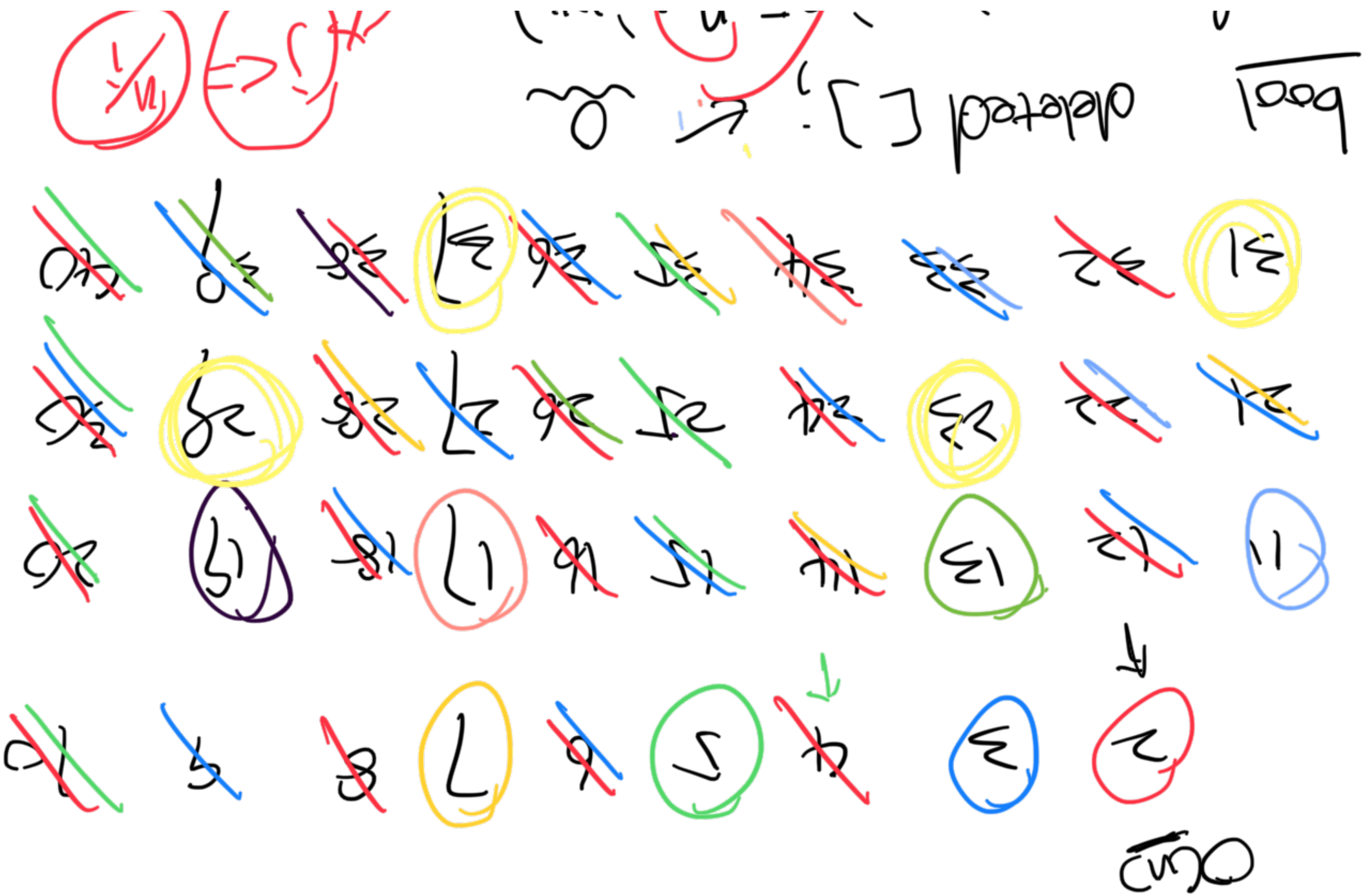


for (int i=2; i<=n; i++)

if (deleted[i] != true)

for (int j=2; ; j++)

deleted[i*j] = true;

i*j <= n

deleted[i] == 0

prime[ ]

for (int i=2; i<=n; i++)

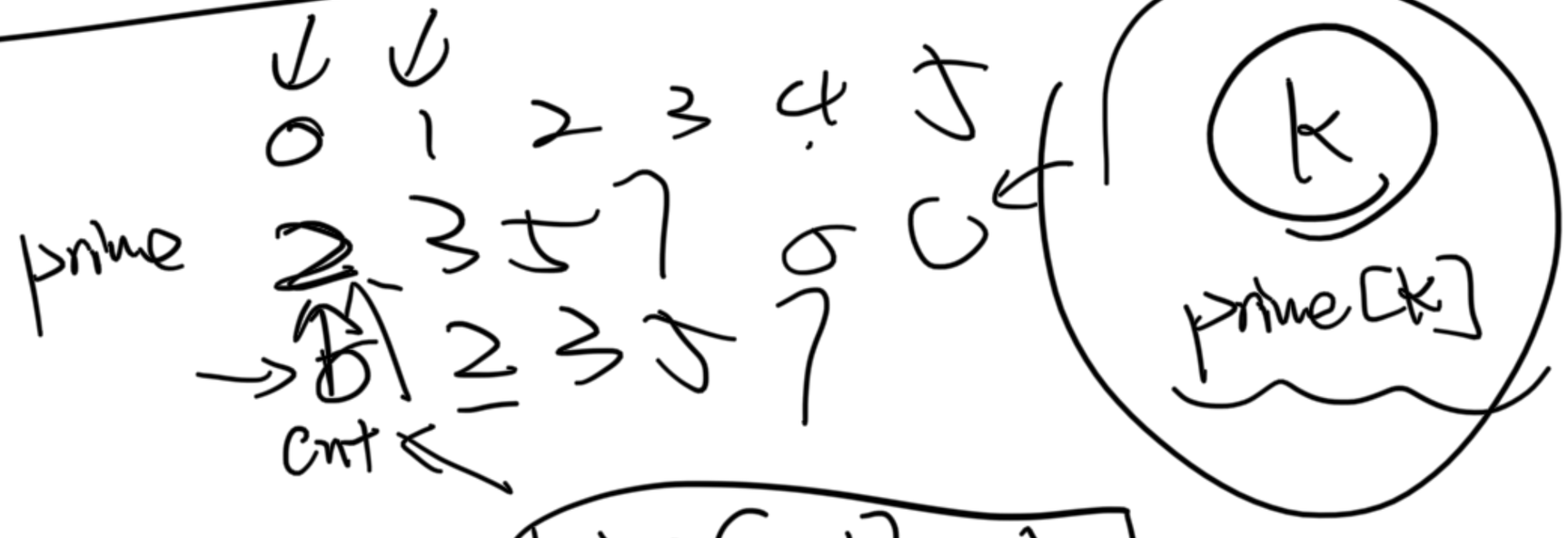
if (deleted[i]==0)

→ prime[cnt] = i

→ cnt++

cnt++

prime[cnt] = i



cnt

prime[cnt] = i

cnt=0

prime[cnt]=3 → cnt++

2 3 5 7 11 13 17 19 23 29 31 37

6

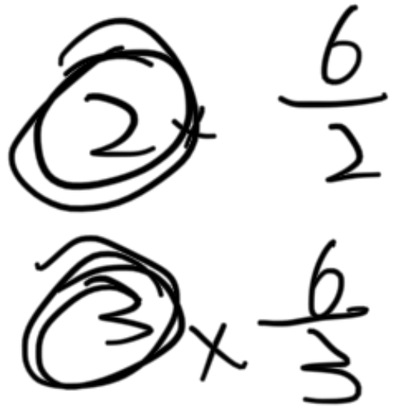
2×2 3×2 5×2 7×2 11×2 13×2 17×2 19×2

2×3 3×3 5×3 7×3 11×3 13×3

2×4 3×4 5×4 7×4

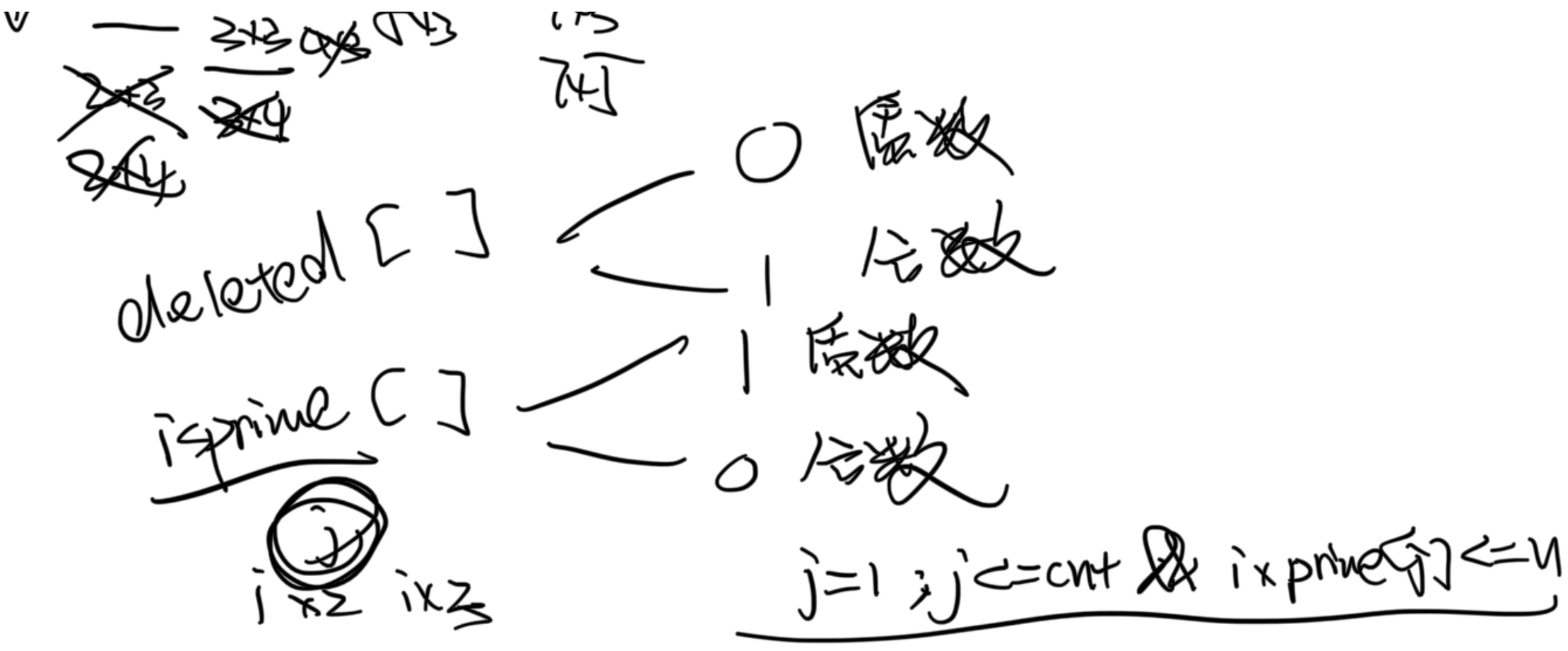
2×5 5×5 7×5

2×20 3×13



2 3 4 5 6 7

2×2 3×2 4×2 5×2 6×2 7×2




```java
for (int i = 2; i <= n; i++) {
    if (isPrime[i]){//没筛掉
        Prime[++cnt] = i; //i成为下一个素数
    }

    for (int j = 1; j <= cnt && i * Prime[j] <= n/*不超上限*/; j++) {
        //从Prime[1]，即最小质数2开始，逐个枚举已知的质数，并期望Prime[j]是(i*Prime[j])的最小质因数
        //当然，i肯定比Prime[j]大，因为Prime[j]是在i之前得出的
        isPrime[i * Prime[j]] = 0;

        if (i % Prime[j] == 0) {//i中也含有Prime[j]这个因子
            break; //重要步骤。见原理
        }
    }
```

n = 17

true

2 3 4 1 6 7 8 5 10 11 12 13 14 15

→ Prime 2 3 1 7 11 13

cnt

x3

x2

6 x 2 = 12

2x2

isprime(x) = false

x

true

→ 2 3 4 3 1 6 7 0

4 6 8 9 10 12 14 15

4 = 2x2

6 = 2x3

8 = 2x4

8 = 2x2x2

9 = 3x3

10 = 2x5

12 = 2x6

12 = 2x2x3

14 = 2x7

15 = 3x5

i = 7x11 = 77

1x2

1x3

77 x 2

77 x 3