ETH Zürich FS 2021
Institute of Theoretical Computer Science
Prof. Emo Welzl
Dr. Johannes Lengler

Algorithmen und Wahrscheinlichkeit
Programming Exercises

Exercise 1 – *Primality testing*

You recently started working in a top-tier cyber security company. You know how cyber security is all about primes, so as a first task your boss gave you a list of numbers and wants you to determine which numbers are prime and which are not.

Input The first line of the input file contains an integer $1 \leq t \leq 10^4$ denoting the number of test cases that follow. Each of the t test cases consists of a line containing an integer `n`. It is guaranteed that $1 \leq n \leq 2^{63} - 1$.

Output For each test case output `yes` if the number is a prime or `no` otherwise.

Points There are two groups of test cases, worth 100 points in total.

1. For the first group of test cases, worth 50 points, you may assume that $1 \leq n \leq 10^5$.
2. For the second group of test cases, worth 50 points, there are no additional assumptions.

Notes

- You should consider using a probabilistic algorithm which gives a wrong answer with some small probability.
- You should consider looking at the pseudocode for the Rabin-Miller algorithm from the script in order to solve this exercise.
- You are strongly advised to use Java's built-in class `BigInteger` and its built-in methods.
- You are strongly advised to use Java's built-in method for fast modular exponentiation `base.modPow(BigInteger exponent, BigInteger modulus)` where `base` is an object of type `BigInteger`.
- You are *strictly prohibited* to use Java's built-in method `n.isProbablePrime(int certainty)` where `n` is an object of type `BigInteger` for which you want to run the test.

Sample Input

```
7
1
2
17
25
2932021007403
3613
2932031007403
```

Sample Output

```
no
yes
yes
no
no
yes
yes
```