Public Key Cryptography – Lab3

The Miller-Rabin Test

Input: $n \in \mathbb{N}, n \geq 3$ odd, $k \in \mathbb{N}^*$
Output: $P(n$ is prime$)$ – the probability that n is prime

```
Algorithm miller_rabin(n, k)
    probability := 1 − 1/4^k
    if n = 2 then
        return 1.0
    # Step 0. Write n − 1 = 2^s t, where t is odd
    s := 0
    while t is even do
        s := s + 1
        t := (n−1)/2^s

    # Step 1: Choose (randomly) 1 < a < n
    # Step 2: Compute the following sequence (modulo n): a^t, a^(2t), a^(2^2 t), …, a^(2^s t)
    # Step 3: If either the first number in the sequence is 1 or one gets the value
1 and its previous number -1, then n is possible to be prime and we repeat the steps
1-3 at most k times
    while k > 0 do
        a := random number between from {2, …, n − 1}
        for i = 1,s do
            e := repeated_squaring_modular_exponentiation(a, 2^i t, n)
            if e = 1 then
                k := k - 1
                goto Step 1
        # Step 4: The algorithm stops and n is composite
            return 0.0
    return probability
```

Input: $x \in \mathbb{N}, y \in \mathbb{N}, n \in \mathbb{N}$
Output: $x^y (\bmod n)$

```
Algorithm repeated_squaring_modular_exponentiation(x, y, n)
    result := 1
    current := x mod n
    while y > 0 do
        if y is odd then
            result := (result * current) mod n
        current := (current * current) mod n
        y := ⌊y/2⌋
    return result
```

Vlad Cîncean / 933
cvie1883@scs.ubbcluj.ro