Tutorial 2

● Graded

Student

Abhinav Shripad

Total Points

2.5 / 3 pts

Question 1

(no title)

2.5 / 3 pts

+ 0 pts Incorrect

✓ **+ 1.5 pts** Correct algorithm using time complexity which is polynomial in $(\log n)$- 1.5 points

+ 1 pt High level proof ideas correct - 1 point
(must also include why binary search is fine to use, ie exponentiation is monotonic)

✓ **+ 0.5 pts** Proving time complexity - 0.5 points
Expo must not be assumed as O(1)

✓ **+ 0.5 pts** Partial correctness

+ 0.6 pts Explicitly written - " I dont know how to solve this"

COL351: Analysis and Design of Algorithms
Tutorial 2

Name: Abhinav R. Shripad

Entry number: 2022CS11596

Date: August 08, 2024

Group: 3

If a solution exists for a particular N, then a solution exists where b is prime. ie say $N = a^b = a^{cd} = (a^c)^d$ so anytime a solution exists, a prime b also exists

<u>Algorithm:-</u>

```
if n==1:
    return True

primes_till = seive-of-erasthoses(⌈log_2(n)⌉)
              # list of primes till log_2⌈n⌉
              # TC → O(log(n) log(log(n)))

for p in primes_till:
    if check(n, p): # log(n)log(p)
                    #        ↑ using binary search
                    #          and fast-exponentiation
        return true

return false
```

$|\text{primes\_till}| = O(\log(n)/\log(\log(n)))$

$TC = O(\log(n)\log\log(n)) + O\left(\frac{\log(n)}{\log(\log(n))}\right) \cdot O(\log(n)\log(p))$

$\boxed{p \le \log(n)}$

$\boxed{TC = O(\log^2(n))}$