Primality Tests

Cryptography - Applied Number Theory

CSE 496
Final Presentation

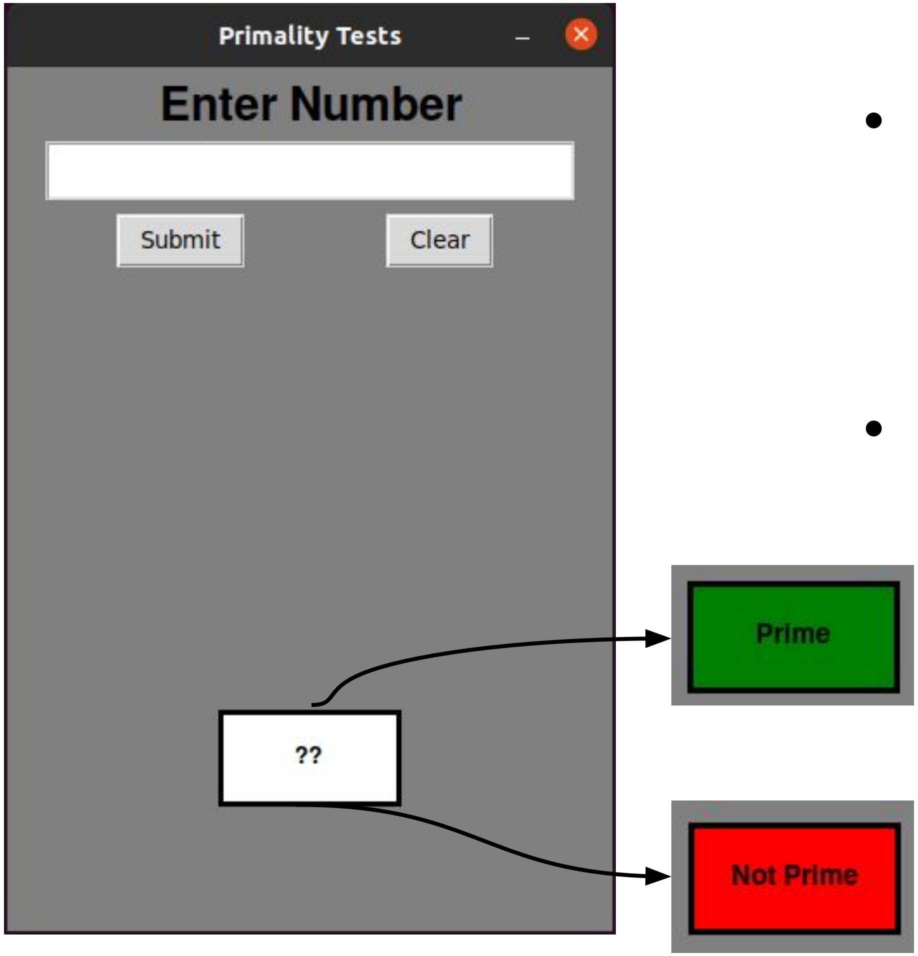
Baran Solmaz

Project Advisor: Dr. Tülay AYYILDIZ AKOĞLU
June 2023

Contents

- Project Summary
- Updates
- GUI
- Solutions and Results
- Room for Improvement
- Resources

Project Summary

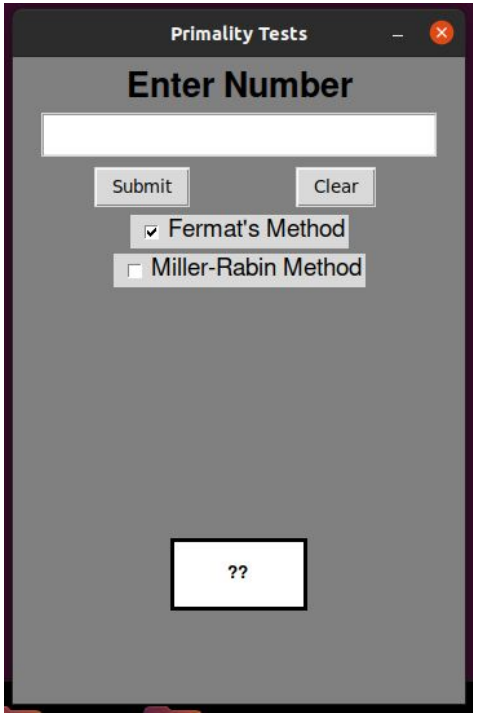


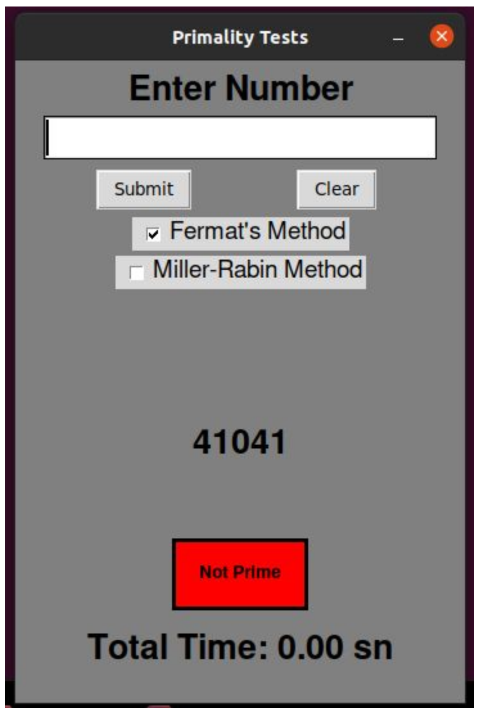
- Project Description:
  - Takes natural number,
  - Checks whether it is prime or not.
- Aim:
  - Handling Pseudo-primes.

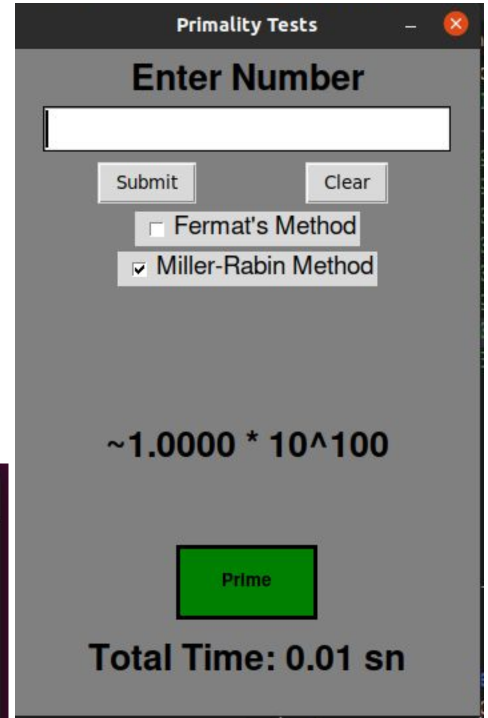
Updates

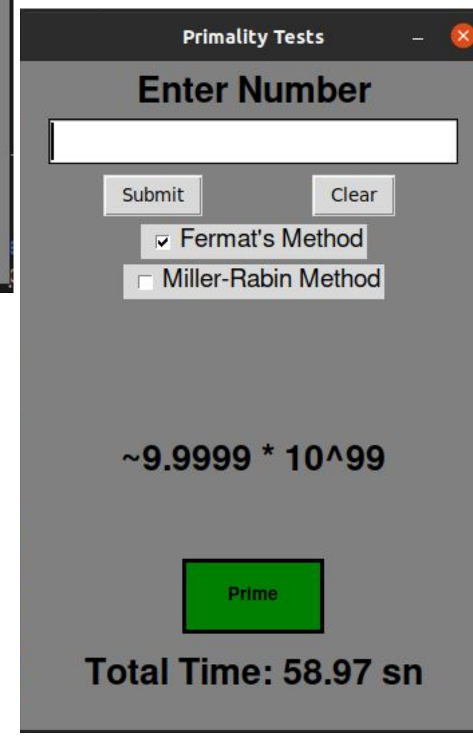
- GUI
- Test Implementations
  - Fermat’s Test
  - Miller-Rabin Primality Test
- Solution Approaches
  - Carmichael Control
  - Strong Pseudoprime Control
- Solution Implementations
  - Carmichael Control
  - Strong Pseudoprime Control - AKS Test

GUI


Primality Tests
Enter Number
Submit
Clear
Fermat's Method
Miller-Rabin Method
??


Primality Tests
Enter Number
Submit
Clear
Fermat's Method
Miller-Rabin Method
41041
Not Prime
Total Time: 0.00 sn


Primality Tests
Enter Number
Submit
Clear
Fermat's Method
Miller-Rabin Method
~1.0000 * 10^100
Prime
Total Time: 0.01 sn


Primality Tests
Enter Number
Submit
Clear
Fermat's Method
Miller-Rabin Method
~9.9999 * 10^99
Prime
Total Time: 58.97 sn

Carmichael Control - I

```python
def isPrime_fermats_method(number, k):
    # Corner cases
    if number == 1 or number == 4:
        return False
    elif number == 2 or number == 3:
        return True
    else:  # Try k times
        for i in range(k):
            # Pick a random number in [2..number-2]
            rand = random.randint(2, number - 2)

            # Fermat's little theorem
            if power(rand, number - 1, number) != 1:
                return False

    if isCarmichaelNumber(number):
        return False

    return True
```

Algorithm 3 isCarmichaelNumber

Require: A number n

Ensure: **True** if n is a Carmichael number, **False** otherwise.

```
digit ← len(str(n))
if digit ≤ 13 then
    return not isPrimeOptimizedBasicMethod(n)[4]
else if digit ≤ 25 then
    return checkCarmichaelLess25digit(n)[5]
else if digit ≤ 50 then
    return checkCarmichaelLess50digit(n)
else if digit ≤ 100 then
    return checkCarmichaelLess100digit(n)
else
    return checkCarmichaelGreater100digit(n)
end if
```

Carmichael Control - II

Algorithm 5 checkCarmichaelLess25digit

Require: A number n

Ensure: **True** if n is a Carmichael number, **False** otherwise.

mainRoot $\leftarrow \lfloor\sqrt{n}\rfloor + 1$
root4 $\leftarrow \lfloor\sqrt{\text{mainRoot}}\rfloor + 1$
root8 $\leftarrow \lfloor\sqrt{\text{root4}}\rfloor + 1$
randStart $\leftarrow$ random.randint$(10, \text{root8})$
randEnd8 $\leftarrow$ random.randint$(1, \text{root8})$
start $\leftarrow$ mainRoot $-$ (root4 $+$ root8) $\times$ randStart
interval $\leftarrow$ mainRoot $-$ (root8 $\times$ root4) $-$ ((root8 $+$ root4) $\times$ randEnd8)
while $\log_{10}(\text{interval}) + 1 > 9$ **do**
    interval $\leftarrow$ interval $\div$ 10
end while
end $\leftarrow$ start $+$ interval
threadNum $\leftarrow$ 10
extras $\leftarrow$ interval $\bmod$ threadNum
threadInterval $\leftarrow$ interval $\div$ threadNum
threadList $\leftarrow$ []
for $i \leftarrow 0$ to threadNum $- 1$ **do**
    threadList.append(ThreadControl$\boxed{6}$, args(i, n,start,start+threadInterval)
    start $\leftarrow$ start $+$ threadInterval
    if $i =$ threadNum $- 2$ **then**
        threadInterval $\leftarrow$ threadInterval $+$ extras
    end if
end for
for $i \leftarrow 0$ to threadNum $- 1$ **do**
    threadList$[i]$.start()
    sleep(0.01) ▷ 0.01 sec
end for
for $i \leftarrow 0$ to threadNum $- 1$ **do**
    threadList$[i]$.join()
end for
return getIsCarm()

Algorithm 6 Thread Control

Require: A possible prime number n, a start value $start$, and an end value end

Ensure: None

for $k \leftarrow$ start to end with step 2 **do**
    if $k \mod 3 = 0$ or $k \mod 5 = 0$ or $k \mod 7 = 0$ **then** ▷ To decrease total number and fasten thread
        continue
    end if
    if getIsCarm() **then** ▷ Returns global variable that holds boolean value
        break
    end if
    if math.gcd(n, k) $\neq 1$ **then**
        setIsCarm(True) ▷ Sets that global variable as True
        break
    end if
end for
return

Carmichael Control - III

Test Results:

Digit Amount	Thread Amount	Added Total Delay(sn)	Computation Time(sn)
<14	1	0	0.1
14 - 25	10	0.1	0.45 - 6.9
26 - 50	20	4	7.2 - 21.0
51 - 100	40	20	35.8 - 63.0
100 < (200)*	50	23	62 - (83.0)*

“ **The greatest number that is tested has 200 digit.*
The results are from prime numbers, so worst case. ”

Strong Pseudoprime Control

- Strong Pseudoprime Control
  - AKS Test:

$$(X + a)^n \equiv X^n + a \,(mod\, n)$$

Algorithm 7 isPseudoPrime

Require: A number n

Ensure: **True** if n is a Strong Pseudo Prime, **False** otherwise.

$digit \leftarrow$ len(str(n))
if digit ≤ 13 **then**
    return not isPrimeOptimizedBasicMethod(n)$\boxed{4}$
else
    return not aKSTest(n)$\boxed{8}$
end if

Algorithm 8 AKS Primality Test

Require: A number n

Ensure: **True** if n passes the AKS primality test, **False** otherwise.

mainRoot $\leftarrow \lfloor\sqrt{\text{n}}\rfloor + 1$
root4 $\leftarrow \lfloor\sqrt{\text{mainRoot}}\rfloor + 1$
randX $\leftarrow$ random.randint(10, mainRoot)
randY $\leftarrow$ random.randint(10, root4)
$a \leftarrow 2^{\text{randY}}$
left $\leftarrow$ power(randX $+ a$, n, n)
right $\leftarrow$ (power(randX, n, n) $+ a$)(mod n)
return (left $\equiv$ right)

Room for Improvement

- Supercomputers,
- Reducing the range,
- Selecting special starting point.

References

- Carmichael Numbers - GFG
- Strong Pseudoprimes - OEIS
- Strong Pseudoprimes -Mathworld Wolfram
- Properties Of Strong Pseudoprimes On Base b - K. Parhi & P. Kumari
- AKS Test - Primes is in P - Agrawal,Kayal,Saxena
- AKS - Mathworld Wolfram
- M. Sipser, Introduction to the Theory of Computation. Thomson Course Tech., 2005.
- J. Von Zur Gathen, Modern Computer Algebra. Cambridge Uni. Press, 2013