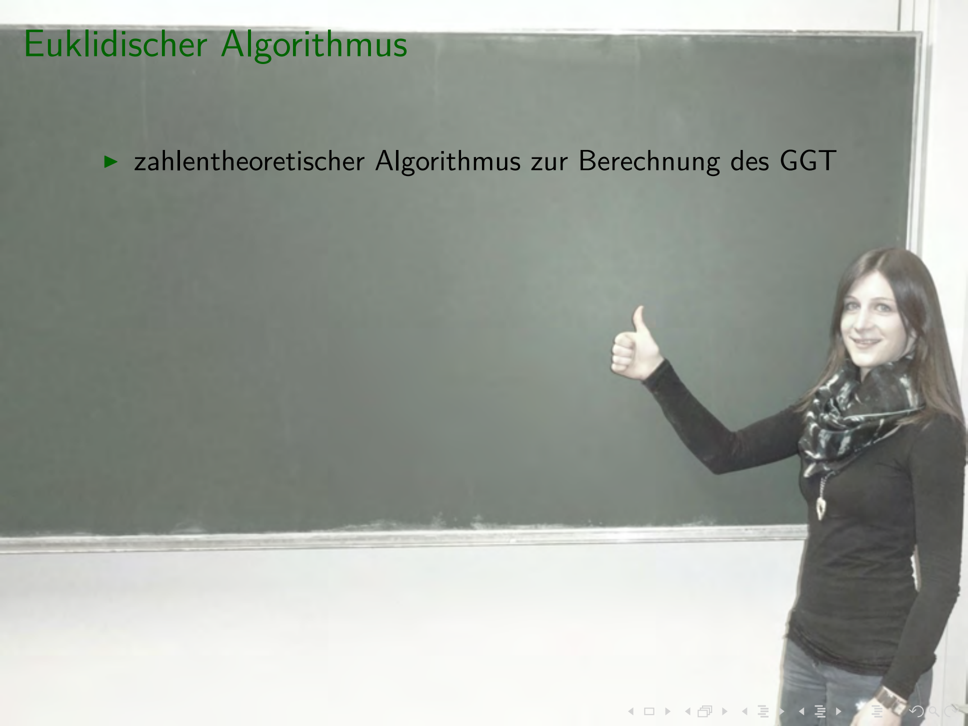
Teresa Bachinger
Cornelia Mayer

EMMA - Ein Projekt im Rahmen von Sparkling Science

Inhaltsverzeichnis



Euklidischer Algorithmus

- zahlentheoretischer Algorithmus zur Berechnung des GGT

Euklidischer Algorithmus

- zahlentheoretischer Algorithmus zur Berechnung des GGT
- alternative Berechnungsmethode: mit Hilfe der Primfaktorzerlegung

Euklidischer Algorithmus

- zahlentheoretischer Algorithmus zur Berechnung des GGT
- alternative Berechnungsmethode: mit Hilfe der Primfaktorzerlegung
- Vorteile des euklidischen Algorithmus:

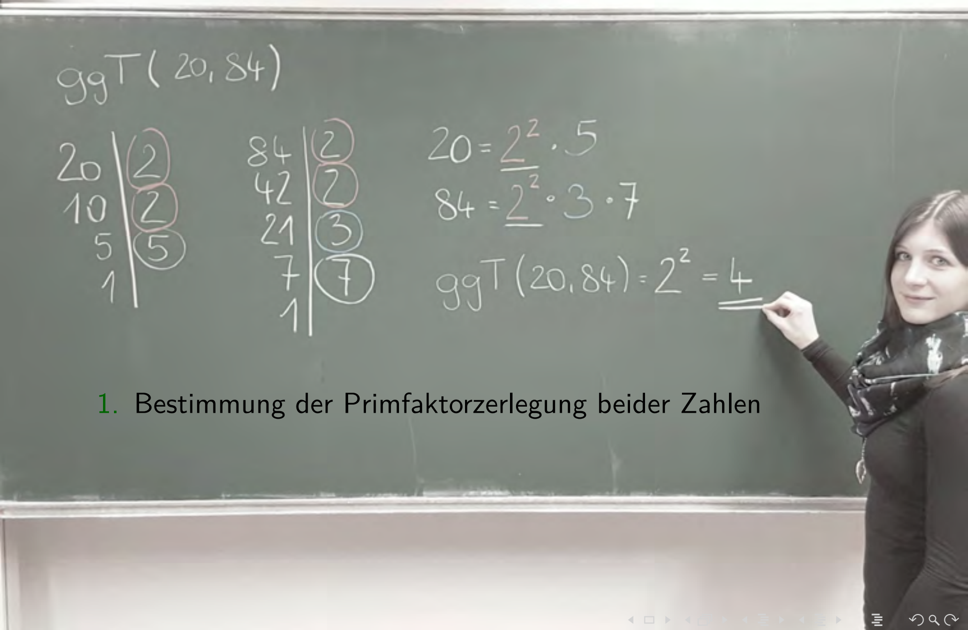
Euklidischer Algorithmus

- zahlentheoretischer Algorithmus zur Berechnung des GGT
- alternative Berechnungsmethode: mit Hilfe der Primfaktorzerlegung
- Vorteile des euklidischen Algorithmus:
  - beruht nicht auf Primfaktorzerlegung

Euklidischer Algorithmus

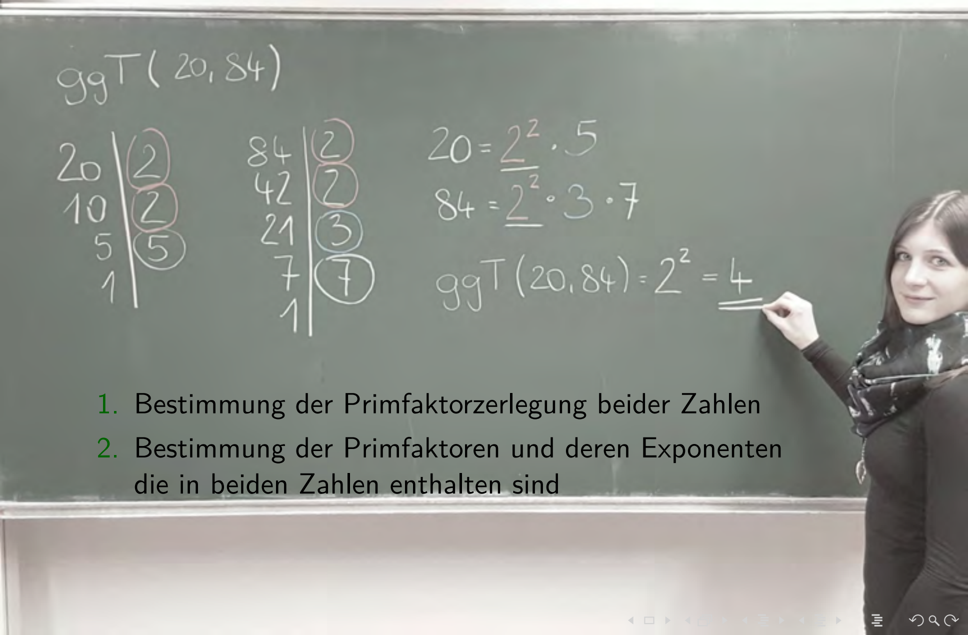
- zahlentheoretischer Algorithmus zur Berechnung des GGT
- alternative Berechnungsmethode: mit Hilfe der Primfaktorzerlegung
- Vorteile des euklidischen Algorithmus:
  - beruht nicht auf Primfaktorzerlegung
  - effizientes Verfahren

Schulmethode zur Bestimmung des GGT



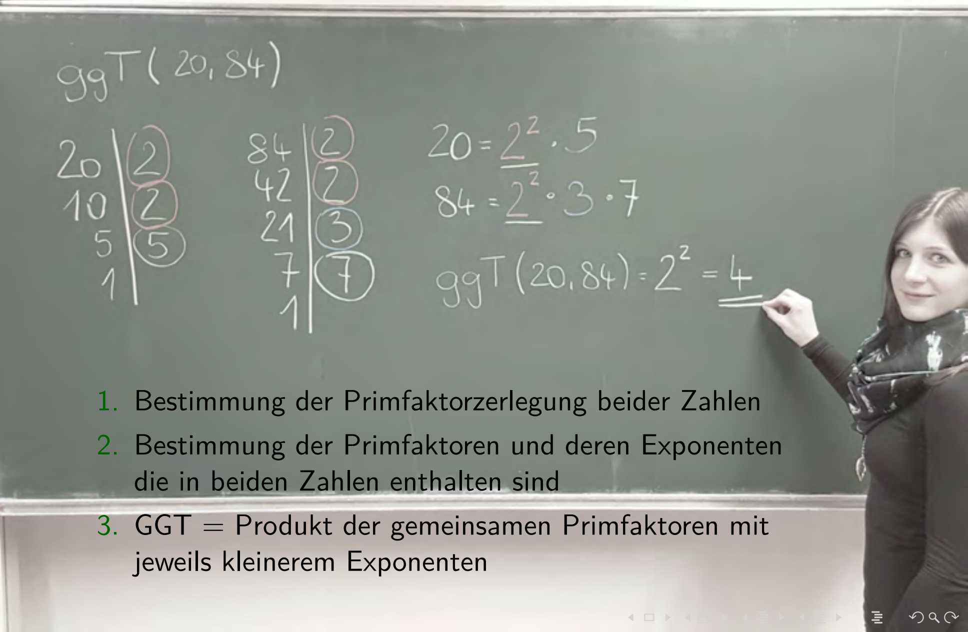
1. Bestimmung der Primfaktorzerlegung beider Zahlen

Schulmethode zur Bestimmung des GGT



1. Bestimmung der Primfaktorzerlegung beider Zahlen
2. Bestimmung der Primfaktoren und deren Exponenten die in beiden Zahlen enthalten sind

Schulmethode zur Bestimmung des GGT



1. Bestimmung der Primfaktorzerlegung beider Zahlen
2. Bestimmung der Primfaktoren und deren Exponenten die in beiden Zahlen enthalten sind
3. GGT = Produkt der gemeinsamen Primfaktoren mit jeweils kleinerem Exponenten

Schulmethode zur Bestimmung des GGT

1. Bestimmung der Primfaktorzerlegung beider Zahlen
2. Bestimmung der Primfaktoren und deren Exponenten die in beiden Zahlen enthalten sind
3. GGT = Produkt der gemeinsamen Primfaktoren mit jeweils kleinerem Exponenten

Um die Bedeutung des euklidischen Algorithmus zu untermauern, wird mittels eines Beispiels an die bereits bekannte Berechnungsmethode für den größten gemeinsamen Teiler erinnert.
Diese stellt jedoch vor allem bei großen Zahlen einen immensen Rechenaufwand dar. Der euklidische Algorithmus schafft dabei Abhilfe.


DER ALGORITHMUS
• Sei $a, b \in \mathbb{Z}_{\geq 0}$
• Es gilt: $ggT(a,b) = \begin{cases} ggT(|a|, b \bmod |a|), & \text{falls } a \neq 0 \\ |b| & \text{sonst} \end{cases}$
• $r_0 = a$, $r_1 = b$
• $r_{k+1} = r_{k-1} \bmod r_k \iff r_{k-1} = r_k \cdot q_k + r_{k+1}$
• Wiederhole den Divisionsschritt solange $r_{k+1} \neq 0$
• Falls $r_{n+1} = 0$, gilt $ggT(a,b) = r_n$

Der euklidische Algorithmus funktioniert wie auf der Folie dargestellt. Das auf der nächsten Folie folgende Beispiel veranschaulicht die Vorgangsweise zur Berechnung des ggT mittels euklidischem Algorithmus.

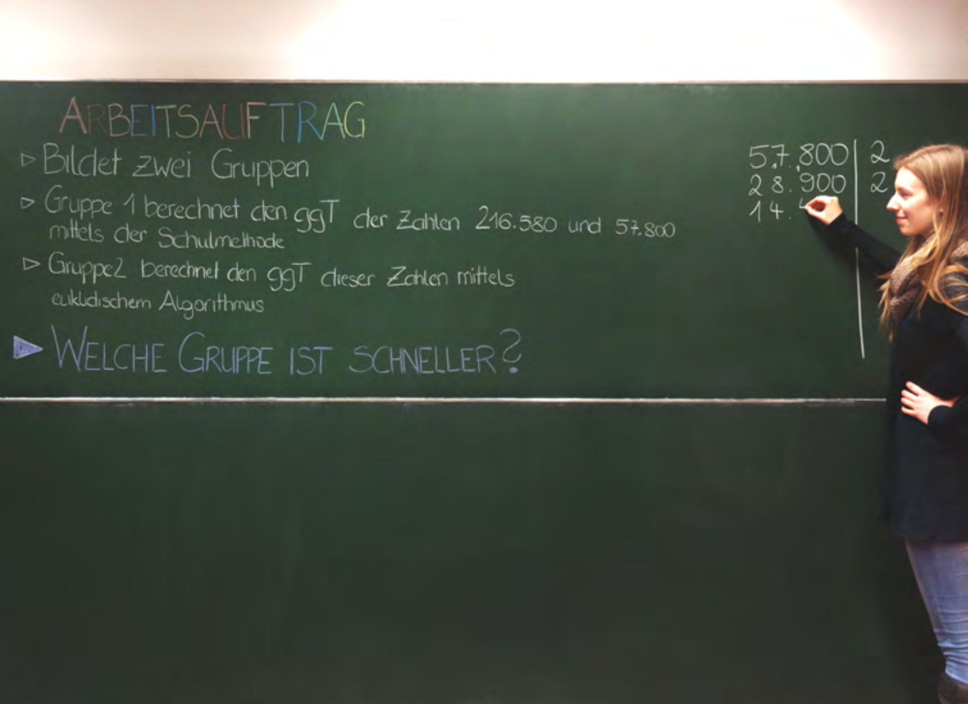
EUKLIDISCHER ALGORITHMUS

$\mathrm{ggT}(6.172.530, 6.279)$

$\boxed{r_0} = 6.172.530, \quad \boxed{r_1} = 6.279$

$$\boxed{6.172.530} = \boxed{6.279} \cdot 983 + \boxed{273}$$
$$\boxed{6.279} = \boxed{273} \cdot 23 + 0$$

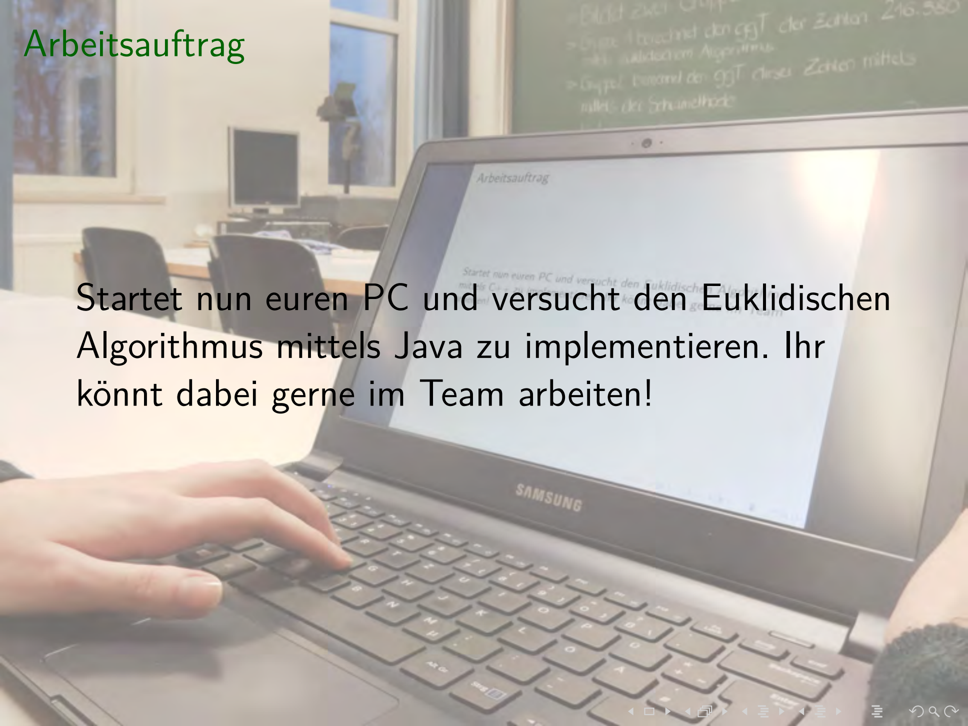
$\Rightarrow \mathrm{ggT}(6.172.530, 6.279) = \boxed{273}$


ARBEITSAUFTRAG
▷ Bildet zwei Gruppen
▷ Gruppe 1 berechnet den ggT der Zahlen 216.580 und 57.800 mittels der Schulmethode
▷ Gruppe2 berechnet den ggT dieser Zahlen mittels euklidischem Algorithmus
▶ WELCHE GRUPPE IST SCHNELLER?
57.800 | 2
28.900 | 2
14.

Um den Unterschied bezüglich des Zeitaufwands deutlich zu machen, sollen die Schüler/innen die beiden Methoden zur Berechnung des ggT in einem kleinen Wettkampf aufgeteilt auf zwei Gurppen vergleichen. Nach der ersten Runde werden die Methoden wie auf der nächsten Folie ersichtlich getauscht.


ARBEITSAUFTRAG
▷ Bildet zwei Gruppen
▷ Gruppe 1 berechnet den ggT der Zahlen 216.580 und 57.800 mittels euklidischem Algorithmus
▷ Gruppe2 berechnet den ggT dieser Zahlen mittels mittels der Schulmethode
▶ WELCHE GRUPPE IST SCHNELLER?
216.580 = 57.800 · 3 + 43.180

Arbeitsauftrag



Startet nun euren PC und versucht den Euklidischen Algorithmus mittels Java zu implementieren. Ihr könnt dabei gerne im Team arbeiten!

Arbeitsauftrag

Startet nun euren PC und versucht den Euklidischen Algorithmus mittels Java zu implementieren. Ihr könnt dabei gerne im Team arbeiten!

Der euklidische Algorithmus soll mittels einer Programmiersprache umgesetzt werden. Dabei ist die Programmiersprache entsprechend der Kenntnisse der Schüler/innen zu wählen.
Die Musterlösung auf der folgenden Folie wurde in Java entwickelt. Das grundlegende Prinzip zur Lösung kann mittels jeder beliebigen Sprache realisiert werden.

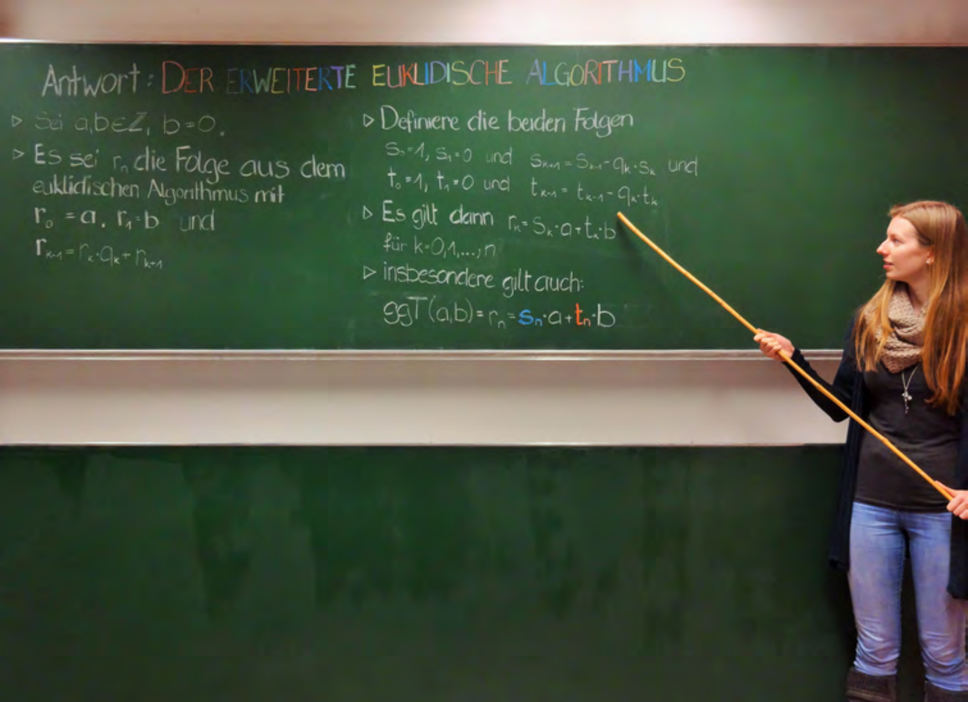
```
public class EuclidianAlgorithm {

   public static void main(String[] args) {

      int orig_a = 927, a = orig_a;
      int orig_b = 351, b = orig_b;
      int r = 0;

      while (b != 0) {

         r = a % b;
         a = b;
         b = r;

      }

      System.out.println("ggt(" + orig_a + ", " + orig_b + ") = " + a);

   }

}
```

```java
public class EuclidianAlgorithm {

   public static void main(String[] args) {

      int orig_a = 927, a = orig_a;
      int orig_b = 351, b = orig_b;
      int r = 0;

      while (b != 0) {

         r = a % b;
         a = b;
         b = r;

      }

      System.out.println("ggt(" + orig_a + ", " + orig_b + ") = " + a);

   }

}
```

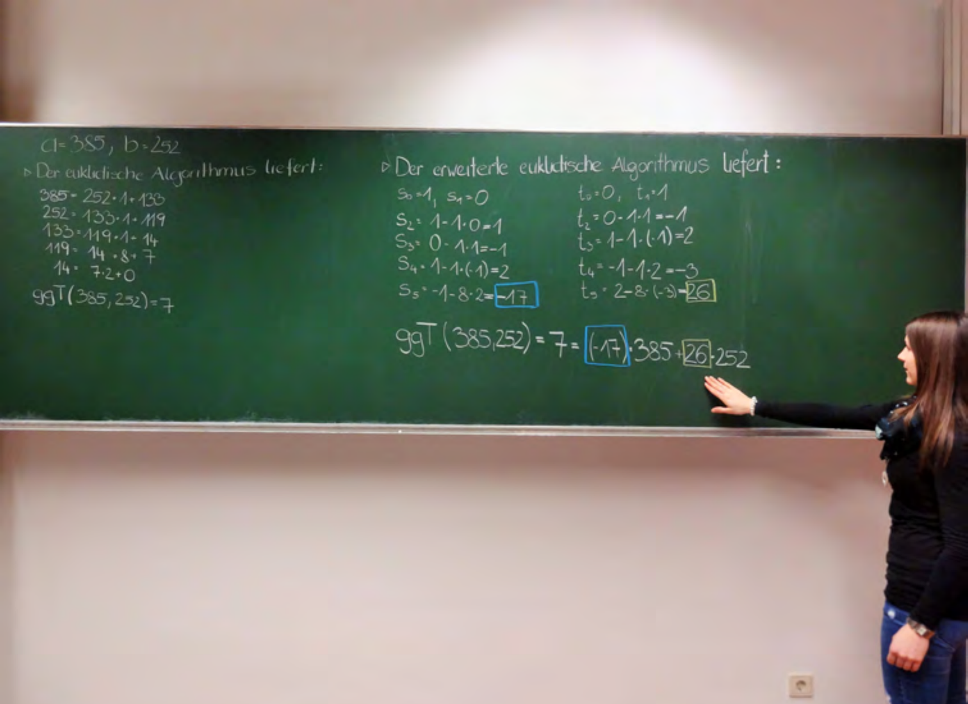
Ergänzend kann auch die “Schulmethode” mittels der selben Programmiersprache umgesetzt werden. Dadurch wird ein interessanter Vergleich der beiden Verfahren möglich. Es können beispielsweise die Laufzeiten der beiden Implementierungen für größere Zahlen gegenübergestellt werden.


LEMMA VON BEZOUT
Der ggT zweier ganzer Zahlen a und b, von denen mind.
eine ungleich 0 ist, lässt sich als Linearkombination von
a und b darstellen.
Also: ggT(a,b) = s·a + t·b, wobei s,t ∈ Z
⇒ Wie berechnet man s und t?

In weiterer Folge soll der erweiterte euklidische Algorithmus thematisiert werden. Die Grundlage dafür bietet das Lemma von Bezout, welches besagt, dass sich der ggT zweier Zahlen als Linearkombination der beiden Zahlen darstellen lässt.


Antwort: DER ERWEITERTE EUKLIDISCHE ALGORITHMUS
▷ Sei a,b∈Z, b≠0.
▷ Es sei r_n die Folge aus dem euklidischen Algorithmus mit
r_0 = a, r_1 = b und
r_{k-1} = r_k·q_k + r_{k+1}
▷ Definiere die beiden Folgen
s_0 = 1, s_1 = 0 und s_{k+1} = s_{k-1} − q_k·s_k und
t_0 = 1, t_1 = 0 und t_{k+1} = t_{k-1} − q_k·t_k
▷ Es gilt dann r_k = s_k·a + t_k·b
für k = 0,1,…,n
▷ insbesondere gilt auch:
ggT(a,b) = r_n = s_n·a + t_n·b

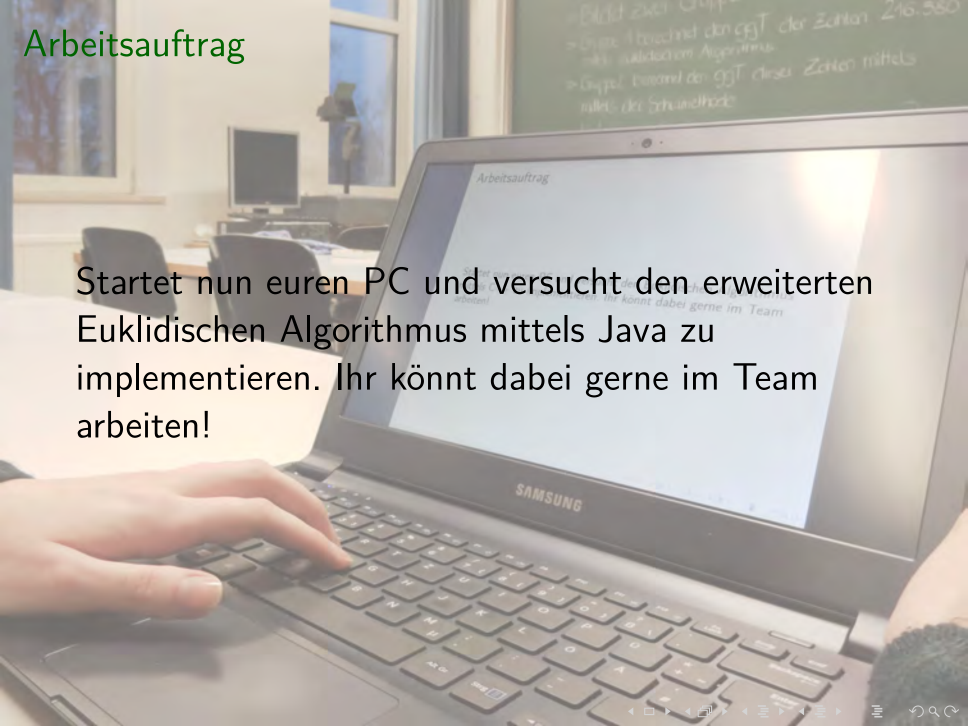
Die Definition des erweiterten euklidischen Algorithmus basiert auf dem Lemma von Bezout, was durch die Farben der Variablen s und t angedeutet wird. Das Beispiel auf der nächsten Folie veranschaulicht die Vorgangsweise.


$a = 385$, $b = 252$
▷ Der euklidische Algorithmus liefert:
$385 = 252 \cdot 1 + 133$
$252 = 133 \cdot 1 + 119$
$133 = 119 \cdot 1 + 14$
$119 = 14 \cdot 8 + 7$
$14 = 7 \cdot 2 + 0$
$ggT(385, 252) = 7$
▷ Der erweiterte euklidische Algorithmus liefert:
$s_0 = 1$, $s_1 = 0$
$s_2 = 1 - 1 \cdot 0 = 1$
$s_3 = 0 - 1 \cdot 1 = -1$
$s_4 = 1 - 1 \cdot (-1) = 2$
$s_5 = -1 - 8 \cdot 2 = \boxed{-17}$
$t_0 = 0$, $t_1 = 1$
$t_2 = 0 - 1 \cdot 1 = -1$
$t_3 = 1 - 1 \cdot (-1) = 2$
$t_4 = -1 - 1 \cdot 2 = -3$
$t_5 = 2 - 8 \cdot (-3) = \boxed{26}$
$ggT(385, 252) = 7 = \boxed{(-17)} \cdot 385 + \boxed{26} \cdot 252$


Übungen:
Stelle den ggT dieser Zahlen als Linearkombination dar:
1. a=189.475, b=15.874
2. a=358, b=479
3. a=15, b=7
1. 189.475 = 15.874

Um die Durchführung des erweiterten euklidischen Algorithmus zu verinnerlichen, sollen die Schüler/innen diesen auf drei konkrete Beispiele anwenden.

Arbeitsauftrag

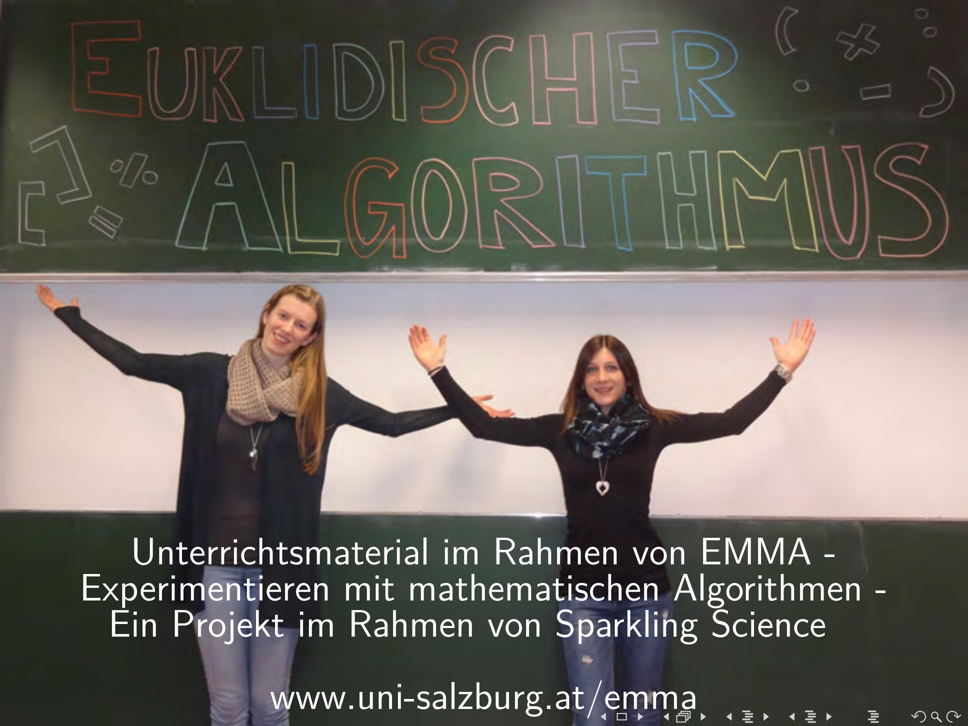


Startet nun euren PC und versucht den erweiterten Euklidischen Algorithmus mittels Java zu implementieren. Ihr könnt dabei gerne im Team arbeiten!

Startet nun euren PC und versucht den erweiterten Euklidischen Algorithmus mittels Java zu implementieren. Ihr könnt dabei gerne im Team arbeiten!

Der erweiterte euklidische Algorithmus soll mittels einer Programmiersprache umgesetzt werden. Dabei ist die Programmiersprache entsprechend der Kenntnisse der Schüler/innen zu wählen.
Die Musterlösung auf der folgenden Folie wurde in Java entwickelt. Das grundlegende Prinzip zur Lösung kann mittels jeder beliebigen Sprache realisiert werden.

```java
public class ExtendedEuclidianAlgorithm {

  public static void main(String[] args) {

    int orig_a = 130900, a = orig_a;
    int orig_b = 33957, b = orig_b;
    int r = 0;
    int q;

    int y_prev = 0, y = 1;
    int x_prev = 1, x = 0;

    int temp = 0;

    while (b != 0) {

      q = a / b;
      r = a % b;
      a = b;
      b = r;

      temp = y_prev;
      y_prev = y;
      y = temp - q * y;

      temp = x_prev;
      x_prev = x;
      x = temp - q * x;

    }

    System.out.println("ggt(" + orig_a + ", " + orig_b + ") = " + a + " = (" + x_prev + ") * " +
                   orig_a + " + (" + y_prev + ") * " + orig_b + " = " + (x_prev * orig_a + y_prev * orig_b));

  }

}
```



Unterrichtsmaterial im Rahmen von EMMA - Experimentieren mit mathematischen Algorithmen - Ein Projekt im Rahmen von Sparkling Science

www.uni-salzburg.at/emma