Primality Tests

CSE 496
Preliminary Presentation

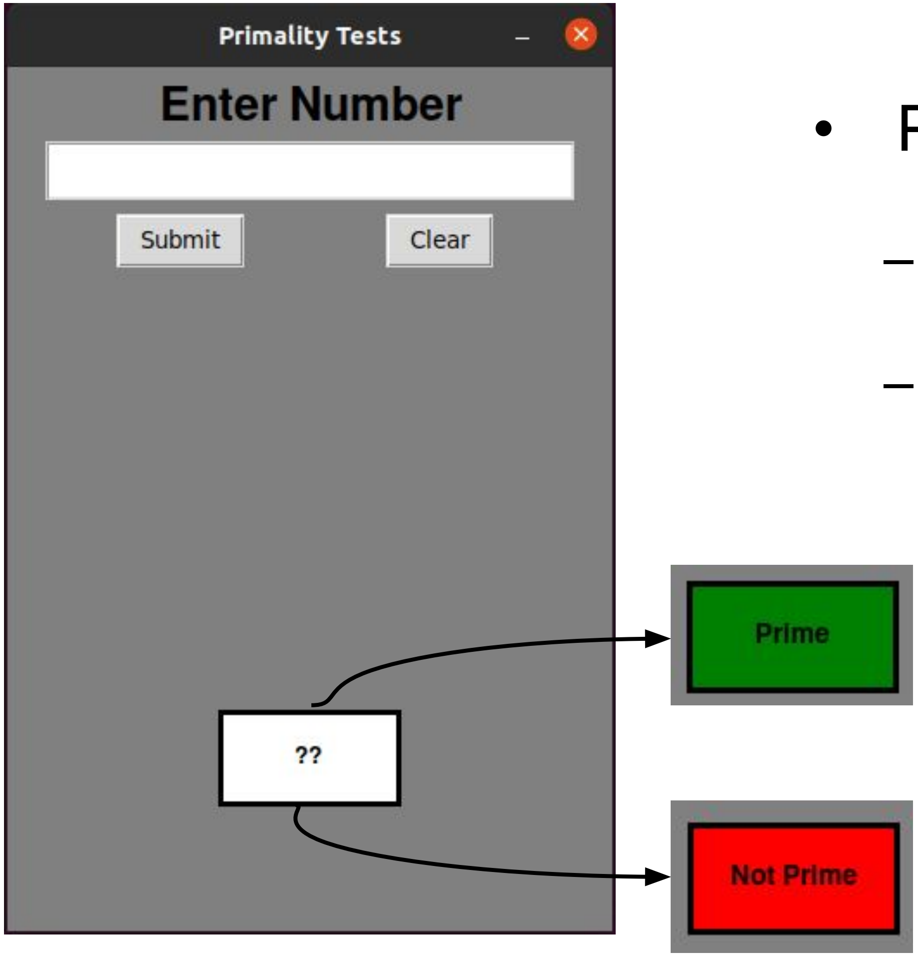
Baran SOLMAZ

Project Advisor: Dr. Tülay AYYILDIZ AKOĞLU
Apr 2023

Contents

- Description of The Project
- Definitions
- Fermat's Test
- Miller-Rabin Test
- Success Criteria
- References

Description of The Project



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

Definitions

- Prime Numbers:
  – Natural numbers that are divisible by only { 1 } and the number itself.
- Composite Numbers:
  – All natural numbers except Primes and { 1 }.
- Pseudoprimes:
  – Composite numbers that can pass probabilistic primality tests.
- Carmichael Numbers:
  – Pseudoprimes that can pass Fermat's Primality Test.
- Strong Pseudoprimes:
  – Composite numbers that can pass Miller-Rabin Primality Test.

Fermat's Test

$$a^{p-1} \equiv 1(mod\, p)$$

Since 5 is prime,
$2^4 \equiv 1 \pmod 5$,
$3^4 \equiv 1 \pmod 5$,
$4^4 \equiv 1 \pmod 5$.

```
function isPrime_fermat(n, k):
    if n <= 1 or n == 4:
        return False
    elif n <= 3:
        return True
    else:
        for i in range(k):
            a = random.randint(2, n - 2)
            if power(a, n - 1, n) != 1:
                return False
    return True
```

- Carmichael Numbers:
  - 561, 1105, 1729, 2465, 2821, 6601,…(OEIS A002997)

$$2^{560} \equiv 1\,(mod\, 561)$$
$$447^{560} \equiv 375\,(mod\, 561)$$

Miller-Rabin Test

- Probabilistic Algorithm
- Strong Pseudoprimes

$$d \cdot 2^r = n - 1$$

$$a^d \,(mod\, n) \equiv 1 \, or \, (n-1)$$

$$x^2 \equiv n - 1 (mod\, n)$$

```
function is_prime(n, k):
    '''
    Handle corner case: n<=4 & n%2==0
    '''
    r, d = 0, n - 1
    while d % 2 == 0:
        r += 1
        d //= 2
    for i in range(k):
        a = random.randint(2, n - 2)
        x = power(a, d, n)
        if x == 1 or x == n - 1:
            continue
        for j in range(r - 1):
            x = power(x, 2, n)
            if x == n - 1:
                break
        else:
            return False
    return True
```

Success Criteria

- Handling pseudoprimes:
  - Carmichael Numbers.
  - Strong Pseudoprimes.
- Optimizing Algorithm:
  - If possible.

References

- Primality Tests
- Fermat’s Test
- Carmichael Numbers
- Miller-Rabin Test
- Joachim von zur Gathen,Modern Computer Algebra Book,3rd Edition,Chapter 18,Primality Testing
- Michael Sipser,Introduction to the Theory of Computation,2nd Edition,Chapter 10, Advanced Topics in Complexity Theory,10.2 Probabilistic Algorithms/Primality