1. Connect each number with its prime factors. $O(n \cdot \text{factor}(U))$.
2. Preprocess an arbitrary prime factor for each number in $1, \dots, U$ using sieve. For each input number, connect all its prime factors. $O(U + n \cdot \frac{\log U}{\log \log U})$. (we can use sieve with sublinear time complexity, e.g. Wheel factorization [2] or Sieve of Atkin [1] in $O(\frac{U}{\log \log U})$ time.)

The running time is also $O(U \log \log U)$ using the sieve of Eratosthenes, because $\sum_i \frac{U}{p_i} = O(U \log \log U)$.
Remark. There could be other running time tradeoffs between n and U.

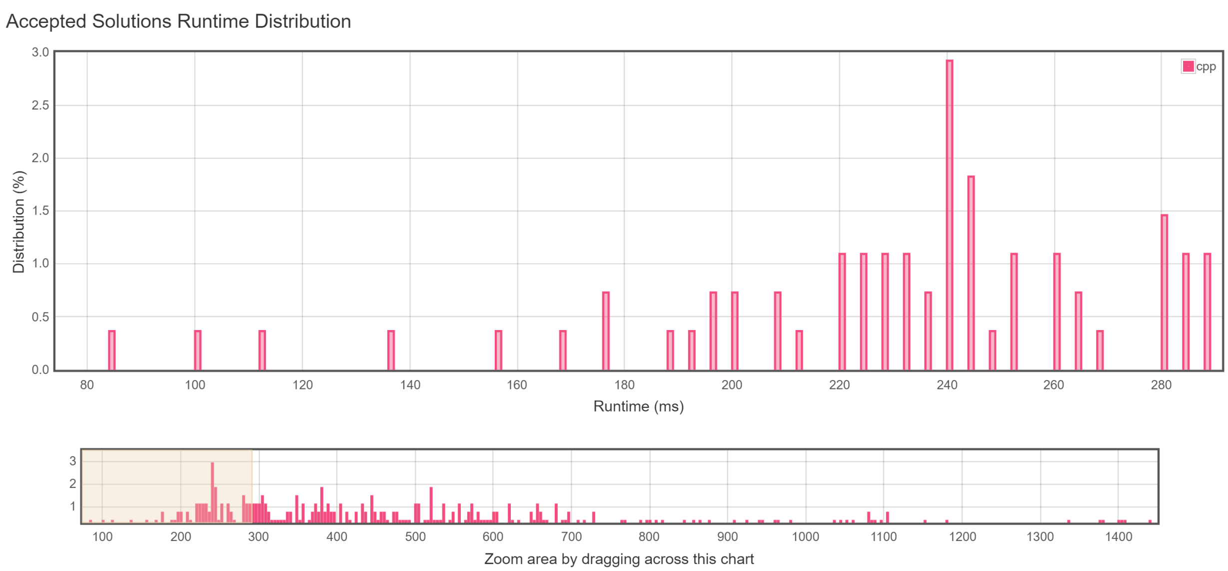
100 / 100 test cases passed.

Runtime: **60 ms**

Memory Usage: **25.8 MB**

Status: Accepted

Submitted: **0 minutes ago**



Runtime: 60 ms, faster than 100.00% of C++ online submissions for Largest Component Size by Common Factor.

Memory Usage: 25.8 MB, less than 97.08% of C++ online submissions for Largest Component Size by Common Factor.

References

[1] Arthur Atkin and Daniel Bernstein. Prime sieves using binary quadratic forms. *Mathematics of Computation*, 73(246):1023–1030, 2004.

[2] Paul Pritchard. Explaining the wheel sieve. *Acta Informatica*, 17:477–485, 1982.