

# Euklidischer Algorithmus

(Euklidischer Algorithmus)

**Stichwörter:** Mehr-Adress-Befehl-Assembler

## Euklidischer Algorithmus

Nach Euklid lässt sich der größte gemeinsamer Teiler zweier Zahlen *a* und *b* bestimmen mit :

*Wenn CD aber AB nicht misst, und man nimmt bei AB, CD abwechselnd immer das kleinere vom größeren weg, dann muss (schließlich) eine Zahl übrig bleiben, die die vorangehende misst.*

Erstelle ein Assemblerprogramm, das seine beiden Parameter über zwei Variablen *a* und *b* aus dem Speicher übernimmt und den  $\text{ggt}(a, b)$  berechnet. Das Ergebnis soll in *R0* liegen.

Lösungsvorschlag

```
-- Euklidischer Algorithmus

-- Nach Euklid lässt sich der grösste gemeinsamer Teiler zweier Zahlen a und b bestimmen mit:
-- Wenn CD aber AB nicht misst, und man nimmt bei AB, CD abwechselnd immer das
-- kleinere vom grösseren weg, dann muss (schliesslich) eine Zahl uebrig bleiben, die
-- die
-- vorangehende misst.
-- Erstelle ein Assemblerprogramm, das seine beiden Parameter ueber zwei Variablen a
-- und b aus dem Speicher uebernimmt und den ggt(a, b) berechnet. Das Ergebnis soll in
-- R0 liegen.

euklid:
SEG

a R0
b R1
erg R0

public static int euklid(int a, int b) {
    if (a == 0)
        return b;
    while (b != 0) {
        if (a > b)
            a = a - b;
        else
            b = b - a;
    }
    return a;
}

einstieg:      MOVE W a, R0
               MOVE W b, R1

               -- if (a == 0)
               CMP W R0, I 0
               JEQ gibBZurueck
```

```

solange:
    CMP W R1, I 0
    JEQ gibAZurueck
    -- if (a > b)
    CMP W R0, R1
    JLE differenzB

    -- a = a - b;
differenzA:
    SUB W R1, R0
    JUMP solange

    -- b = b - a;
differenzB:
    SUB W R0, R1
    JUMP solange

    -- return b;
gibBZurueck:
    MOVE W R1, R0
    HALT

    -- return a;
gibAZurueck:
    HALT

a:
    DD W 7
b:
    DD W 49 -- 7

-- Tests

-- a:
-- b:
-- DD W 3780
-- DD W 3528 -- 252

-- a:
-- b:
-- DD W 12
-- DD W 18 -- 6

-- a:
-- b:
-- DD W 17
-- DD W 1 -- 1

-- a:
-- b:
-- DD W 0
-- DD W 3 -- 3

-- a:
-- b:
-- DD W 3
-- DD W 0 -- 3
END

```

```

public class Euklid {

    public static int euklid(int a, int b) {
        if (a == 0)
            return b;
        while (b != 0) {
            if (a > b)
                a = a - b;
            else
                b = b - a;
        }
    }
}

```

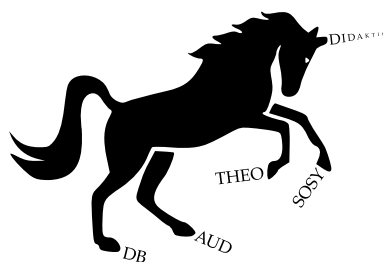
```

    }
    return a;
}

public static void main(String[] args) {
    System.out.println(euklid(3780, 3528)); // 252
    System.out.println(euklid(12, 18)); // 6
    System.out.println(euklid(17, 1)); // 1
    System.out.println(euklid(1, 17)); // 1
    System.out.println(euklid(0, 3)); // 3
    System.out.println(euklid(3, 0)); // 3
}
}

```

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/aufgaben/tech\\_info/assembler/mehr\\_adress/Euklid.java](https://github.com/src/main/java/org/bschlangaul/aufgaben/tech_info/assembler/mehr_adress/Euklid.java)



## Die Bschlangaul-Sammlung

### Hermine Bschlangaul and Friends

Eine freie Aufgabensammlung mit Lösungen von Studierenden für Studierende zur Vorbereitung auf die 1. Staatsexamensprüfungen des Lehramts Informatik in Bayern.



Diese Materialsammlung unterliegt den Bestimmungen der Creative Commons Namensnennung-Nicht kommerziell-Share Alike 4.0 International-Lizenz.

Hilf mit! Die Hermine schafft das nicht allein! Das ist ein Community-Projekt! Verbesserungsvorschläge, Fehlerkorrekturen, weitere Lösungen sind herzlich willkommen - egal wie - per Pull-Request oder per E-Mail an [hermine.bschlangaul@gmx.net](mailto:hermine.bschlangaul@gmx.net). Der TeX-Quelltext dieses Dokuments kann unter folgender URL aufgerufen werden: [https://github.com/bschlangaul-sammlung/examens-aufgaben/blob/main/Module/50\\_TECH/20\\_Mehr-Adress/Aufgabe\\_08-Euklid.tex](https://github.com/bschlangaul-sammlung/examens-aufgaben/blob/main/Module/50_TECH/20_Mehr-Adress/Aufgabe_08-Euklid.tex)