204. Count Primes

link

cs593的轮盘办法

自己写的, 有点🍔

```
public int countPrimes(int n) {
    if(n <= 1) return 0;
    int[] res = new int[n];
    int i = 2, count = 0;
    while(i < n){
        if(res[i] == 0) {
            res[i] = 1;
            count++;
        }else if(res[i++] == -1)
            continue;
        int j = 2;
        while(j * i < n){
            res[j * i] = -1;
            j++;
        }
        i++;
    }
    return count;
}
```

人家的

```
public int countPrimes(int n) {
    boolean[] notPrime = new boolean[n];
    int count = 0;
    for (int i = 2; i < n; i++) {
        if (notPrime[i] == false) {
            count++;
            for (int j = 2; i*j < n; j++) {
                notPrime[i*j] = true;
            }
        }
    }

    return count;
}
```

可以根据奇偶以及性质继续优化

```
publib int countPrimes(int n) {
    if (n < 3)
        return 0;

    boolean[] f = new boolean[n];
    //Arrays.fill(f, true); boolean[] are initialed as false by default
    int count = n / 2;
    for (int i = 3; i * i < n; i += 2) {
        if (f[i])
            continue;

        for (int j = i * i; j < n; j += 2 * i) {
            if (!f[j]) {
                --count;
                f[j] = true;
            }
        }
    }
    return count;
}
```

解释

```
public int countPrimes(int n) {
    //默认值是false,如果是不是素数我们标为true
    //当然你也可以使用 Arrays.fill(f, true);  默认全设为true，如果不是素数我们设为false
    boolean s[] = new boolean[n];

    //如果n小于3的话就没有  因为找到n以内  所以2以内是没有的
    if(n < 3)return 0;

    //c 可以理解为最多的素数个数
    //以为我们知道所有的偶数都不是素数，所有我们可以剔除一半
    //但是你可能会有疑问 2 不是偶数吗   --> 这里 2 和 1 相抵消
    //比如 5 以内的话  5 / 2 = 2 素数就为 2 和 3
    //首先我们假设 小于 c 的奇数全是素数
    int c = n / 2;


    // 之后我们只要剔除 在这个奇数范围内 不是素数的数就可以了
    // 因为我们已经把偶数去掉了，所以只要剔除奇数的奇数倍就可以了
    for(int i = 3; i * i < n; i += 2) {
        //说明 i 是不是素数，而且已经是剔除过的
        if(s[i])
            continue;

        //这里是计算c 中剔除完不是素数的奇数个数  下面解释各个值的含义
        //我们要剔除的是 i 的 奇数倍
        //为什么是 i * i开始呢   我们打个比方，假设我们此时i = 5
        //那么我们开始剔除  j = 1 时就是本身，此时要么已经被剔除，要么就是素数，所以 1 不考虑
        //当 j = 2 || j = 4时，乘积为偶数所以也不在我们考虑范围内
        //当 j = 3时，我们考虑 3 * 5 但是这种情况已经是当 i = 3的时候被考虑进去了所以我们只要考虑之后的就可以了

        //那么为什么 j = j + i * 2呢
        //根据上面所说 我们从3开始考虑 3 * 3,3 * 5,3 * 7....只要 j < n 我们就剔除
        //带入i : i * i, i * ( i + 2 ) , i * ( i + 4 )....
        for(int j = i * i; j < n; j+= i * 2) {
            //只要找到c个奇数中的合数，c就减1，把 j标记为非素数
            if(!s[j]) {
                s[j] = true;
                c --;
            }
        }
    }
    return c;
}
```