divisors count, sum #numberTheory

```
bitset<10000> bs;
vector<int> primes;

void sieve(long long upper_bound) {
    primes.erase(primes.begin(), primes.end());
    bs.set();
    bs[0] = bs[1] = 0;
    primes.push_back(2);
    for(long long i = 3; i <= upper_bound + 1; i += 2) {
        if(bs[i]) {
            for(long long j = i * i; j <= upper_bound + 1; j += 2*i)
                bs[j] = 0;
            primes.push_back((int) i);
        }
    }
}
int countDivisor1(int n) {
    int divisor = 0;
    for (int i = 1; i * i <= n; i++) {
        if (i * i == n) {
            divisor += 1;
        } else if (n % i == 0) {
            divisor += 2;
        }
    }
    return divisor;
}
int countDivisor2(int n) {
    /*
     * Let X = p1^x + p2^y.... where p1, p2 are prime numbers
     * Then total number of divisors that X has is
     * (x+1) * (y+1) * ....
     */
    if (n < 1) return 0;
    sieve(n+1);
    int divisor = 1;
    for (int i = 0; primes[i]*primes[i] <= n; i++) {
        if (n % primes[i] == 0) {
            int cnt = 1;
            while (n % primes[i] == 0) {
                n /= primes[i];
                cnt++;
            }
            divisor *= cnt;
        }
    }
    if (n > 1) divisor *= 2;
    return divisor;
}
int divisorSum1(int n) {
    int cnt = 0;
    for (int i = 1; i * i <= n; i++) {
        if (i * i == n) {
            cnt += 2*n;
        } else if (n % i == 0) {
            cnt += (i + n/i);
        }
    }
    return cnt;
}
```

```
int divisorSum2(int n) {
    if (n < 1) return 0;
    sieve(n+1);
    int sum = 1;
    for (int i = 0; primes[i]*primes[i] <= n; i++) {
        if (n % primes[i] == 0) {
            int cnt = 1;
            while (n % primes[i] == 0) {
                n /= primes[i];
                cnt++;
            }
            sum *= (pow(primes[i], cnt) - 1) / (primes[i] - 1);
        }
    }
    if (n > 1) sum *= (pow(n, 2) - 1) / (n - 1);
    /*
     * this is just a formula of a series (1 + p + p^2 + ... + p^n)
     * = ((p^(n+1)) - 1)/(p-1)
     */
    return sum;
}
```