Miller-Rabin Primality Test: Implementing and understanding its use in probabilistic primality testing

Ahmed Amir - Samuel Marzouk - Marwan Mohamed - Amr Khaled
Mentor: Dr.Abdallah Awad Aboutahoun
Zewail University - EGYPT

Abstract

The Miller-Rabin Primality Test is a widely used algorithm for determining whether a given integer is prime or composite. It is a probabilistic algorithm, meaning that it can sometimes incorrectly identify a composite number as prime, but the probability of this happening can be made arbitrarily small by repeating the test multiple times.

This report aims to provide a detailed overview of the Miller-Rabin Primality Test, including its mathematical foundations, implementation, and practical applications. It will begin by introducing the concept of primality testing and the challenges associated with deterministic prime number testing algorithms. It will then delve into the theoretical underpinnings of the Miller-Rabin test, explaining the key ideas of the Fermat's Little Theorem and the Miller-Rabin Theorem, which form the basis of the algorithm.

The report will then present a step-by-step implementation of the Miller-Rabin Primality Test, highlighting the key steps and optimizations that can be made to improve its efficiency. It will also discuss the concept of the Miller-Rabin witness and explain how the number of witnesses used can affect the reliability of the test.

Finally, the report will explore the practical applications of the Miller-Rabin Primality Test, particularly in the field of cryptography, where it is used to generate and verify large prime numbers for use in public-key cryptography algorithms, such as RSA and Diffie-Hellman key exchange. It will also discuss the limitations of the Miller-Rabin test and situations where alternative prime testing algorithms may be more appropriate

1 Introduction

The Miller-Rabin primality test mainly depends on features of modular exponentiation and basic ideas from discrete mathematics, especially modular arithmetic.

Modular Arithmetic: A lot of modular arithmetic procedures, including computing $a^d \mod n$, are used in the Miller-Rabin test. The test's mathematical basis comes from modular arithmetic, a fundamental subject in discrete mathematics that examines the characteristics of numbers as they undergo the modulo operation.

Fermat's Little Theorem: This test is a better version of the Fermat primality test, which was susceptible to manipulation by Carmichael numbers. A fundamental result in number theory, Fermat's Little Theorem asserts that for every a coprime to n, $a^{(n-1)} \equiv 1 \pmod{n}$ if n is prime. This attribute is used by the Miller-Rabin test to more accurately discriminate between prime and composite numbers.

Discrete Logarithms: The formula $x = a^{2^i \cdot d} \mod n$ calculates the series of values. This has to do with the idea of discrete logarithms, which is crucial to understanding both discrete mathematics and encryption.

Computational Complexity: The Miller-Rabin test's effectiveness is strongly related to findings in computational complexity theory, a significant field of discrete mathematics, due to its capacity to test huge integers for primality rapidly. $O(k \cdot \log^3 n)$ is the test's time complexity, where k is the total number of witness numbers employed.

Data is growing at an astoundingly rapid rate, and better information security is required to protect increasing quantities of data. Improved data protection requires more sophisticated cryptographic methods. Improved cryptography necessitates the use of larger semiprimes that are extremely difficult to factor, and thus requires the verification of primality of larger primes. Therefore, faster and more efficient primality tests are key to better information security.

2 The Miller-Rabin Primality Test

The Miller-Rabin Primality Test is an extension of the Fermat Primality Test. The test works as follows: Suppose we have an odd integer n, such that $n = 1 + d \cdot 2^e$ and d is odd.

We choose a positive integer $a < n$. If either $a^d \equiv 1 \pmod{n}$, or $a^{2^r \cdot d} \equiv -1 \pmod{n}$ for some $r < e$, then n is probably prime. Else, n is composite. If n is probably prime, then either n is prime, or n is composite, in which case we say that a is a nonwitness to n and that n is a strong pseudoprime to the base a. We say that a is a witness to n if $a < n$, and a is not a nonwitness [2, 3].

Example 1.1

Suppose $n = 65$. If we consider $a = 8$, we notice that $n = 1 + 1 \cdot 2^6$,

$$8^1 \equiv 8 \not\equiv 1,$$

but,

$$8^{2^5} = 64 \equiv -1.$$

Thus, either 65 is prime, or 65 is a composite and 8 is a nonwitness. Of course 65 is not prime, but just to check, we consider $a = 2$. Clearly,

$$2^1 \not\equiv 1,$$

$$2^{2^0} = 2 \not\equiv -1,$$
$$2^{2^1} = 4 \not\equiv -1,$$
$$2^{2^2} = 16 \not\equiv -1,$$
$$2^{2^3} = 61 \not\equiv -1,$$
$$2^{4-1} = 61 \not\equiv -1.$$

Since 65 fails the Miller-Rabin Primality Test in base 2, we know that 65 is composite. We also know $a = 2$ is witness to 65, but 8 is a nonwitness to 65.

The Miller-Rabin Primality Test is significantly more accurate than the Fermat Primality Test. There exist an infinite number of composite integers known as Carmichael numbers, which satisfy the property that $\forall n$, where n is a Carmichael number, if $(a, n) = 1$, then

$$a^{n-1} \equiv 1 (modn)$$

[4]. However, Michael O. Rabin proved that for any composite odd integer n, the number of nonwitnesses of n is at most

$$\frac{n}{4},$$

and can even be reduced to

$$\frac{\varphi(n)}{4}$$

if $n > 25$.

To demonstrate the improved effectiveness of the Miller-Rabin Primality Test, we check whether 91 is prime or composite.

Example 1.2

We test 91 with the base of 3. If we use the Fermat Primality Test, we get 390 1 (mod 91). If we use the Miller-Rabin Primality Test, since 91 = 2 · 45 + 1, and since 345 27 (mod 91), it is clear that 3 is a witness to 91 for the Miller-Rabin Primality Test even though 3 is a false witness for the Fermat Primality Test.

3 Miller-Rabin Method with Python

The Miller-Rabin Primality Test is a probabilistic algorithm used to determine whether a given number is prime. It is an extension of the Fermat Primality Test and provides more accurate results by reducing the number of false positives (composite numbers identified as primes).

3.1 Algorithm Description

Suppose we have an odd integer n that we want to test for primality. The Miller-Rabin Primality Test works as follows:

1. Write $n - 1$ as $d \cdot 2^r$ where d is an odd integer. 2. Choose a random integer a such that $1 < a < n - 1$. 3. Compute $x = a^d \mod n$. 4. If $x = 1$ or $x = n - 1$, then n is probably prime. 5. Repeat the following for $r - 1$ times: - Compute $x = x^2 \mod n$. - If $x = n - 1$, then n is probably prime. 6. If none of the above conditions hold, then n is composite.

If the test declares that n is probably prime, it means that either n is prime or it is a composite number that passed the test (a strong pseudoprime). By repeating the test with different values of a, we can reduce the probability of incorrectly identifying a composite number as prime.

4 Example

Let's apply the Miller-Rabin Primality Test to check if 65 is prime.

Step-by-Step Example

Suppose we want to test if $n = 65$.

1. Write $64 = 65 - 1 = d \cdot 2^r = 1 \cdot 2^6$, so $d = 1$ and $r = 6$. 2. Choose a random integer, say $a = 8$. 3. Compute:

$$x = 8^1 \mod 65 = 8$$

4. Since $x \neq 1$ and $x \neq 64(n-1)$, proceed to step 5. 5. Repeat for $r-1 = 5$: - Compute:

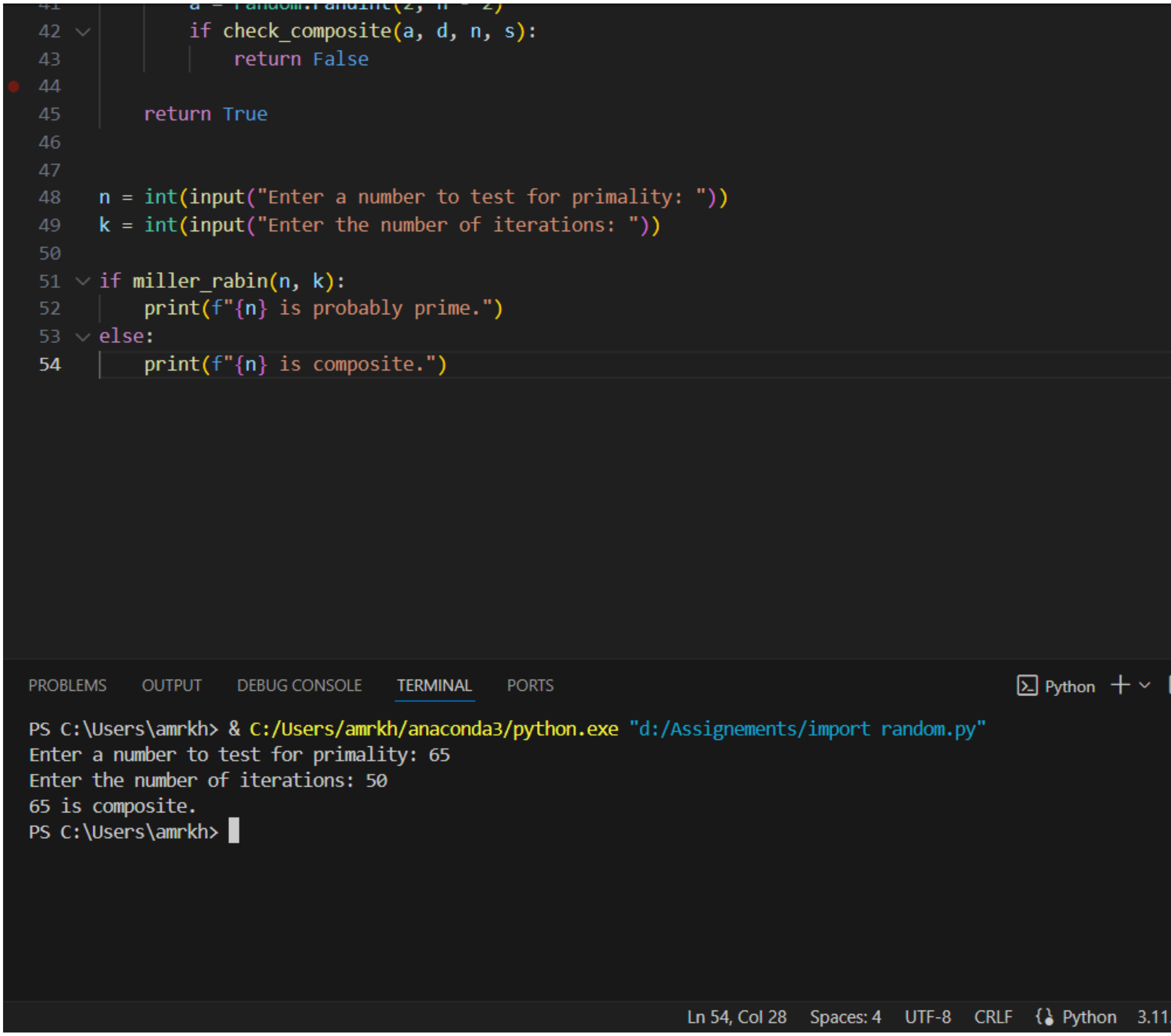
$$x = 8^2 \mod 65 = 64$$

- Since $x = n - 1 = 64$, declare that 65 is probably prime.

However, since we know that 65 is not actually prime, this indicates that our choice of base was not sufficient to detect its compositeness in this single iteration.

To increase accuracy, repeat with different bases.

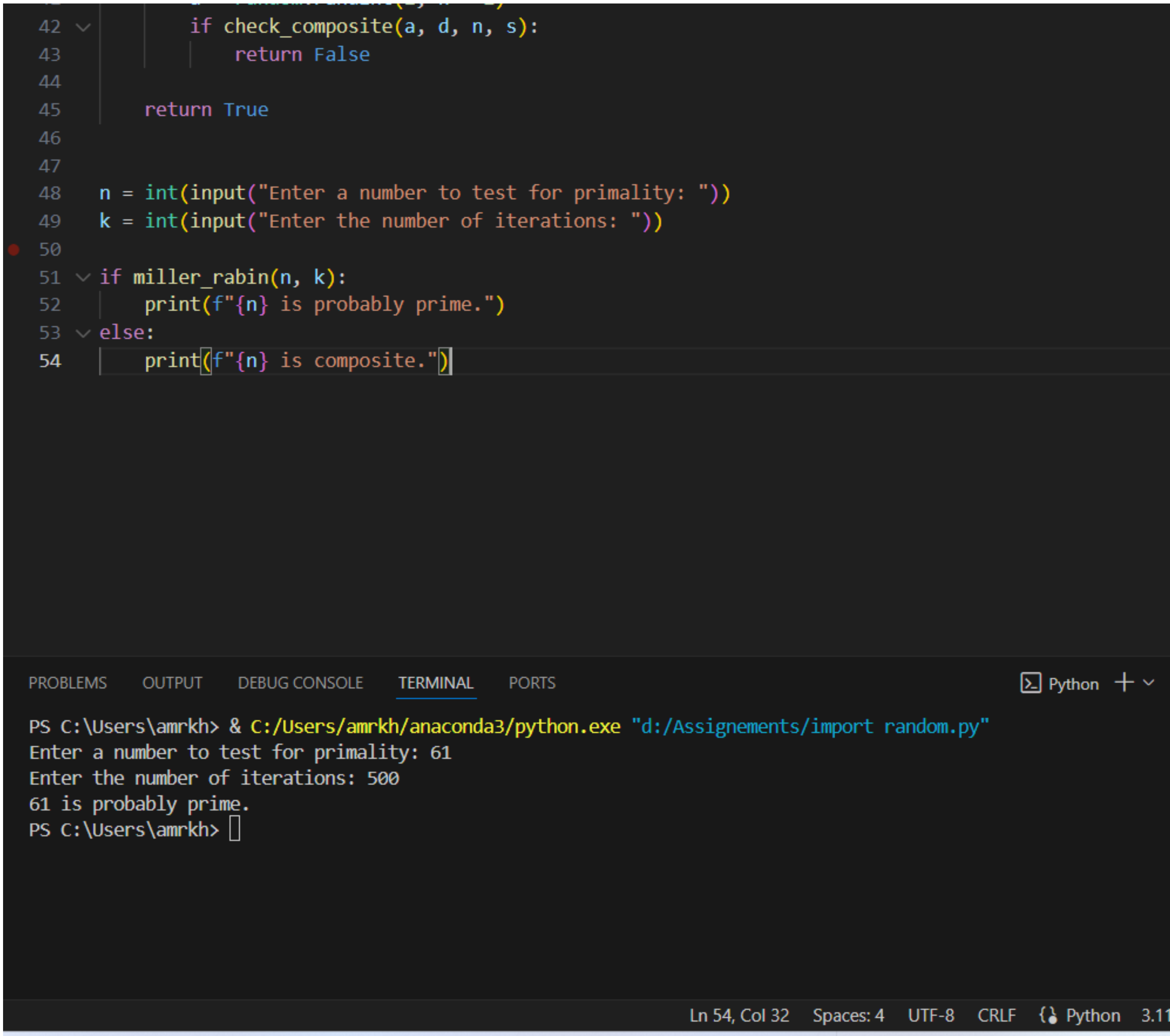
'



```
        a = random.randint(2, n - 2)
        if check_composite(a, d, n, s):
            return False

    return True


n = int(input("Enter a number to test for primality: "))
k = int(input("Enter the number of iterations: "))

if miller_rabin(n, k):
    print(f"{n} is probably prime.")
else:
    print(f"{n} is composite.")
```

```
PS C:\Users\amrkh> & C:/Users/amrkh/anaconda3/python.exe "d:/Assignements/import random.py"
Enter a number to test for primality: 65
Enter the number of iterations: 50
65 is composite.
PS C:\Users\amrkh> 
```

Figure 1: Primality Test for n=65(Composite Number)



```
        if check_composite(a, d, n, s):
            return False

    return True


n = int(input("Enter a number to test for primality: "))
k = int(input("Enter the number of iterations: "))

if miller_rabin(n, k):
    print(f"{n} is probably prime.")
else:
    print(f"{n} is composite.")
```

```
PS C:\Users\amrkh> & C:/Users/amrkh/anaconda3/python.exe "d:/Assignements/import random.py"
Enter a number to test for primality: 61
Enter the number of iterations: 500
61 is probably prime.
PS C:\Users\amrkh> 
```

Figure 2: Primality Test for n=61 (Prime Number)

5 Real Life Applications

Data is growing at an astoundingly rapid rate, and better information security is required to protect increasing quantities of data. Improved data protection requires more sophisticated cryptographic methods. Improved cryptography necessitates the use of larger semiprimes that are extremely difficult to factor, and thus requires the verification of primality of larger primes. Therefore, faster and more efficient primality tests are key to better information security The Rabin-Miller primality test has several applications. It is used in public key cryptographic systems such as RSA and SSC to determine whether a given integer is prime or composite [1]. The test is known for its efficiency and reliability in generating large prime numbers, which are essential for ensuring high security in cryptographic algorithms [2]. The test's ability to determine prime integers is based on the difference in the number of primality witnesses for composite and prime integers [3]. By studying the test's effectiveness, researchers have derived new formulas for the power of primality witnesses and have shown that the average probability of errors decreases as the length of tested integers increases [4]. Additionally, modifications and optimizations of the Miller-Rabin algorithm have been implemented to improve its speed and reliability in generating keys for encryption algorithms like RSA and DSA

6 Concolusion

article amsmath

The Miller-Rabin Primality Test stands as a crucial tool in the realm of computational number theory and cryptography. Its probabilistic nature, while introducing a slight risk of misidentifying composite numbers as prime, provides a balance between efficiency and accuracy that is particularly valuable in practical applications. By leveraging the principles of Fermat's Little Theorem and the Miller-Rabin Theorem, this algorithm offers a robust method for primality testing that is both theoretically sound and practically efficient.

Throughout this report, we have explored the mathematical foundations that underpin the Miller-Rabin Primality Test, elucidating the roles of witness selection and repetition in minimizing error probability. The step-by-step implementation outlined herein emphasizes key optimizations that enhance the test's performance, making it a preferred choice in scenarios demanding rapid and reliable prime number verification.

The practical significance of the Miller-Rabin test is underscored by its widespread use in cryptographic protocols, such as RSA and Diffie-Hellman key exchange, where the generation and validation of large prime numbers are paramount. Despite its probabilistic nature, the test's ability to provide near-certain primality verification through repeated trials ensures its suitability for high-stakes applications.

However, it is important to recognize the limitations of the Miller-Rabin Primality Test. In contexts requiring absolute certainty, deterministic algorithms or additional probabilistic tests may be necessary. As such, the Miller-Rabin test is best viewed as a component within a broader toolkit for prime testing, complemented by other methods when appropriate.

the Miller-Rabin Primality Test exemplifies the power and utility of probabilistic algorithms in modern computational tasks. Its blend of mathematical elegance, implementation simplicity, and practical efficacy solidifies its role as a cornerstone in the study and application of prime number theory. As advancements in computational techniques continue, the Miller-Rabin test will remain an essential algorithm, continually adapting to meet the evolving demands of cryptographic security and number theory research.

'

References

[1] Agrawal, M., Kayal, N., and Saxena, N. (2004). PRIMES is in P. *Annals of Mathematics*, *160*(2), 781-793.

[2] Miller, G. (1976). Riemann's Hypothesis and Tests for Primality. *Journal of Computer and System Sciences 13*(3): 300–317.

[3] Rabin, M. (1980). Probabilistic algorithm for testing primality. *Journal of Number Theory*, *12*(1), 128-138.

[4] Alford, W. R., Granville, A., and Pomerance, C. (1994). There are Infinitely Many Carmichael Numbers. *Annals of Mathematics*, *139*, 703–722.

[5] Menezes, A. J., van Oorschot, P. C., Vanstone, S. A. (1996). Handbook of Applied Cryptography. CRC Press.