Sieve Prime factorization

- generate all prime factors for queries --> time complexity --> $nlog(n)$
- same as sieve but instead of true and false we put the prime itself
- now every index has the smallest prime possible
- we can now index/value to get the new index and so on

code 1

```
// outside the tests : sieve_prime_fact()
// inside tests :prime_fact(the number that require prime fac)
const int N = 1000005;
int arr[N];
void sieve_prime_fact() {
    for (int i = 0; i < N; ++i) arr[i] = -1;
    for (int i = 2; i < N; ++i) {
        if (arr[i] == -1) {
            for (int j = i; j < N; j += i) {
                if (arr[j] == -1) arr[j] = i;
            }
        }
    }
}
vector<int> prime_fact(int n){
    vector<int> fac;
    while (n != 1)
        fac.push_back(arr[n]), n/=arr[n];
    return fac;
}
```

code 2

```
// outside the tests :prime_fact_queries()
const int N = 1000005;
vector<vector<pair<int, int>>> prime_fact_queries(){
    int n = N;
    vector<vector<pair<int, int>>> arr(n);
    vector<bool> prime(n, true);
    for (int i=2; i < n; i++) {
        if (!prime[i]) continue;
        for (int j = i; j < n; j += i) {
            prime[j] = false;
            int cnt = 0, cop = j;
            while (cop % i == 0)
                cop /= i, cnt++;
            arr[j].push_back({i, cnt});
        }
    }
    return arr;
}
```