CF1946F

排列 a，对于提问 $[l,r]$，回答$[l,r]$ 中**索引单增**的且**前一个数能被后一个数整除**的索引集合的个数。

- 扫描线，令 $F[i]$ 表示以 i 结尾的 $a_{end} = a[x]$ 的序列个数。
- 离线记录每个 l 对应的 $\{r, id\}$，i 从 n 扫到 1，$[i,r]$ 答案为 $\sum_i^r F[i]$ 利用线段树或树状数组维护
- 考虑如何为 $F[i]$ 贡献答案。
  - 处理 $x = a[i]$，$dp[x] = 1$，$dp[k]$ 意味着数值为 k 的位置要增加的贡献，显然 $[i,i]$ 对应 $dp[x] = 1$
  - 处理 x 的倍数，找到 x, y, z 使得 $x|y, y|z$，且 $pos[y] >= pos[x], pos[z] >= pos[y]$
    - 可以 DP 求解，从小到大枚举倒数第二个数 y，和最后一个数 z，$dp[z]$ += $dp[y]$
- 复杂度 $O(nlog^2n)$
  - $\sum_i \sum_j N/i/j < N \sum_i 1/i \sum_j 1/j = NlogNlogN$

```
void solve() {
    int n, Q;
    std::cin >> n >> Q;
    std::vector<int> a(n + 1), pos(n + 1);
    for (int i = 1; i <= n; i ++) {
        std::cin >> a[i];
        pos[a[i]] = i;
    }
    std::vector<std::vector<std::array<int, 2 > > > q(n + 1);
    std::vector<int> ans(Q + 1);
    for (int i = 1; i <= Q; i ++) {
        int l, r;
        std::cin >> l >> r;
        q[l].push_back({r, i});
    }
    Fenwick t(n);
    std::vector<int> dp(n + 1);

    for (int i = n; i >= 1; i --) { // end with i.
        int x = a[i];
        dp[x] = 1;
        for (int y = x; y <= n; y += x) {
            if (pos[y] < pos[x]) continue;
            for (int z = 2 * y; z <= n; z += y) {
                if (pos[z] < pos[y]) continue;
                dp[z] += dp[y];
            }
        }

        for (int y = x; y <= n; y += x) {
            t.add(pos[y], dp[y]);
            dp[y] = 0;
        }

        for (auto it : q[i]) {
            int r = it[0], id = it[1];
            ans[id] += t.rangeSum(i, r);
        }
    }
```

```
    for (int i = 1; i <= Q; i ++) {
        std::cout << ans[i] << ' ';
    }
    std::cout << '\n';
}
```