Fermat and Rabin Miller

Euclid Theorem Says : (Euler totient cp algo (down))

$$a^{\phi(m)-1} \% \; m = 1 \quad -(1)$$

where $\phi(m)$ is Euler totient of m.

and a and m are co prime.

Fermat's Little Theorem – is a special case of above theorem

As we know, if p is prime

$$\phi(P) = P-1$$

Keeping this in eqn (1) we get

$$a^{(P-1)} \% \; p = 1$$

$$\equiv \boxed{a^{P-1} = (1 \bmod P)}$$

Note

* We can use the above relation to check, if the number is prime, by putting 'n' in place of p, and see if relation holds.

* However, it is found that there are some composite numbers that also satisfy the above relation.

* Fermat's Little Theorem is only a probabilistic test of primality and not a deterministic one.

and hence we move to Rabin Miller, which is implemented with little manipulation in fermat's theo

Code (Fermat)

```
bool probablyPrimeFermat(int n, int iter=5) {
    if (n < 4)
        return n == 2 || n == 3;

    for (int i = 0; i < iter; i++) {
        int a = 2 + rand() % (n - 3);
        if (binpower(a, n - 1, n) != 1)
            return false;
    }
    return true;
}
```

self written Power fn:

Rabin Miller:

* Task = check if n is prime?

we know that except 2 every prime is odd; Let n is odd

so $(n-1) = (2^s) \cdot d \leftarrow$ (hence remaining odd no).

$n-1$: even; 2^s: (all powers of 2)

$\Rightarrow$ if n is prime then

$a^{n-1} \% n = 1$ (Fermat)

$a^{2^s \cdot d - 1} \% n = 1$

$\left(a^{2^s \cdot d} - (1)^2\right) \% n = 0$

$(a^2 - b^2) = (a+b)(a-b)$

$\left(a^{2^{s-1} \cdot d} + 1\right) \left(a^{2^{s-1} \cdot d} - 1\right) = 0 \bmod n$

same with this

$\left(a^{2^{s-1} \cdot d} + 1\right) \left(a^{2^{s-2} \cdot d} + 1\right) \left(a^{2^{s-2} \cdot d} - 1\right) = \sim$

$\left(a^{2^{s-1} \cdot d} + 1\right) \left(a^{2^{s-2} \cdot d} + 1\right) \cdots \left(a^{d} + 1\right)\left(a^{d} - 1\right) = 0 \bmod n$

So we need to check

if $a^d \% n = 1$

or $a^{2^r \cdot d} \% n = -1$

i.e $(n+1)$

* It is experimentally found that if we check for first 12 primes as values of a, then Rabin Miller is determ^tic.

Power fn

```
u64 binpower(u64 base, u64 e, u64 mod) {
    u64 result = 1;
    base %= mod;
    while (e) {
        if (e & 1)
            result = (u128)result * base % mod;
        base = (u128)base * base % mod;
        e >>= 1;
    }
    return result;
}
```

To check if n is composite:

```
bool check_composite(u64 n, u64 a, u64 d, int s) {
    u64 x = binpower(a, d, n);
    if (x == 1 || x == n - 1)
        return false;
    for (int r = 1; r < s; r++) {
        x = (u128)x * x % n;
        if (x == n - 1)
            return false;
    }
    return true;
};
```

To check for '12' different bases of a.

```
bool MillerRabin(u64 n) { // returns true if n is prime, else returns false.
    if (n < 2)
        return false;

    int r = 0;
    u64 d = n - 1;
    while ((d & 1) == 0) {
        d >>= 1;
        r++;
    }

    for (int a : {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37}) {
        if (n == a)
            return true;
        if (check_composite(n, a, d, r))
            return false;
    }
    return true;
}
```