image.png
PNG 70.46KB

list all the properties of gcd all the properties
1. all properties used in cp/dsa
if a,b
gcd(a,b)= greatest common factor of a,b
2. all the common and basic and medium level properties used on cf/lc and mathematics
3. all uses and properties like euclidean steP properties list them all
gcd(a,b)==gcd(a,a%b) a<b a and b are coprime
give like i can take save it as pdf print and also give equivalent cpp code for relevent things

All Properties of GCD (Greatest Common Divisor)

1. Basic Definition & Elementary Properties

```text
gcd(a,b) = largest positive integer d such that d|a and d|b
```

Properties:

1. **Non-negative**: gcd(a,b) ≥ 0
2. **Identity**: gcd(a,0) = |a|
3. **Zero case**: gcd(0,0) is undefined (usually treated as 0)
4. **Commutativity**: gcd(a,b) = gcd(b,a)
5. **Absorption**: gcd(a,a) = |a|
6. **gcd(1,a) = 1** for all integers a
7. **gcd(a,b) = gcd(|a|,|b|)** (absolute values)

2. Euclidean Algorithm Properties

Core Recursive Property:

```text
```

```
gcd(a,b) = gcd(b, a mod b)   (for b ≠ 0)
gcd(a,b) = gcd(a-b,b)        (alternative form)
```

Extended Euclidean Algorithm:

For any integers a,b, ∃ integers x,y such that:

```text
ax + by = gcd(a,b)
```

(Bézout's Identity)

3. Algebraic Properties

Distributive Properties:

1. **gcd(ka,kb) = |k|·gcd(a,b)** (Homogeneity)
2. **gcd(a,b) = gcd(a, b+ka)** for any integer k
3. **gcd(a,b) = gcd(a, a+b)** (and similar combinations)

Multi-argument GCD:

1. **gcd(a,b,c) = gcd(gcd(a,b),c)** (Associative-like property)
2. **$\gcd(a_1,a_2,\ldots,a_n) = \gcd(\gcd(a_1,\ldots,a_{n-1}),a_n)$**

LCM Relationship:

```text
a·b = gcd(a,b)·lcm(a,b)   (for a,b > 0)
```

Therefore: `lcm(a,b) = a·b / gcd(a,b)`

4. Number Theory Properties

Coprime (Relatively Prime) Properties:

1. **gcd(a,b)=1** ⇔ a and b are coprime
2. If gcd(a,b)=1 and a|bc, then a|c
3. If gcd(a,b)=1, then gcd(a,bc)=gcd(a,c)
4. If gcd(a,b)=1 and gcd(a,c)=1, then gcd(a,bc)=1

Division Properties:

1. If d|a and d|b, then d|gcd(a,b)
2. If gcd(a,b)=d, then gcd(a/d, b/d)=1
3. If a|bc and gcd(a,b)=1, then a|c

Prime Factorization:

```text
If a = ∏ p_i^{α_i}, b = ∏ p_i^{β_i}, then:
gcd(a,b) = ∏ p_i^{min(α_i,β_i)}
```

5. Important Theorems & Corollaries

Euclid's Lemma:

If p is prime and p|ab, then p|a or p|b

Chinese Remainder Theorem (CRT) Condition:

System x ≡ a (mod m), x ≡ b (mod n) has solution iff gcd(m,n) | (a-b)

Linear Diophantine Equations:

ax + by = c has integer solutions iff gcd(a,b) | c

6. Computational Properties for CP/DSA

Complexity:

- Euclidean Algorithm: O(log(min(a,b)))

- Extended Euclidean: O(log(min(a,b)))

Useful Facts:

1. **gcd(a,b) = gcd(a-b,b)** (used in binary GCD)
2. **gcd(2a,2b) = 2·gcd(a,b)**
3. **gcd(2a,b) = gcd(a,b)** if b is odd
4. For Fibonacci numbers: gcd(F_m, F_n) = F_{gcd(m,n)}

Binary GCD (Stein's Algorithm) Properties:

1. gcd(0,v)=v
2. gcd(2u,2v)=2·gcd(u,v)
3. gcd(2u,v)=gcd(u,v) if v is odd
4. gcd(u,v)=gcd(|u-v|, min(u,v))

7. Special Cases & Patterns

Consecutive Numbers:

```text
gcd(n, n+1) = 1
gcd(n, n+k) divides k
```

Powers:

```text
gcd(a^m - 1, a^n - 1) = a^{gcd(m,n)} - 1
gcd(a^m + b^m, a^n + b^n) = a^{gcd(m,n)} + b^{gcd(m,n)} for odd m,n
```

Linear Combinations:

```text
gcd(a,b) = gcd(a, b±a) = gcd(a, b±ka) for any k
```

8. C++ Implementations

cpp

```cpp
// Basic Euclidean Algorithm (Recursive)
int gcd(int a, int b) {
    return b == 0 ? a : gcd(b, a % b);
}

// Basic Euclidean Algorithm (Iterative)
int gcd_iterative(int a, int b) {
    while (b) {
        a %= b;
        swap(a, b);
    }
    return a;
}

// Extended Euclidean Algorithm
// Returns gcd(a,b) and finds x,y such that ax + by = gcd(a,b)
int extended_gcd(int a, int b, int &x, int &y) {
    if (b == 0) {
        x = 1;
        y = 0;
        return a;
    }
    int x1, y1;
    int d = extended_gcd(b, a % b, x1, y1);
    x = y1;
    y = x1 - y1 * (a / b);
    return d;
}

// Binary GCD (Stein's Algorithm) - efficient for large numbers
int binary_gcd(int a, int b) {
    if (a == 0) return b;
    if (b == 0) return a;

    int shift = __builtin_ctz(a | b);
    a >>= __builtin_ctz(a);

    do {
        b >>= __builtin_ctz(b);
        if (a > b) swap(a, b);
        b -= a;
    } while (b != 0);
```

```
    return a << shift;
}

// LCM using GCD
int lcm(int a, int b) {
    return a / gcd(a, b) * b;  // Multiply after division to avoid overflow
}

// GCD for array of numbers
int gcd_array(vector<int>& arr) {
    int result = arr[0];
    for (int i = 1; i < arr.size(); i++) {
        result = gcd(result, arr[i]);
        if (result == 1) return 1;
    }
    return result;
}

// Check if numbers are pairwise coprime
bool are_pairwise_coprime(vector<int>& arr) {
    for (int i = 0; i < arr.size(); i++) {
        for (int j = i + 1; j < arr.size(); j++) {
            if (gcd(arr[i], arr[j]) != 1) return false;
        }
    }
    return true;
}
```

9. Common Problems & Applications

Frequently Used in:

1. **Simplifying fractions**: a/b = (a/gcd(a,b)) / (b/gcd(a,b))
2. **Modular inverses**: a^{-1} mod m exists iff gcd(a,m)=1
3. **Cipher algorithms** (RSA)
4. **Diophantine equations**
5. **Number of reduced fractions** with denominator ≤ n
6. **Count coprime pairs** ≤ n

Optimization Properties:

- **gcd(a,b)** divides any linear combination of a and b
- If d = gcd(a,b), then ∃ integers x,y with $|x| \leq |b/d|$, $|y| \leq |a/d|$
- For random numbers, P(gcd(a,b)=1) ≈ $6/\pi^2$ ≈ 0.6079

10. Important Inequalities & Bounds

1. **gcd(a,b) ≤ min(|a|,|b|)**
2. **gcd(a,b) · lcm(a,b) = |a·b|**
3. For a,b > 0: gcd(a,b) · lcm(a,b) = a·b
4. **gcd(a,b) ≥ 1** for a,b ≠ 0

Save this as a comprehensive reference for GCD properties! This covers everything from basic definitions to advanced competitive programming applications with ready-to-use C++ code.