NHẬP MÔN MÃ HÓA MẬT MÃ

BÀI TẬP NHÓM 1

1. **Lý thuyết**
2. Tìm hiểu

Phép kiểm tra tính nguyên tố của Fermat

* Kiểm tra

N: là số kiểm tra

Chọn ngẫu nhiên 1 số a từ 2 đến N-1

Tìm ước chung lớn nhất của a và N

Nếu khác 1 thì trả về Hợp số

Tiếp tục kiểm tra Fermat theo công thức lý thuyết aN-1  ≡ 1 mod N thì trả về Số nguyên tố còn khác 1 thì trả về Hợp số

* Độ phức tạp: O(log3N)
* Thuật toán dựa trên định lý nhỏ của Fermat aN-1  ≡ 1 mod N

Phép kiểm tra tính nguyên tố của Miller-Rabin

* Kiểm tra:

N: là số kiểm tra

m: là số lẻ

Chọn ngẫu nhiên 1 số a từ 1 đến N-1

Tìm ước chung lớn nhất của a và N

Nếu khác 1 thì trả về Hợp số

Tiếp tục kiểm tra Fermat theo công thức lý thuyết aN-1  % N ≠1 thì trả về Hợp số

Tiếp tục kiểm tra Miller Rabin theo công thức N-1 = 2km và am ≡ 1 mod N

* Độ phức tạp: O(log3N)
* Thuật toán dựa trên định lý nhỏ của Fermat và Miller Rabin

1. Độ phức tạp và tính đúng đắn của 2 thuật toán trên

* Độ phức tạp: như nhau
* Tính đúng đắn:
* Xác suất sai của Fermat là Số giả Fermat
* Xác suất sai của Miller Rabin là (lnN -2)/lnN

1. **Thực hành**
2. Đánh giá

* Ưu điểm:
* Dễ cài đặt.
* Nhược điểm:
* Độ chính xác của Fermat thì thấp hơn của Miller-Rabin.
* Độ phức tạp thời gian cao.
* Thời gian thực hiện: aN-1  ≡ 1 mod N
* Thời gian để sinh chuỗi s: O(len) (len: chiều dài của khóa)
* Thời gian để chuyển từ số nhị phân sang thập phân(BigInt): **O(len \* (n + log(m)))**

**len**: chiều dài của khóa

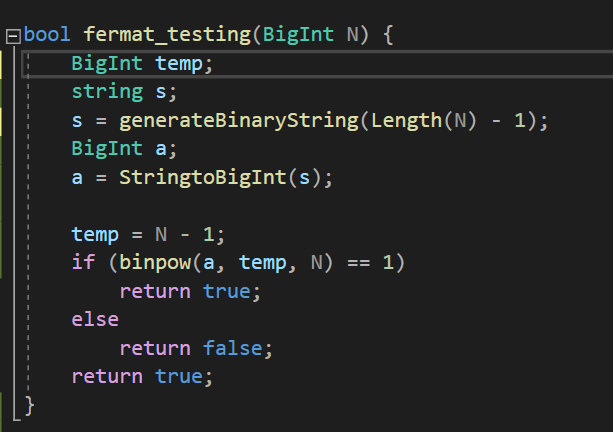
**n**(thời gian thực hiện phép nhân 2 số BigInt): chiều dài của chuỗi

**log(m)**(thời gian thực hiện phép mũ)

* Thời gian để tính aN-1  mod N (hàm binpow): O(2n.log(n))

**n**: chiều dài của số đang kiểm tra

**2n**: thời gian để thực hiện phép nhân và module



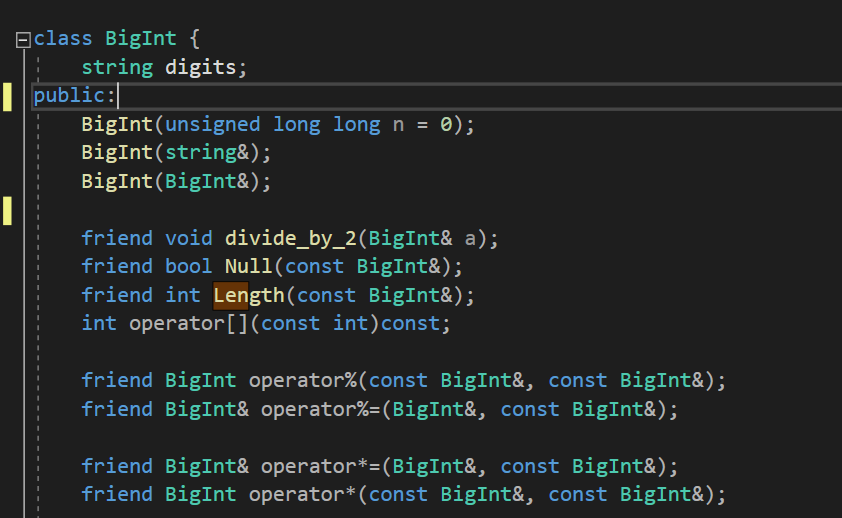
* Thời gian thực hiện của thuật toán: **O(len + (len \* (n + log(m)) + (2n.log(n)))**

1. Báo cáo chi tiết
2. Cách chạy chương trình:

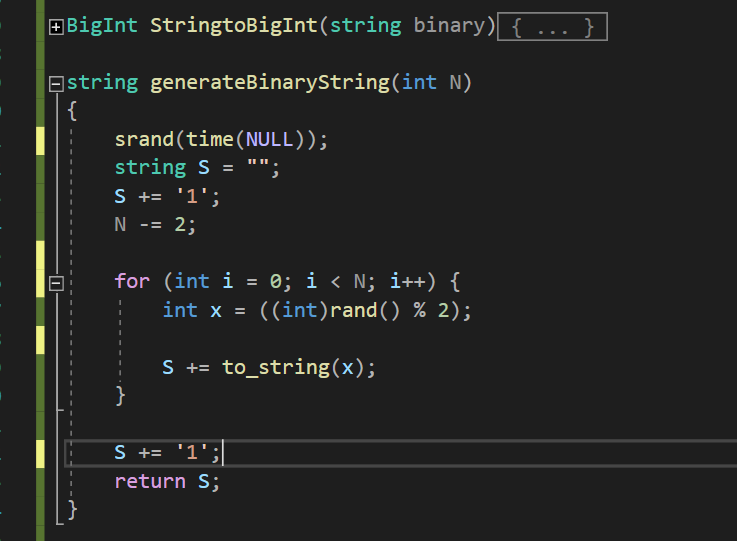
* Đầu tiên, màn hình console hiện lên yêu cầu người dùng nhập độ dài khóa.
* Tiếp theo, người dùng nhập khóa rồi Enter để chạy chương trình.
* Sau đó chờ chương trình chạy cho tới khi in ra được khóa trên màn hình console.

1. Thiết kế chương trình:

* Tính toán và lưu trữ số nguyên lớn
* string digits: dùng để lưu trữ số nguyên lớn
* Overload các hàm operator để tính toán các số nguyên lớn



* Các sinh số nguyên tố lớn tương ứng với từng độ dài khóa
* Sử dụng hàm generateBinaryString để sinh chuỗi bit tương ứng với độ dài khóa (set bit đầu và cuối bằng 1)
* Sử dụng hàm StringtoBigInt để chuyển từ chuỗi bit sang BigInt



* Sử dụng hàm fermat\_testing để kiểm tra số vừa sinh có phải là số nguyên tố hay không?

1. Đánh giá:

* Ưu điểm:
* Dễ cài đặt.
* Nhược điểm:
* Độ phức tạp thời gian cao.
* Thời gian thực hiện: O(a.(**len + (len \* (n + log(m)) + (2n.log(n))**))

a: số lần sinh khóa để được số nguyên tố