Contest Duration: 2025-06-28(Sat) 08:00 (http://www.timeanddate.com/worldclock/fixedtime.html?iso=20250628T2100&p1=248) - 2025-06-28(Sat) 09:40 (http://www.timeanddate.com/worldclock/fixedtime.html?iso=20250628T2240&p1=248) (local time) (100 minutes) Back to Home (/home)

Top (/contests/abc412) Tasks (/contests/abc412/tasks)

Clarifications (/contests/abc412/clarifications) Submit (/contests/abc412/submit)

Results Standings (/contests/abc412/standings)

Virtual Standings (/contests/abc412/standings/virtual)

Custom Test (/contests/abc412/custom_test) Editorial (/contests/abc412/editorial)

Discuss (https://codeforces.com/blog/entry/144250)

Official

E - LCM Sequence (/contests/abc412/tasks/abc412_e) Editorial by **en_translator** (/users/en_translator)

Since A_n is monotonically increasing, the answer to the problem is

- $1+$ (the number of integers i among $L+1 \leq i \leq R$ and $A_{i-1} < A_i$)

(The first 1 corresponds to A_L.)

Which A_i satisfies $A_{i-1} < A_i$?
We have $A_n = \mathrm{LCM}(1, 2, \ldots, n)$. Consider how many times p divides A_n. Among $1, 2, \ldots, n$, the integer that can be divided by p that most times is

- the maximum p^k with $p^k \leq n$ (k is an integer),

and the number of times p divides the integer is k, so A_n is also divided by p, k times (using the integer k above).

Therefore, if and only if n can be represented as a power of a prime p (which is called a **prime power**), A_n can be divided by p one more time than A_{n-1}. For the other primes than p, the number of times it divides A_n is same as that for A_{n-1}, so we can say that A_n is p times A_{n-1}.
Conversely, if n is not a prime power, then for any prime p, the number of times p divides A_n is the same as that for A_{n-1}, so $A_n = A_{n-1}$.

By the discussion above, the answer to this problem turns out to be

- $1+$ (the number of prime powers i such that $L+1 \leq i \leq R$)

2025-06-30 (Mon) 00:06:39 -04:00

It is known that one can find the prime factorizations of all integers $L+1,\ldots,R$ using an algorithm called **segmented sieve** in $\mathrm{O}(M \log\log M)$ time, where $M = \max(R-L, \sqrt{R})$. This algorithm is already featured in past problems, so please refer to the editorial for the past problem (https://atcoder.jp/contests/abc227/editorial/2909)).
To roll up, the problem can be solved in a total of $\mathrm{O}(M \log\log M)$ time, where $M = \max(R - L, \sqrt{R})$, which is fast enough.

- Sample code (C++)

Copy

```
#include <cmath>
#include <iostream>
#include <vector>
using namespace std;

vector<int> prime_enumerate(int N) {
  vector<bool> is_prime(N + 1, true);
  vector<int> primes;
  if (N < 2) return primes;
  is_prime[0] = is_prime[1] = false;
  for (int i = 2; i * i <= N; ++i) {
    if (is_prime[i]) {
      for (int j = i * i; j <= N; j += i) is_prime[j] = false;
    }
  }
  for (int i = 2; i <= N; ++i) {
    if (is_prime[i]) primes.push_back(i);
  }
  return primes;
}

int main() {
  long long L, R;
  cin >> L >> R;
  vector<int> vis(R - L);
  int ans = 1;
  for (int p : prime_enumerate(sqrt(R) + 100)) {
    for (long long x = (L / p + 1) * p; x <= R; x += p) {
      if (vis[x - (L + 1)]) continue;
      vis[x - (L + 1)] = 1;
      long long y = x;
      while (y % p == 0) y /= p;
      if (y == 1) ans++;
    }
  }
  for (int v : vis) ans += v == 0;
  cout << ans << "\n";
}
```

posted: a day ago

last update: a day ago

am)

ps%3A%2F%2Fatcoder.jp%2Fcontests%2Fabc412%2Feditorial%2F13393%3Flang%3Den&title=Editorial%20-
)412)

Rule (/contests/abc412/rules) **Glossary (/contests/abc412/glossary)**

Terms of service (/tos) **Privacy Policy (/privacy)** **Information Protection Policy (/personal)** **Company (/company)**
FAQ (/faq) **Contact (/contact)**

2025-06-30 (Mon)
00:06:39 -04:00