

# Aufgabe 1:

```
CMakeLists.txt main.c
1 #include <stdio.h>
2
3 int iterative(int a, int b) {
4     int r;
5     do {
6         r = a % b;
7         a = b;
8         b = r;
9     } while (b != 0);
10
11     return a;
12 }
13
14 int recursive(int a, int b) {
15     if (b == 0)
16         return a;
17     return recursive(a, b - a % b);
18 }
19
20
```

↙ Iterative

↙ recursive

performanter  
wenn je R. & C für  
Stack overflow

Rekursive Funktion degradiert

```
int kgV_func(int a, int b) {
    int val = iterative(a, b);
    return (abs(x:a * b) / val);
}

int main(void) {
    // Werte sind: a, b, a*b, ggT(a,b). kgV(a,b)
    for (int a = 30; a <= 40; a++) {
        for (int b = 30; b <= 40; b++) {
            int product = a * b;
            int it = iterative(a, b);
            int kgV = kgV_func(a, b);
            printf(" %d %d %d %d\n",
                   a, b, product, it, kgV);
        }
    }
    return 0;
}
```

```
40 35 1400 5 280
40 36 1440 4 360
40 37 1480 1 1480
40 38 1520 2 760
40 39 1560 1 1560
40 40 1600 40 40
```

```
Process finished with exit code 0
```

Was fällt auf?

$$a \cdot b = ggT(a,b) \cdot kgV(a,b)$$

# Aufgabe 2:

1)

2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28
29	30	31	32	33	34	35	36	37
38	39	40	41	42	43	44	45	46
47	48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63	64
65	66	67	68	69	70	71	72	73
74	75	76	77	78	79	80	81	82
83	84	85	86	87	88	89	90	91
92	93	94	95	96	97	98	99	100

2)

```
CMakeLists.txt    main.c    a2.c x
1 #include <stdio.h>
2
3 #include <stdbool.h>
4
5 void sieb_von_eratosthenes(int k, bool isPrime[])
6 {
7     for (int i = 0; i <= k; i++) {
8         if(i==0) {
9             isPrime[i] = true;
10        }
11    }
12
13    for (int i = 2; i * i <= k; i++) {
14        if (isPrime[i]) {
15            for (int j = i * i; j <= k; j += i) {
16                isPrime[j] = false;
17            }
18        }
19    }
20 }
21
22 int main(void)
23 {
24     int k;
25     printf("Bitte \"k\" eingeben: ");
26     scanf("%d", &k);
27
28     bool isPrime[10000];
29
30     sieb_von_eratosthenes(k, isPrime);
31
32     for (int i = 2; i <= k; i++) {
33         if (isPrime[i]) {
34             printf("%d ", i);
35         }
36     }
37     return 0;
38 }
```

Alles auf true setzen

Aussortieren

Abfrage von k  
berechnen  
Ausgabe

3)

```
/home/usr/CLionProjects/ad/cmake-build-debug/ad
k: 100000
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89
Process finished with exit code 0
```

Aufgabe 3:

Datei a3.c

Mit Tutor besprochen

- Evtl. nächstes mal mit Klasse