1.1 Definition. \( a \divides b \) if there exist integers \( a, b \in \mathbb{Z} \) such that \( b = ac \) for some \( c \in \mathbb{Z} \).

1.2 Definition. \( a \divides b \) and \( b \divides a \) if and only if \( a = bc \) and \( b = ac \) for some \( c \in \mathbb{Z} \).

1.3 Definition. For \( a, b, c \in \mathbb{Z} \), if \( a \divides b \), then \( a \divides (b + c) \) and \( a \divides (b - c) \).

1.4 Definition. Let \( a, b \in \mathbb{Z} \). Then \( a \divides b \) if and only if \( b = ad \) for some \( d \in \mathbb{Z} \).

1.5 Definition. The greatest common divisor (GCD) of \( a, b \) is the greatest integer \( d \) such that \( d \divides a \) and \( d \divides b \).

\[ \gcd(a, b) = \max \{ d \in \mathbb{N} : d \divides a \text{ and } d \divides b \} \]

\[ \gcd(a, b) = (a, b) \]

1.6 Definition. \( a, b \) are coprime if \( \gcd(a, b) = 1 \).

1.7 Definition. For \( a, b \in \mathbb{Z} \), the greatest common divisor of \( a \) and \( b \) is given by

\[ \gcd(a, b) = \frac{ab}{\gcd(a, b)} \]
The Euclidean algorithm finds the greatest common divisor (GCD) of two numbers, which is the largest number that divides both of them without leaving a remainder.

1.8 Theorem: Given two integers \( a \) and \( b \), the GCD is found by the following steps:

1. If \( b = 0 \), then \( \text{gcd}(a, b) = a \).
2. Otherwise, \( \text{gcd}(a, b) = \text{gcd}(b, a \mod b) \).

1.9 Proof: We use induction on the number of distinct prime factors of the numbers.

We want to prove that \( d \mid \text{gcd}(a, b) \) if and only if \( d \mid a \) and \( d \mid b \).

Let \( q \) be a common divisor of \( a \) and \( b \). Without loss of generality, let \( d = \text{gcd}(a, b) \) be the greatest common divisor.

We can write \( a = dq_1 \) and \( b = dq_2 \) for some integers \( q_1 \) and \( q_2 \).

Then, \( d \mid a \) and \( d \mid b \), which confirms our claim.

The algorithm is as follows:

```python
def gcd(a, b):
    if b == 0:
        return a
    else:
        return gcd(b, a % b)
```

2.0 Algorithm: Euclid's algorithm for finding the greatest common divisor (GCD) of two numbers:

- **Input:** Two positive integers \( a \) and \( b \).
- **Output:** The greatest common divisor \( \text{gcd}(a, b) \).

The algorithm repeatedly applies the operation of replacing the larger number with the remainder of the division of the larger number by the smaller number until the remainder is zero. The last non-zero remainder is the GCD.

The time complexity of the algorithm is \( O \left( \log \max(a, b) \right) \).
3 Extended GCD

\[ sa + tb = d \quad \text{with} \quad s, t \in \mathbb{Z} \quad \text{and} \quad d = \gcd(a, b) \]

Proof: Consider the output \((s, t)\) of the Euclidean algorithm, where \(\gcd(a, b) = d\) holds.

\[ s \cdot a + t \cdot b = d \]

**Claim:** Given \((a, b)\), we can find integers \(s, t\) such that

\[ s \cdot a + t \cdot b = d \]

**Algorithm:**

```python
# Assumes a >= b
# return value is interpolated as ( s, t )
def extendedGCD( a, b ) :  
    if b == 0 : return (1,0)  
    else :  
        (s, t) = extendedGCD( b, a % b )  
        return (t, s - t * (a // b))  
```

The extended Euclidean algorithm computes \(s, t\) such that

\[ s \cdot a + t \cdot b = d = \gcd(a, b) \]

The algorithm is as follows:

1. If \(b = 0\), return \((1, 0)\).
2. Otherwise, compute \((s', t')\) from \((b, a \mod b)\) recursively.
3. Return \((t, s - t \cdot (a \div b))\).

Using this algorithm,

\[ s' \cdot a + t' \cdot (a \mod b) = d \]

We have

\[ s' = s' \cdot a + t' \cdot (a \mod b) \]

and

\[ t' = t' \cdot a + (s' - t' \cdot (a \div b)) \cdot b \]

3.1 Theorem

The algorithm computes unique integers \(s, t\) such that

\[ s \cdot a + t \cdot b = d = \gcd(a, b) \]

Note: For any integers \(a, b\), the existence of such integers \(s, t\) is guaranteed.

The algorithm computes unique integers \(s, t\) such that

\[ s \cdot a + t \cdot b = d = \gcd(a, b) \]

Note: For any integers \(a, b\), the existence of such integers \(s, t\) is guaranteed.