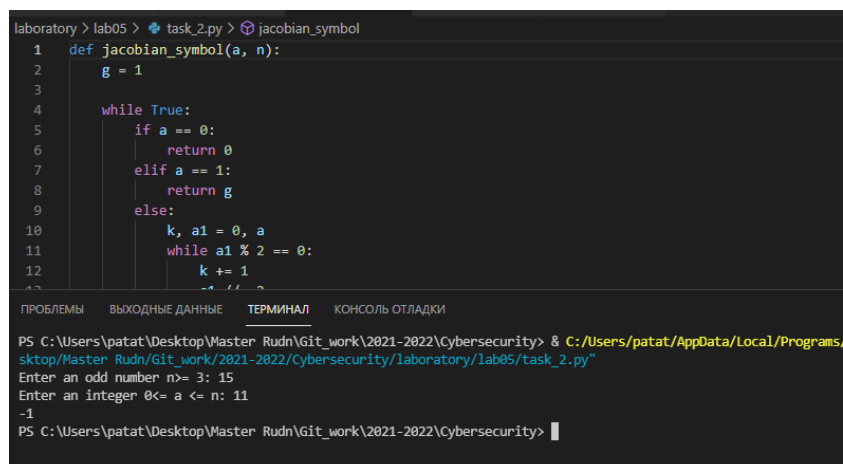
The screenshot shows a Python script named `task_1.py` in a dark-themed IDE. The script imports the `random` module, prompts the user for an odd integer `n` greater than or equal to 5, and then generates a random integer `a` between 2 and `n-2`. It calculates `r = (a ** (n - 1)) % n` and checks if `r` is equal to 1. If true, it prints that the number is probably prime; otherwise, it prints that the number is composite. The terminal window below the script shows the execution path and the user inputting 11, resulting in the output: "Число 11 ,вероятно, простое".

```

laboratory > lab05 > task_1.py > ...
1  import random
2
3  n = int(input('Введите нечетное целое число n>=5: '))
4  a = random.randint(2, n - 2)
5  r = (a ** (n - 1)) % n
6  if r == 1:
7      print(f'Число {n} ,вероятно, простое')
8  else:
9      print(f'Число {n} - составное')
10
ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  ТЕРМИНАЛ  КОНСОЛЬ ОТЛАДКИ
PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity> & C:/Users/patat/AppData/Local/Programs/Python/Python39-6/Python.exe C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity\laboratory\lab05\task_1.py
Введите нечетное целое число n>=5: 11
Число 11 ,вероятно, простое
PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity>

```

Figure 3.1: Тест Ферма



The screenshot shows a Python script named `task_2.py` in a dark-themed IDE. The script defines a function `jacobian_symbol(a, n)`. It handles the base cases where `a` is 0 or 1, and then enters a loop where it repeatedly updates `a` and `n` based on their parity and modulo values. The terminal window shows the execution path and the user inputting 3 and 15, resulting in the output: "-1".

```

laboratory > lab05 > task_2.py > jacobian_symbol
1  def jacobian_symbol(a, n):
2      g = 1
3
4      while True:
5          if a == 0:
6              return 0
7          elif a == 1:
8              return g
9          else:
10             k, a1 = 0, a
11             while a1 % 2 == 0:
12                 k += 1
13                 a1 //= 2
14             if a1 % 4 == 3 and n % 4 == 3:
15                 g = -g
16             a, n = a1, n
17             if n == 1:
18                 return g
19
20
ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  ТЕРМИНАЛ  КОНСОЛЬ ОТЛАДКИ
PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity> & C:/Users/patat/AppData/Local/Programs/Python/Python39-6/Python.exe C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity\laboratory\lab05\task_2.py
Enter an odd number n>= 3: 15
Enter an integer 0<= a <= n: 11
-1
PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity>

```

Figure 3.2: Символ Якоби

```
laboratory > lab05 > task_3.py > ...
1 import random
2 from task_2 import jacobian_symbol
3
4 n = int(input('Введите нечетное целое число n>=5: '))
5 a = random.randint(2, n - 2)
6 r = a ** ((n - 1) / 2) % n
7 if r != 1 and r != n - 1:
8     print(f'Число {n} - составное')
9 else:
10     s = jacobian_symbol(a, n)
11     if r % n == s:
12         print(f'Число {n} составное')
13     ...

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ

PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity> & C:\Users\patat\AppData\Local\Programs\Python\
sktop\Master Rudn\Git_work\2021-2022\Cybersecurity\laboratory\lab05\task_3.py*
Введите нечетное целое число n>=5: 17
Число 17 , вероятно, простое
PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity> |
```

Figure 3.3: Алгоритм , реализующий тест Соловея - Штрассена

```

laboratory > lab05 > task_4.py > ...
1  import random
2
3  n = int(input('Введите нечетное целое число n>=5: '))
4  s, r = 0, n - 1
5  while r % 2 == 0:
6      s += 1
7      r //= 2
8  a = random.randint(2, n - 2)
9  y = (a ** r) % n
10 if y != 1 and y != n - 1:
11     j = 1
12     if j <= s - 1 and y != n - 1:
13         ...

```

ПРОБЛЕМЫ

ВЫХОДНЫЕ ДАННЫЕ

ТЕРМИНАЛ

КОНСОЛЬ ОТЛАДКИ

```

PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity> & C:/Users/patat/AppData\
sktop\Master Rudn\Git_work\2021-2022\Cybersecurity\laboratory\lab05\task_4.py"
Введите нечетное целое число n>=5: 21
Число 21 составное
PS C:\Users\patat\Desktop\Master Rudn\Git_work\2021-2022\Cybersecurity>

```

Figure 3.4: Алгоритм , реализующий тест Миллера - Рабина

4 Выводы

Мной были изучены вероятностные алгоритмы проверки чисел на простоту.

Список литературы

1. Инструкция к лабораторной работе №5