
# All-in at the River

## Standard Code Library

Shanghai Jiao Tong University

Desprado2 fstqwq AntiLeaf

# 目录

# 1 数学

## 1.1 FWT

矩阵表示

or 形式 (子集卷积):

$$T_{ij} = [i|j = i] = [j \in i]$$

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

and 形式 (超集卷积):

$$T_{ij} = [i\&j = i] = [i \in j]$$

$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

xor 形式 ($T$与自身互为逆矩阵):

$$T_{ij} = (-1)^{parity(i\&j)}$$

$$\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}$$

```cpp
using ll = long long;

// or
void FWT(ll *a, int len, int inv) {
    for (int h = 1; h < len; h <<= 1) {
        for (int i = 0; i < len; i += (h << 1)) {
            for (int j = 0; j < h; ++j) {
                a[i + j + h] += a[i + j] * inv;
            }
        }
    }
}
// and
void FWT(ll *a, int len, int inv) {
    for (int h = 1; h < len; h <<= 1) {
        for (int i = 0; i < len; i += (h << 1)) {
            for (int j = 0; j < h; ++j) {
                a[i + j] += a[i + j + h] * inv;
            }
        }
    }
}
// xor
void FWT(ll *a, int len, int inv) {
    for (int h = 1; h < len; h <<= 1) {
        for (int i = 0; i < len; i += (h << 1)) {
            for (int j = 0; j < h; ++j) {
                ll x = a[i + j], y = a[i + j + h];
                a[i + j] = x + y, a[i + j + h] = x - y;
                if (inv == -1)
                    a[i + j] /= 2, a[i + j + h] /= 2;
            }
        }
    }
}
```

## 1.2 多项式

### 1.2.1 FFT

```cpp
// 使用时一定要注意double的精度是否足够(极限大概是10 ^ 14)

const double pi = acos((double)-1.0);

// 手写复数类
// 支持加减乘三种运算
// += 运算符如果用的不多可以不重载
struct Complex {
    double a, b; // 由于long double精度和double几乎相同，通常没有必要用long double

    Complex(double a = 0.0, double b = 0.0) : a(a), b(b) {}

    Complex operator + (const Complex &x) const {
        return Complex(a + x.a, b + x.b);
    }

    Complex operator - (const Complex &x) const {
        return Complex(a - x.a, b - x.b);
    }

    Complex operator * (const Complex &x) const {
        return Complex(a * x.a - b * x.b, a * x.b + b * x.a);
    }

    Complex operator * (double x) const {
        return Complex(a * x, b * x);
    }

    Complex &operator += (const Complex &x) {
        return *this = *this + x;
    }

    Complex conj() const { // 共轭，一般只有MTT需要用
        return Complex(a, -b);
    }
} omega[maxn], omega_inv[maxn];
const Complex ima = Complex(0, 1);

int fft_n; // 要在主函数里初始化

// FFT初始化
void FFT_init(int n) {
    fft_n = n;

    for (int i = 0; i < n; i++) // 根据单位根的旋转性质可以节省计算单位根逆元的时间
        omega[i] = Complex(cos(2 * pi / n * i), sin(2 * pi / n * i));

    omega_inv[0] = omega[0];
    for (int i = 1; i < n; i++)
        omega_inv[i] = omega[n - i];
    // 当然不存单位根也可以，只不过在FFT次数较多时很可能会增大常数
}

// FFT主过程
void FFT(Complex *a, int n, int tp) {
    for (int i = 1, j = 0, k; i < n - 1; i++) {
        k = n;
        do
            j ^= (k >>= 1);
        while (j < k);

        if (i < j)
            swap(a[i], a[j]);
    }

    for (int k = 2, m = fft_n / 2; k <= n; k *= 2, m /= 2)
        for (int i = 0; i < n; i += k)
            for (int j = 0; j < k / 2; j++) {
```

```cpp
                Complex u = a[i + j], v = (tp > 0 ? omega : omega_inv)[m * j] * a[i + j + k / 2];

                a[i + j] = u + v;
                a[i + j + k / 2] = u - v;
            }

    if (tp < 0)
        for (int i = 0; i < n; i++) {
            a[i].a /= n;
            a[i].b /= n; // 一般情况下是不需要的, 只有MTT时才需要
        }
}
```

## 1.3 组合数行区间和

```cpp
using ll = long long;

// 边界从 (s, x) 移动到 (s + 1, nx)
ll move(int x, int nx, int s, ll sum) {
    assert(x >= -1);
    ll res = (2 * sum + mod - C(s, x)) % mod;
    while (x + 1 <= nx) {
        x++;
        res = (res + C(s + 1, x)) % mod;
    }
    while (x > nx) {
        res = (res + mod - C(s + 1, x)) % mod;
        x--;
    }
    return res;
};

void proc(int k) {
    // 第 s 行的 左右边界
    auto le = [&](int s) -> int {
        // ...
    };
    auto ri = [&](int s) -> int {
        // ...
    };
    // 这里应该暴力计算首行和□当首行为 0 时也可以这样写
    ll lsum = move(-1, le(0), -1, 0);
    ll rsum = move(-1, ri(0), -1, 0);

    for (int s = 0; s <= k; s++) {
        if (le(s) < ri(s)) {
            // ... 第 s 行对答案的贡献
        }
        lsum = move(le(s), le(s + 1), s, lsum);
        rsum = move(ri(s), ri(s + 1), s, rsum);
    }
}
```

## 1.4 二项式反演

形式一:

$$f(n)=\sum_{i=m}^{n}\binom{n}{i} \iff g(i)g(n)=\sum_{i=m}^{n}(-1)^{n-i}\binom{n}{i}f(i)$$

形式二: (常用)

$$f(n)=\sum_{i=n}^{m}\binom{i}{n}g(i) \iff g(n)=\sum_{i=n}^{m}(-1)^{i-n}\binom{i}{n}f(i)$$

**常见用法**

钦定 (至少) $k$ 个与恰好 $k$ 个之间的转化

记号:

记 $f(n)$ 表示先钦定至少选 $n$ 个, 再统计钦定情况如此的方案数之和, 其中会包含重复的方案数.

记 $g(n)$ 表示恰好选 $n$ 个的方案数, 不会重复.

那么, 对于 $i \geq n$, $g(i)$ 在 $f(n)$ 中被重复计算了 $\binom{i}{n}$ 次, 故 $f(n)=\sum_{i=n}^{m}\binom{i}{n}g(i)$, 其中 $m$ 为数目上限

通常, $f$ 较易求, 再通过反演即可求 $g$

# 2 数论

## 2.1 常见预处理与快速幂

```cpp
// 预处理组合数
const int N = 2e5 + 7;
const ll mod = 998244353;

ll fac[N], ifac[N];
void init() {
    fac[0] = 1;
    for (int i = 1; i < N; ++i) fac[i] = i * fac[i-1] % mod;
    ifac[N - 1] = fpow(fac[N - 1], mod - 2);
    for (int i = N - 1; i; --i) ifac[i - 1] = i * ifac[i] % mod;
}

ll C(int n, int k) {
    if (k < 0 || k > n) return 0;
    return (fac[n] * ifac[k] % mod) * ifac[n - k] % mod;
}

// 线性求逆元，注意有效的 i < mod
ll inv[maxn];
void init() {
    inv[0] = 0, inv[1] = 1;
    for (int i = 2; i < N; ++i)
        inv[i] = inv[mod % i] * (mod - mod / i) % mod;
}

// 快速幂
ll fpow(ll a, ll k = mod - 2, ll p = mod) {
    ll res = 1; a %= p;
    for (; k; k >>= 1, a = a * a % p) {
        if (k & 1)
            res = res * a % p;
    }
    return res;
}
```

## 2.2 因数分解与素性判定

### 2.2.1 朴素因数分解

```cpp
// 素因数分解
int p[maxn], l[maxn], cnt2 = 0;
void Fact(int x) {
    cnt2 = 0;
    for (int i = 2; 1ll * i * i <= x; i++) {
        if(x % i == 0) {
            p[++cnt2] = i; l[cnt2] = 0;
            while(x % i == 0) {
                x /= i; ++l[cnt2];
            }
        }
    }
    if (x != 1) {  // 则此时x一定是素数，且为原本x的大于根号x的唯一素因子
        p[++cnt2] = x; l[cnt2] = 1;
    }
}

// vector ver. 无次数
void Fact(ll x, vector<int>& fact) {
    for (ll i = 2; 1ll * i * i <= x; ++i) {
        if(x % i == 0) {
            fact.push_back(i);
            while(x % i == 0) x /= i;
        }
    }
    if (x != 1) fact.push_back(x);
}
```

### 2.2.2 Miller-Rabin 与 Pollard-Rho

```cpp
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;

ll randint(ll l, ll r) {
    static mt19937 eng(time(0));
    uniform_int_distribution<ll> dis(l, r);
    return dis(eng);
}

bool is_prime(ll x) {
    int s = 0; ll t = x - 1;
    if (x == 2) return true;
    if (x < 2 || !(x & 1)) return false;
    while (!(t & 1)) {   //将x分解成(2^s)*t的样子
        s++; t >>= 1;
    }
    ll lst[] = {2, 325, 9375, 28178, 450775, 9780504, 1795265022};
    for(ll a : lst) { //随便选一个素数进行测试
        if(a >= x) break;
        ll b = Pow(a, t, x); //先算出a^t
        for (int j = 1; j <= s; ++j) { //然后进行s次平方
            ll k = mul(b, b, x);        //求b的平方
            if (k == 1 && b != 1 && b != x - 1) //用二次探测判断
                return false;
            b = k;
        }
        if (b != 1)
            return false; //用费马小定律判断
    }
    return true; //如果进行多次测试都是对的，那么x就很有可能是素数
}

ll gcd(ll a, ll b) { return b == 0 ? a : gcd(b, a % b); }

// @author: Pecco
ll Pollard_Rho(ll n) {
    if (n == 4) return 2;
    if (is_prime(n)) return n;
    while (1) {
        ll c = randint(1, n - 1); // 生成随机的c
        auto f = [=](ll x) { return ((__int128)x * x + c) % n; }; // lll表示__int128，防溢出
        ll t = f(0), r = f(f(0));
        while (t != r) {
            ll d = __gcd(abs(t - r), n);
            if (d > 1)
                return d;
            t = f(t), r = f(f(r));
        }
    }
}


// 优化掉一个log
ll Pollard_Rho(ll n) {
    if (n == 4) return 2;
    if (is_prime(n)) return n;
    while (1) {
        ll c = randint(1, n - 1);
        auto f = [=](ll x) { return ((__int128)x * x + c) % n; };
```

```cpp
        ll t = 0, r = 0, p = 1, q;
        do {
            for (int i = 0; i < 128; ++i) { // 令固定距离C=128
                t = f(t), r = f(f(r));
                if (t == r || (q = (lll)p * abs(t - r) % n) == 0) // 如果发现环□或者积即将为0□退出
                    break;
                p = q;
            }
            ll d = gcd(p, n);
            if (d > 1)
                return d;
        } while (t != r);
    }
}

vector<ll> factors;

void getfactors(ll n) {
    if (n == 1) return;
    if (is_prime(n)) { factors.push_back(n); return; } // 如果是质因子
    ll p = n;
    while (p == n)
        p = Pollard_Rho(n);
    getfactors(n / p), getfactors(p); //递归处理
}
```

## 2.3 筛法

### 2.3.1 线性筛

```cpp
const int maxn = 1000000 + 5;
bool isnt[maxn];
int prime[maxn];
int cnt = 0;

// 线性筛法 [1, n] 内素数
void Prime(int n) {
    isnt[1] = true;
    cnt = 0;
    for (int i = 2; i <= n; i++) {
        if (!isnt[i]) prime[++cnt] = i;
        for (int j = 1; j <= cnt; j++) {
            if (1ll * i * prime[j] > n) break;
            isnt[i * prime[j]] = 1;
            if (i % prime[j] == 0) break;
        }
    }
}

// 线性筛求积性函数
int phi[maxn], mu[maxn], d[maxn], D[maxn], q[maxn];
void Sieve(int n) {
    isnt[1] = true;
    phi[1] = 1;
    //mu[1] = 1;
    cnt = 0;
    for(int i = 2; i <= n; i++) {
        if (!isnt[i]) {
            prime[++cnt] = i;
            phi[i] = i - 1;
            //mu[i] = -1;
            // d[i] = 2; q[i] = 1;
            // D[i] = i + 1; q[i] = 1;
        }
        for (int j = 1; j <= cnt; j++) {
            int x = i * prime[j];
            if (x > n) break;
            isnt[x] = 1;
            if (i % prime[j] == 0) {
                phi[x] = phi[i] * prime[j];
                // mu[x] = 0;
                // d[x] = d[i] / (q[i] + 1) * (q[i] + 2), q[x] = q[i] + 1;
                // D[x] = D[i] / (prime[j] ^ (q[i] + 1) - 1) * (prime[j] ^ (q[i] + 2) - 1), q[x] = q[i] + 1;
                break;
            } else {
                phi[x] = phi[i] * (prime[j] - 1); // mu[x] = -mu[i]
                // d[x] = 2 * d[i], q[x] = 1;
                // D[x] = (prime[j] + 1) * D[i], q[x] = 1;
            }
        }
    }
}
```

### 2.3.2 Min25 筛

```cpp
using ll = long long;
using i128 = __int128;
//using i128 = int64_t;

const ll mod = 998244353;

namespace min25 {
    const int N = 1e6 + 10;
    ll po[40][N];
    inline ll fpow(ll e, ll k) {
        return po[e][k];
    }
    void precalc() {
        for(int e = 0; e < 40; ++e) {
            po[e][0] = 1;
            for(int i = 1; i < N; i++)
                po[e][i] = e * po[e][i - 1] % mod;
        }
    }
    ll n;
    int B;
    int _id[N * 2];
    inline int id(ll x) {
        return x <= B ? x : n / x + B;
    }
    inline int Id(ll x) {
        return _id[id(x)];
    }
    // f(p) = p - 1 = fh(p) - fg(p);
    inline ll fg(ll x) {
        // assert(x <= sqrt(n));
        return 1;
    }
    inline ll fh(ll x) {
        // assert(x <= sqrt(n));
        return x;
    }
    // \sum_{i=2}^n fg(i)
    inline ll sg(ll x) {
        return (x - 1) % mod;
    }
    // \sum_{i=2}^n fh(i)
    inline ll sh(ll x) {
        return ((i128) x * (x + 1) / 2 + mod - 1) % mod;
    }
    // f(p^e)
    inline ll f(ll p, ll e) {
        //return (pe - pe / p) % mod;
        return (p + (mod - 2) * fpow(e, p)) % mod;
    }
```

```cpp
    bitset<N> np;
    ll p[N>>2], pn;
    ll pg[N>>2], ph[N>>2];
    void sieve(ll sz) {
        for(int i = 2; i <= sz; i++) {
            if(!np[i]) {
                p[++pn] = i;
                pg[pn] = (pg[pn - 1] + fg(i)) % mod;
                ph[pn] = (ph[pn - 1] + fh(i)) % mod;
            }
            for(int j = 1; j <= pn && i * p[j] <= sz; j++) {
                np[i * p[j]] = 1;
                if(i % p[j] == 0) {
                    break;
                }
            }
        }
    }
    ll m;
    ll g[N * 2], h[N * 2];
    ll w[N * 2];

    void compress() {
        for (int i = 1; i <= m; i++) {
            g[i] = (h[i] + mod - g[i] + mod - g[i]) % mod;
        }
        for (int i = 1; i <= pn; i++) {
            pg[i] = (ph[i] + mod - pg[i] + mod - pg[i]) % mod;
        }
    }

    ll dfs_F(int k, ll n) {
        if (n < p[k] || n <= 1) return 0;
        ll res = g[Id(n)] + mod - pg[k - 1], pw2;
        for (int i = k; i <= pn && (pw2 = (ll) p[i] * p[i]) <= n; ++i) {
            ll pw = p[i];
            for (int c = 1; pw2 <= n; ++c, pw = pw2, pw2 *= p[i])
                res = (res + ((ll) f(p[i], c) * dfs_F(i + 1, n / pw) + f(p[i], c + 1))) % mod;
        }
        return res;
    }

    void init(ll _n) {
        n = _n;
        B = sqrt(n) + 100;
        pn = 0;
        sieve(B);
        m = 0;
        for(ll i = 1, j; i <= n; i = j + 1) {
            j = n / (n / i);
            ll t = n / i;
            _id[id(t)] = ++m;
            w[m] = t;
            g[m] = sg(t);
            h[m] = sh(t);
            //printf("id: %lld, w: %lld, g: %lld, h: %lld\n", m, t, g[m], h[m]);
        }
        for (int j = 1; j <= pn; j++) {
            ll z = (ll) p[j] * p[j];
            for(int i = 1; i <= m && z <= w[i]; i++) {
                int k = Id(w[i] / p[j]);
                g[i] = (g[i] + (ll) (mod - fg(p[j])) * (g[k] - pg[j - 1] + mod)) % mod;
                h[i] = (h[i] + (ll) (mod - fh(p[j])) * (h[k] - ph[j - 1] + mod)) % mod;
            }
        }
        compress();

        /* 递推 min25
        for(int j = pn; j > 0; j--) {
            ll z = (ll) p[j] * p[j];
            for(int i = 1; i <= m && z <= w[i]; i++) {
                ll pe = p[j];
                for(int e = 1; pe * p[j] <= w[i]; e++, pe *= p[j]) {
                    g[i] = (g[i] + (ll) f(p[j], e) * (g[Id(w[i] / pe)] - pg[j] + mod) + f(p[j], e + 1)) % mod;
                }
            }
        } */
    }
    ll get(ll x) { // x == n / i
        if(x < 1) return 0;
        return (dfs_F(1, x) + 1) % mod;
    }
    ll get(ll l, ll r) {
        return get(r) - get(l - 1);
    }
}

void Solve() {
    long long n;
    scanf("%lld", &n);
    min25::init(n);
    long long res = min25::get(n);
    printf("%lld\n", res);
}

int main() {
    min25::precalc();
    int T;
    scanf("%d", &T);
    while(T--) {
        Solve();
    }
}
```

## 2.4 扩展欧几里得

```cpp
using i128 = __int128;

// ax + by = c
// 有解当且仅当 gcd(a, b) | c
// 要求 a, b 不全为 0
// 无合法性检查
void exgcd(i128 a, i128 b, i128 &x, i128 &y, i128 c = 1) {
    if (b == 0) {
        x = c / a;
        y = 0;
    } else {
        exgcd(b, a % b, x, y, c);
        i128 tmp = x;
        x = y;
        y = tmp - (a / b) * y;
    }
}
```

## 2.5 中国剩余定理

### 2.5.1 两个数的 crt

```cpp
// mul 表示慢速乘
ll crt(ll a1, ll p, ll a2, ll q) {
    ll ip = fpow(p, q-2, q);
    ll iq = fpow(q, p-2, p);
    ll n = p * q;
    return (mul(mul(a1, q, n), iq, n) + mul(mul(a2, p, n), ip, n)) % n;
}
```

### 2.5.2 excrt

```cpp
using ll = long long;

ll gcd(ll a, ll b) {
    return b == 0 ? a : gcd(b, a % b);
}

// x === a1 (mod b1), x === a2 (mod b2)
// 合法性检查□返回 -1 则为无解
pair<ll, ll> excrt(ll a1, ll b1, ll a2, ll b2) {
    ll g = gcd(b1, b2);
    ll lcm = (b1 / g) * b2;

    if ((a1 - a2) % g) return {-1, -1};

    i128 x, y;
    exgcd(b1, b2, x, y, a1 - a2);
    ll res = (a1 - b1 * x) % lcm;
    if (res < 0) res += lcm;
    return {res, lcm};
}
```

## 2.6 卢卡斯定理

### 2.6.1 模素数卢卡斯

```cpp
// 卢卡斯定理，要求 p 为素数
ll lucas(ll n, ll m, ll p) {
    if (!m) return 1;
    return C(n % p, m % p, p) * lucas(n / p, m / p, p) % p;
}
```

### 2.6.2 扩展卢卡斯

```cpp
// 扩展卢卡斯定理

// 扩欧求逆元
ll INV(ll a, ll p) {
    ll x, y;
    exgcd(a, p, x, y);
    return (x % p + p) % p;
}

// 递归求解(n! / px) mod pk
ll F(ll n, ll p, ll pk) {
    if (n == 0) return 1;
    ll rou = 1; // 循环节
    ll rem = 1; // 余项
    for (ll i = 1; i <= pk; ++i) {
        if (i % p)
            rou = rou * i % pk;
    }
    rou = fpow(rou, n / pk, pk);
    for (ll i = pk * (n / pk); i <= n; ++i) {
        if (i % p)
            rem = rem * (i % pk) % pk; // 小心i炸int
    }
    return F(n / p, p, pk) * rou % pk * rem % pk;
}

// 素数p在n!中的次数
ll G(ll n, ll p) {
    if (n < p) return 0;
    return G(n / p, p) + (n / p);
}

ll C_pk(ll n, ll m, ll p, ll pk) {
    ll fz = F(n, p, pk), fm1 = INV(F(m, p, pk), pk),
       fm2 = INV(F(n - m, p, pk), pk);
    ll mi = fpow(p, G(n, p) - G(m, p) - G(n - m, p), pk);
    return fz * fm1 % pk * fm2 % pk * mi % pk;
}

ll exlucas(ll n, ll m, ll P) {
    Fact(P); // 素因子分解□见素因子分解.cpp
    for (int i = 1; i <= cnt2; ++i) {
        ll pk = 1;
        for (int j = 0; j < l[i]; ++j) {
            pk *= p[i];
        }
        bi[i] = pk, ai[i] = C_pk(n, m, p[i], pk);
    }
    return excrt(cnt2) % P;
}
```

## 2.7 原根与离散对数

### 2.7.1 原根

```cpp
// 得到 p 的原根
ll generator(ll p) {
    static ll rec, ans;
    if (p == rec)
        return ans;
    rec = p;
    vector<ll> fact;
    ll phi = p - 1, n = phi;
    for (ll i = 2; 1ll * i * i <= n; ++i) {
        if (n % i == 0) {
            fact.push_back(i);
            while (n % i == 0)
                n /= i;
        }
    }
    if (n > 1)
        fact.push_back(n);
    for (ll res = 2; res <= p; ++res) {
        bool ok = 1;
        for (ll factor : fact) {
            if (fpow(res, phi / factor, p) == 1) {
                ok = false;
                break;
            }
        }
        if (ok)
            return ans = res;
    }
    return ans = -1;
}
```

### 2.7.2 BSGS

```cpp
// a ^ k == b mod p
ll BSGS(ll a, ll b, ll p) { // p <= 1e9
    static ll rec;
    static map<ll, ll> mp;
```

```cpp

    ll sq = (ll)ceil(sqrt(p));
    if (rec != p) {
        rec = p;
        mp.clear();
        ll le = 1, bs = fpow(a, sq, p);
        for (ll i = 1; i <= sq; ++i) {
            le = le * bs % p;
            if (le < 0)
                le += p;
            mp[le] = i * sq;
        }
    }

    ll ri = (b % p);
    if (ri < 0)
        ri += p;
    for (ll j = 0; j <= sq; ++j) {
        if (mp.count(ri)) {
            return mp[ri] - j;
        }
        ri = ri * a % p;
        if (ri < 0)
            ri += p;
    }
    return -1;
}

// x ^ a == b mod p
ll calc(ll a, ll b, ll p) {
    ll g = generator(p); // 求原根-见原根.cpp
    ll ga = fpow(g, a, p);
    ll c = BSGS(ga, b, p);
    ll res = fpow(g, c, p);
    return res;
}
```

## 2.8 杂项

### 2.8.1 大数整除小数取模

计算 $\frac{a}{b} \bmod p$
当 a 的本值太大无法表示时 可以计算 a 对 b * p 取模的结果 再除 b 模 p

$$\frac{a}{b} \bmod p = \frac{a \bmod b * p}{b} \bmod p$$

### 2.8.2 立方根复杂度求 mobius 函数

```cpp
int getmu(int x) {
    int pr, cur = 0;
    for (int i = 1; i <= cnt; ++i) {
        cur = 0;
        while (x % prime[i] == 0) {
            ++cur; x /= prime[i];
        }
        if (cur > 1) return 0;
    }
    if (x == 1) return 1;
    int sq = sqrt(x) + 0.5;
    if (1ll * sq * sq == x) return 0;
    return 1;
}
```

### 2.8.3 直接求 euler 函数

```cpp
// primes 为预处理的素数表
ll getPhi(ll x) {
    ll phi = x, num = x;
    for (int p : primes) {
        if (p > x / p) break;
        if (x % p == 0)
            phi = (phi / p) * (p - 1), x /= p;
        while (x % p == 0)
            x /= p;
    }
    if (x > 1)
        phi = phi / x * (x - 1);
    return phi;
}
```

# 3 图论

## 3.1 最小生成树

### 3.1.1 Boruvka算法

思想：每次选择连接每个连通块的最小边，把连通块缩起来.
每次连通块个数至少减半，所以迭代$O(\log n)$次即可得到最小生成树.
一种比较简单的实现方法：每次迭代遍历所有边，用并查集维护连通性和每个连通块的最小边权.
应用：最小异或生成树

### 3.1.2 动态最小生成树

动态最小生成树的离线算法比较容易，而在线算法通常极为复杂.
一个跑得比较快的离线做法是对时间分治，在每层分治时找出一定在/不在MST上的边，只带着不确定边继续递归.
简单起见，找确定边的过程用Kruskal算法实现，过程中的两种重要操作如下：

- Reduction：待修改边标为+INF，跑MST后把非树边删掉，减少无用边
- Contraction：待修改边标为-INF，跑MST后缩除待修改边之外的所有MST边，计算必须边

每轮分治需要Reduction-Contraction，借此减少不确定边，从而保证复杂度.
复杂度证明：假设当前区间有$k$条待修改边，$n$和$m$表示点数和边数，那么最坏情况下R-C的效果为$(n, m) \to (n, n+k-1) \to (k+1, 2k)$.

```cpp
// 全局结构体与数组定义
struct edge { //边的定义
    int u, v, w, id; // id表示边在原图中的编号
    bool vis; // 在Kruskal时用,记录这条边是否是树边
    bool operator < (const edge &e) const { return w < e.w; }
} e[20][maxn], t[maxn]; // 为了便于回滚,在每层分治存一个副本


// 用于存储修改的结构体,表示第id条边的权值从u修改为v
struct A {
    int id, u, v;
} a[maxn];


int id[20][maxn]; // 每条边在当前图中的编号
int p[maxn], size[maxn], stk[maxn], top; // p和size是并查集数组,stk是用来撤销的栈
int n, m, q; // 点数,边数,修改数


// 方便起见,附上可能需要用到的预处理代码
for (int i = 1; i <= n; i++) { // 并查集初始化
    p[i] = i;
    size[i] = 1;
}

for (int i = 1; i <= m; i++) { // 读入与预标号
    scanf("%d%d%d", &e[0][i].u, &e[0][i].v, &e[0][i].w);
    e[0][i].id = i;
    id[0][i] = i;
}

for (int i = 1; i <= q; i++) { // 预处理出调用数组
    scanf("%d%d", &a[i].id, &a[i].v);
    a[i].u = e[0][a[i].id].w;
    e[0][a[i].id].w = a[i].v;
}

for(int i = q; i; i--)
    e[0][a[i].id].w = a[i].u;

CDQ(1, q, 0, m, 0); // 这是调用方法


// 分治主过程 O(nlog^2n)
// 需要调用Reduction和Contraction
void CDQ(int l, int r, int d, int m, long long ans) { // CDQ分治
    if (l == r) { // 区间长度已减小到1,输出答案,退出
        e[d][id[d][a[l].id]].w = a[l].v;
        printf("%lld\n", ans + Kruskal(m, e[d]));
        e[d][id[d][a[l].id]].w=a[l].u;
        return;
    }

    int tmp = top;

    Reduction(l, r, d, m);
    ans += Contraction(l, r, d, m); // R-C

    int mid = (l + r) / 2;

    copy(e[d] + 1, e[d] + m + 1, e[d + 1] + 1);
    for (int i = 1; i <= m; i++)
        id[d + 1][e[d][i].id] = i; // 准备好下一层要用的数组

    CDQ(l, mid, d + 1, m, ans);

    for (int i = 1; i <= mid; i++)
        e[d][id[d][a[i].id]].w = a[i].v; // 进行左边的修改

    copy(e[d] + 1, e[d] + m + 1, e[d + 1] + 1);
    for (int i = 1; i <= m; i++)
        id[d + 1][e[d][i].id] = i; // 重新准备下一层要用的数组

    CDQ(mid + 1, r, d + 1, m, ans);

    for (int i = top; i > tmp; i--)
        cut(stk[i]);//撤销所有操作
    top = tmp;
}


// Reduction(减少无用边):待修改边标为+INF,跑MST后把非树边删掉,减少无用边
// 需要调用Kruskal
void Reduction(int l, int r, int d, int &m) {
    for (int i = l; i <= r; i++)
        e[d][id[d][a[i].id]].w = INF;//待修改的边标为INF

    Kruskal(m, e[d]);

    copy(e[d] + 1, e[d] + m + 1, t + 1);

    int cnt = 0;
    for (int i = 1; i <= m; i++)
        if (t[i].w == INF || t[i].vis){ // 非树边扔掉
            id[d][t[i].id] = ++cnt; // 给边重新编号
            e[d][cnt] = t[i];
        }

    for (int i = r; i >= l; i--)
        e[d][id[d][a[i].id]].w = a[i].u; // 把待修改的边改回去

```

```cpp
    m=cnt;
}


// Contraction(缩必须边):待修改边标为-INF,跑MST后缩除待修改边之外的所有树边
// 返回缩掉的边的总权值
// 需要调用Kruskal
long long Contraction(int l, int r, int d, int &m) {
    long long ans = 0;

    for (int i = l; i <= r; i++)
        e[d][id[d][a[i].id]].w = -INF; // 待修改边标为-INF

    Kruskal(m, e[d]);
    copy(e[d] + 1, e[d] + m + 1, t + 1);

    int cnt = 0;
    for (int i = 1; i <= m ; i++) {

        if (t[i].w != -INF && t[i].vis) { // 必须边
            ans += t[i].w;
            mergeset(t[i].u, t[i].v);
        }
        else { // 不确定边
            id[d][t[i].id]=++cnt;
            e[d][cnt]=t[i];
        }
    }

    for (int i = r ; i >= l; i--) {
        e[d][id[d][a[i].id]].w = a[i].u; // 把待修改的边改回去
        e[d][id[d][a[i].id]].vis = false;
    }

    m = cnt;

    return ans;
}


// Kruskal算法 O(mlogn)
// 方便起见,这里直接沿用进行过缩点的并查集,在过程结束后撤销即可
long long Kruskal(int m, edge *e) {
    int tmp = top;
    long long ans = 0;

    sort(e + 1, e + m + 1); // 比较函数在结构体中定义过了

    for (int i = 1; i <= m; i++) {
        if (findroot(e[i].u) != findroot(e[i].v)) {
            e[i].vis = true;
            ans += e[i].w;
            mergeset(e[i].u, e[i].v);
        }
        else
            e[i].vis = false;
    }

    for(int i = top; i > tmp; i--)
        cut(stk[i]); // 撤销所有操作
    top = tmp;

    return ans;
}


// 以下是并查集相关函数
int findroot(int x) { // 因为需要撤销,不写路径压缩
    while (p[x] != x)
        x = p[x];

    return x;
}

void mergeset(int x, int y) { // 按size合并,如果想跑得更快就写一个按秩合并
    x = findroot(x); // 但是按秩合并要再开一个栈记录合并之前的秩
    y = findroot(y);

    if (x == y)
        return;

    if (size[x] > size[y])
        swap(x, y);

    p[x] = y;
    size[y] += size[x];
    stk[++top] = x;
}

void cut(int x) { // 并查集撤销
    int y = x;

    do
        size[y = p[y]] -= size[x];
    while (p[y]! = y);

    p[x] = x;
}
```

## 3.2 费用流

### 3.2.1 SPFA费用流

```cpp
constexpr int maxn = 20005, maxm = 200005;

struct edge {
    int to, prev, cap, w;
} e[maxm * 2];

int last[maxn], cnte, d[maxn], p[maxn]; // 记得把last初始化成-1, 不然会死循环
bool inq[maxn];

void spfa(int s) {

    memset(d, -63, sizeof(d));
    memset(p, -1, sizeof(p));

    queue<int> q;

    q.push(s);
    d[s] = 0;

    while (!q.empty()) {
        int x = q.front();
        q.pop();
        inq[x] = false;

        for (int i = last[x]; ~i; i = e[i].prev)
            if (e[i].cap) {
                int y = e[i].to;

                if (d[x] + e[i].w > d[y]) {
                    p[y] = i;
                    d[y] = d[x] + e[i].w;
```

```
                if (!inq[y]) {
                    q.push(y);
                    inq[y] = true;
                }
            }
        }
    }
}

int mcmf(int s, int t) {
    int ans = 0;

    while (spfa(s), d[t] > 0) {
        int flow = 0x3f3f3f3f;
        for (int x = t; x != s; x = e[p[x] ^ 1].to)
            flow = min(flow, e[p[x]].cap);

        ans += flow * d[t];

        for (int x = t; x != s; x = e[p[x] ^ 1].to) {
            e[p[x]].cap -= flow;
            e[p[x] ^ 1].cap += flow;
        }
    }

    return ans;
}

void add(int x, int y, int c, int w) {
    e[cnte].to = y;
    e[cnte].cap = c;
    e[cnte].w = w;

    e[cnte].prev = last[x];
    last[x] = cnte++;
}

void addedge(int x, int y, int c, int w) {
    add(x, y, c, w);
    add(y, x, 0, -w);
}
```

### 3.2.2 Dijkstra费用流

有的地方也叫原始-对偶费用流.

原理和求多源最短路的Johnson算法是一样的, 都是给每个点维护一个势$h_u$, 使得对任何有向边$u \to v$都满足$w + h_u - h_v \geq 0$.

如果有负费用则从$s$开始跑一遍SPFA初始化, 否则可以直接初始化$h_u = 0$.

每次增广时得到的路径长度就是$d_{s,t} + h_t$, 增广之后让所有$h_u = h'_u + d'_{s,u}$, 直到$d_{s,t} = \infty$(最小费用最大流)或$d_{s,t} \geq 0$(最小费用流)为止.

注意最大费用流要转成取负之后的最小费用流, 因为Dijkstra求的是最短路.

```
struct edge {
    int to, cap, prev, w;
} e[maxe * 2];

int last[maxn], cnte;

long long d[maxn], h[maxn];
int p[maxn];

bool vis[maxn];
int s, t;

void Adde(int x, int y, int z, int w) {
    e[cnte].to = y;
    e[cnte].cap = z;
    e[cnte].w = w;
    e[cnte].prev = last[x];
    last[x] = cnte++;
}

void addedge(int x, int y, int z, int w) {
    Adde(x, y, z, w);
    Adde(y, x, 0, -w);
}

void dijkstra() {
    memset(d, 63, sizeof(d));
    memset(vis, 0, sizeof(vis));

    priority_queue<pair<long long, int> > heap;

    d[s] = 0;
    heap.push(make_pair(0ll, s));

    while (!heap.empty()) {
        int x = heap.top().second;
        heap.pop();

        if (vis[x])
            continue;

        vis[x] = true;
        for (int i = last[x]; ~i; i = e[i].prev)
            if (e[i].cap > 0 && d[e[i].to] > d[x] + e[i].w
↪ + h[x] - h[e[i].to]) {
                d[e[i].to] = d[x] + e[i].w + h[x] -
↪ h[e[i].to];
                p[e[i].to] = i;
                heap.push(make_pair(-d[e[i].to], e[i].to));
            }
    }
}

pair<long long, long long> mcmf() {
    /*
    spfa();
    for (int i = 1; i <= t; i++)
        h[i] = d[i];
    // 如果初始有负权就像这样跑一遍SPFA预处理
    */

    long long flow = 0, cost = 0;

    while (dijkstra(), d[t] < 0x3f3f3f3f) {
        for (int i = 1; i <= t; i++)
            h[i] += d[i];

        int a = 0x3f3f3f3f;

        for (int x = t; x != s; x = e[p[x] ^ 1].to)
            a = min(a, e[p[x]].cap);

        flow += a;
        cost += (long long)a * h[t];

        for (int x = t; x != s; x = e[p[x] ^ 1].to) {
            e[p[x]].cap -= a;
            e[p[x] ^ 1].cap += a;
        }

    }

    return make_pair(flow, cost);
}
```

```
// 记得初始化
memset(last, -1, sizeof(last));
```

# 4 字符串

## 4.1 后缀自动机

```cpp
// 在字符集比较小的时候可以直接开go数组，否则需要用map或者
    ↪ 哈希表替换
// 注意!!!结点数要开成串长的两倍

// 全局变量与数组定义
int last, len[maxn], fa[maxn], go[maxn][26], sam_cnt;
int c[maxn], q[maxn]; // 用来桶排序

// 在主函数开头加上这句初始化
last = sam_cnt = 1;

// 以下是按val进行桶排序的代码
for (int i = 1; i <= sam_cnt; i++)
    c[len[i] + 1]++;
for (int i = 1; i <= n; i++)
    c[i] += c[i - 1]; // 这里n是串长
for (int i = 1; i <= sam_cnt; i++)
    q[++c[len[i]]] = i;

//加入一个字符 均摊O(1)
void extend(int c) {
    int p = last, np = ++sam_cnt;
    len[np] = len[p] + 1;

    while (p && !go[p][c]) {
        go[p][c] = np;
        p = fa[p];
    }

    if (!p)
        fa[np] = 1;
    else {
        int q = go[p][c];

        if (len[q] == len[p] + 1)
            fa[np] = q;
        else {
            int nq = ++sam_cnt;
            len[nq] = len[p] + 1;
            memcpy(go[nq], go[q], sizeof(go[q]));

            fa[nq] = fa[q];
            fa[np] = fa[q] = nq;

            while (p && go[p][c] == q){
                go[p][c] = nq;
                p = fa[p];
            }
        }
    }

    last = np;
}
```

## Theoretical Computer Science Cheat Sheet

| Definitions | |
|---|---|
| $f(n) = O(g(n))$ | iff $\exists$ positive $c, n_0$ such that $0 \le f(n) \le cg(n)$ $\forall n \ge n_0$. |
| $f(n) = \Omega(g(n))$ | iff $\exists$ positive $c, n_0$ such that $f(n) \ge cg(n) \ge 0$ $\forall n \ge n_0$. |
| $f(n) = \Theta(g(n))$ | iff $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. |
| $f(n) = o(g(n))$ | iff $\lim_{n\to\infty} f(n)/g(n) = 0$. |
| $\lim\limits_{n\to\infty} a_n = a$ | iff $\forall \epsilon > 0$, $\exists n_0$ such that $\lvert a_n - a\rvert < \epsilon$, $\forall n \ge n_0$. |
| $\sup S$ | least $b \in \mathbb{R}$ such that $b \ge s$, $\forall s \in S$. |
| $\inf S$ | greatest $b \in \mathbb{R}$ such that $b \le s$, $\forall s \in S$. |
| $\liminf\limits_{n\to\infty} a_n$ | $\lim\limits_{n\to\infty} \inf\{a_i \mid i \ge n, i \in \mathbb{N}\}$. |
| $\limsup\limits_{n\to\infty} a_n$ | $\lim\limits_{n\to\infty} \sup\{a_i \mid i \ge n, i \in \mathbb{N}\}$. |
| $\binom{n}{k}$ | Combinations: Size $k$ subsets of a size $n$ set. |
| $\left[{n \atop k}\right]$ | Stirling numbers (1st kind): Arrangements of an $n$ element set into $k$ cycles. |
| $\left\{{n \atop k}\right\}$ | Stirling numbers (2nd kind): Partitions of an $n$ element set into $k$ non-empty sets. |
| $\left\langle{n \atop k}\right\rangle$ | 1st order Eulerian numbers: Permutations $\pi_1\pi_2\ldots\pi_n$ on $\{1, 2, \ldots, n\}$ with $k$ ascents. |
| $\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle$ | 2nd order Eulerian numbers. |
| $C_n$ | Catalan Numbers: Binary trees with $n+1$ vertices. |

### Series

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}, \quad \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}, \quad \sum_{i=1}^{n} i^3 = \frac{n^2(n+1)^2}{4}.$$

In general:

$$\sum_{i=1}^{n} i^m = \frac{1}{m+1}\left[(n+1)^{m+1} - 1 - \sum_{i=1}^{n}\left((i+1)^{m+1} - i^{m+1} - (m+1)i^m\right)\right]$$

$$\sum_{i=1}^{n-1} i^m = \frac{1}{m+1}\sum_{k=0}^{m}\binom{m+1}{k} B_k n^{m+1-k}.$$

Geometric series:

$$\sum_{i=0}^{n} c^i = \frac{c^{n+1}-1}{c-1}, \quad c \ne 1, \quad \sum_{i=0}^{\infty} c^i = \frac{1}{1-c}, \quad \sum_{i=1}^{\infty} c^i = \frac{c}{1-c}, \quad |c| < 1,$$

$$\sum_{i=0}^{n} ic^i = \frac{nc^{n+2} - (n+1)c^{n+1} + c}{(c-1)^2}, \quad c \ne 1, \quad \sum_{i=0}^{\infty} ic^i = \frac{c}{(1-c)^2}, \quad |c| < 1.$$

Harmonic series:

$$H_n = \sum_{i=1}^{n} \frac{1}{i}, \quad \sum_{i=1}^{n} iH_i = \frac{n(n+1)}{2}H_n - \frac{n(n-1)}{4}.$$

$$\sum_{i=1}^{n} H_i = (n+1)H_n - n, \quad \sum_{i=1}^{n}\binom{i}{m}H_i = \binom{n+1}{m+1}\left(H_{n+1} - \frac{1}{m+1}\right).$$

**1.** $\displaystyle\binom{n}{k} = \frac{n!}{(n-k)!k!}$, **2.** $\displaystyle\sum_{k=0}^{n}\binom{n}{k} = 2^n$, **3.** $\displaystyle\binom{n}{k} = \binom{n}{n-k}$,

**4.** $\displaystyle\binom{n}{k} = \frac{n}{k}\binom{n-1}{k-1}$, **5.** $\displaystyle\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$,

**6.** $\displaystyle\binom{n}{m}\binom{m}{k} = \binom{n}{k}\binom{n-k}{m-k}$, **7.** $\displaystyle\sum_{k=0}^{n}\binom{r+k}{k} = \binom{r+n+1}{n}$,

**8.** $\displaystyle\sum_{k=0}^{n}\binom{k}{m} = \binom{n+1}{m+1}$, **9.** $\displaystyle\sum_{k=0}^{n}\binom{r}{k}\binom{s}{n-k} = \binom{r+s}{n}$,

**10.** $\displaystyle\binom{n}{k} = (-1)^k\binom{k-n-1}{k}$, **11.** $\displaystyle\left\{{n \atop 1}\right\} = \left\{{n \atop n}\right\} = 1$,

**12.** $\displaystyle\left\{{n \atop 2}\right\} = 2^{n-1} - 1$, **13.** $\displaystyle\left\{{n \atop k}\right\} = k\left\{{n-1 \atop k}\right\} + \left\{{n-1 \atop k-1}\right\}$,

**14.** $\displaystyle\left[{n \atop 1}\right] = (n-1)!$, **15.** $\displaystyle\left[{n \atop 2}\right] = (n-1)!H_{n-1}$, **16.** $\displaystyle\left[{n \atop n}\right] = 1$, **17.** $\displaystyle\left[{n \atop k}\right] \ge \left\{{n \atop k}\right\}$,

**18.** $\displaystyle\left[{n \atop k}\right] = (n-1)\left[{n-1 \atop k}\right] + \left[{n-1 \atop k-1}\right]$, **19.** $\displaystyle\left\{{n \atop n-1}\right\} = \left[{n \atop n-1}\right] = \binom{n}{2}$, **20.** $\displaystyle\sum_{k=0}^{n}\left[{n \atop k}\right] = n!$, **21.** $\displaystyle C_n = \frac{1}{n+1}\binom{2n}{n}$,

**22.** $\displaystyle\left\langle{n \atop 0}\right\rangle = \left\langle{n \atop n-1}\right\rangle = 1$, **23.** $\displaystyle\left\langle{n \atop k}\right\rangle = \left\langle{n \atop n-1-k}\right\rangle$, **24.** $\displaystyle\left\langle{n \atop k}\right\rangle = (k+1)\left\langle{n-1 \atop k}\right\rangle + (n-k)\left\langle{n-1 \atop k-1}\right\rangle$,

**25.** $\displaystyle\left\langle{0 \atop k}\right\rangle = \begin{cases} 1 & \text{if } k = 0, \\ 0 & \text{otherwise} \end{cases}$ **26.** $\displaystyle\left\langle{n \atop 1}\right\rangle = 2^n - n - 1$, **27.** $\displaystyle\left\langle{n \atop 2}\right\rangle = 3^n - (n+1)2^n + \binom{n+1}{2}$,

**28.** $\displaystyle x^n = \sum_{k=0}^{n}\left\langle{n \atop k}\right\rangle\binom{x+k}{n}$, **29.** $\displaystyle\left\langle{n \atop m}\right\rangle = \sum_{k=0}^{m}\binom{n+1}{k}(m+1-k)^n(-1)^k$, **30.** $\displaystyle m!\left\{{n \atop m}\right\} = \sum_{k=0}^{n}\left\langle{n \atop k}\right\rangle\binom{k}{n-m}$,

**31.** $\displaystyle\left\langle{n \atop m}\right\rangle = \sum_{k=0}^{n}\left\{{n \atop k}\right\}\binom{n-k}{m}(-1)^{n-k-m}k!$, **32.** $\displaystyle\left\langle\!\left\langle{n \atop 0}\right\rangle\!\right\rangle = 1$, **33.** $\displaystyle\left\langle\!\left\langle{n \atop n}\right\rangle\!\right\rangle = 0 \quad \text{for } n \ne 0$,

**34.** $\displaystyle\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle = (k+1)\left\langle\!\left\langle{n-1 \atop k}\right\rangle\!\right\rangle + (2n-1-k)\left\langle\!\left\langle{n-1 \atop k-1}\right\rangle\!\right\rangle$, **35.** $\displaystyle\sum_{k=0}^{n}\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle = \frac{(2n)^{\underline{n}}}{2^n}$,

**36.** $\displaystyle\left\{{x \atop x-n}\right\} = \sum_{k=0}^{n}\left\langle\!\left\langle{n \atop k}\right\rangle\!\right\rangle\binom{x+n-1-k}{2n}$, **37.** $\displaystyle\left\{{n+1 \atop m+1}\right\} = \sum_{k}\binom{n}{k}\left\{{k \atop m}\right\} = \sum_{k=0}^{n}\left\{{k \atop m}\right\}(m+1)^{n-k}$,

# Theoretical Computer Science Cheat Sheet

## Identities Cont.

**38.** $\left[{n+1 \atop m+1}\right] = \sum_k \left[{n \atop k}\right]\binom{k}{m} = \sum_{k=0}^{n}\left[{k \atop m}\right] n^{\underline{n-k}} = n!\sum_{k=0}^{n}\frac{1}{k!}\left[{k \atop m}\right]$,

**39.** $\left[{x \atop x-n}\right] = \sum_{k=0}^{n}\left\langle\!\!\left\langle {n \atop k} \right\rangle\!\!\right\rangle \binom{x+k}{2n}$,

**40.** $\left\{{n \atop m}\right\} = \sum_k \binom{n}{k}\left\{{k+1 \atop m+1}\right\}(-1)^{n-k}$,

**41.** $\left[{n \atop m}\right] = \sum_k \left[{n+1 \atop k+1}\right]\binom{k}{m}(-1)^{m-k}$,

**42.** $\left\{{m+n+1 \atop m}\right\} = \sum_{k=0}^{m} k\left\{{n+k \atop k}\right\}$,

**43.** $\left[{m+n+1 \atop m}\right] = \sum_{k=0}^{m} k(n+k)\left[{n+k \atop k}\right]$,

**44.** $\binom{n}{m} = \sum_k \left\{{n+1 \atop k+1}\right\}\left[{k \atop m}\right](-1)^{m-k}$,

**45.** $(n-m)!\binom{n}{m} = \sum_k \left[{n+1 \atop k+1}\right]\left\{{k \atop m}\right\}(-1)^{m-k}$, for $n \geq m$,

**46.** $\left\{{n \atop n-m}\right\} = \sum_k \binom{m-n}{m+k}\binom{m+n}{n+k}\left[{m+k \atop k}\right]$,

**47.** $\left[{n \atop n-m}\right] = \sum_k \binom{m-n}{m+k}\binom{m+n}{n+k}\left\{{m+k \atop k}\right\}$,

**48.** $\left\{{n \atop \ell+m}\right\}\binom{\ell+m}{\ell} = \sum_k \left\{{k \atop \ell}\right\}\left\{{n-k \atop m}\right\}\binom{n}{k}$,

**49.** $\left[{n \atop \ell+m}\right]\binom{\ell+m}{\ell} = \sum_k \left[{k \atop \ell}\right]\left[{n-k \atop m}\right]\binom{n}{k}$.

## Trees

Every tree with $n$ vertices has $n-1$ edges.

Kraft inequality: If the depths of the leaves of a binary tree are $d_1, \ldots, d_n$:
$$\sum_{i=1}^{n} 2^{-d_i} \leq 1,$$
and equality holds only if every internal node has 2 sons.

## Recurrences

Master method:
$$T(n) = aT(n/b) + f(n), \quad a \geq 1, b > 1$$

If $\exists \epsilon > 0$ such that $f(n) = O(n^{\log_b a - \epsilon})$ then
$$T(n) = \Theta(n^{\log_b a}).$$

If $f(n) = \Theta(n^{\log_b a})$ then
$$T(n) = \Theta(n^{\log_b a}\log_2 n).$$

If $\exists \epsilon > 0$ such that $f(n) = \Omega(n^{\log_b a + \epsilon})$, and $\exists c < 1$ such that $af(n/b) \leq cf(n)$ for large $n$, then
$$T(n) = \Theta(f(n)).$$

Substitution (example): Consider the following recurrence
$$T_{i+1} = 2^{2^i} \cdot T_i^2, \quad T_1 = 2.$$

Note that $T_i$ is always a power of two. Let $t_i = \log_2 T_i$. Then we have
$$t_{i+1} = 2^i + 2t_i, \quad t_1 = 1.$$

Let $u_i = t_i/2^i$. Dividing both sides of the previous equation by $2^{i+1}$ we get
$$\frac{t_{i+1}}{2^{i+1}} = \frac{2^i}{2^{i+1}} + \frac{t_i}{2^i}.$$

Substituting we find
$$u_{i+1} = \tfrac{1}{2} + u_i, \qquad u_1 = \tfrac{1}{2},$$

which is simply $u_i = i/2$. So we find that $T_i$ has the closed form $T_i = 2^{i2^{i-1}}$.

Summing factors (example): Consider the following recurrence
$$T(n) = 3T(n/2) + n, \quad T(1) = 1.$$

Rewrite so that all terms involving $T$ are on the left side
$$T(n) - 3T(n/2) = n.$$

Now expand the recurrence, and choose a factor which makes the left side "telescope"

$$1\big(T(n) - 3T(n/2) = n\big)$$
$$3\big(T(n/2) - 3T(n/4) = n/2\big)$$
$$\vdots \qquad \vdots \qquad \vdots$$
$$3^{\log_2 n - 1}\big(T(2) - 3T(1) = 2\big)$$

Let $m = \log_2 n$. Summing the left side we get $T(n) - 3^m T(1) = T(n) - 3^m = T(n) - n^k$ where $k = \log_2 3 \approx 1.58496$. Summing the right side we get
$$\sum_{i=0}^{m-1}\frac{n}{2^i}3^i = n\sum_{i=0}^{m-1}\left(\tfrac{3}{2}\right)^i.$$

Let $c = \frac{3}{2}$. Then we have
$$\begin{aligned} n\sum_{i=0}^{m-1} c^i &= n\left(\frac{c^m - 1}{c-1}\right) \\ &= 2n(c^{\log_2 n} - 1) \\ &= 2n(c^{(k-1)\log_c n} - 1) \\ &= 2n^k - 2n, \end{aligned}$$

and so $T(n) = 3n^k - 2n$. Full history recurrences can often be changed to limited history ones (example): Consider
$$T_i = 1 + \sum_{j=0}^{i-1} T_j, \quad T_0 = 1.$$

Note that
$$T_{i+1} = 1 + \sum_{j=0}^{i} T_j.$$

Subtracting we find
$$\begin{aligned} T_{i+1} - T_i &= 1 + \sum_{j=0}^{i} T_j - 1 - \sum_{j=0}^{i-1} T_j \\ &= T_i. \end{aligned}$$

And so $T_{i+1} = 2T_i = 2^{i+1}$.

Generating functions:

1. Multiply both sides of the equation by $x^i$.
2. Sum both sides over all $i$ for which the equation is valid.
3. Choose a generating function $G(x)$. Usually $G(x) = \sum_{i=0}^{\infty} x^i g_i$.
3. Rewrite the equation in terms of the generating function $G(x)$.
4. Solve for $G(x)$.
5. The coefficient of $x^i$ in $G(x)$ is $g_i$.

Example:
$$g_{i+1} = 2g_i + 1, \quad g_0 = 0.$$

Multiply and sum:
$$\sum_{i\geq 0} g_{i+1}x^i = \sum_{i\geq 0} 2g_i x^i + \sum_{i\geq 0} x^i.$$

We choose $G(x) = \sum_{i\geq 0} x^i g_i$. Rewrite in terms of $G(x)$:
$$\frac{G(x) - g_0}{x} = 2G(x) + \sum_{i\geq 0} x^i.$$

Simplify:
$$\frac{G(x)}{x} = 2G(x) + \frac{1}{1-x}.$$

Solve for $G(x)$:
$$G(x) = \frac{x}{(1-x)(1-2x)}.$$

Expand this using partial fractions:
$$\begin{aligned} G(x) &= x\left(\frac{2}{1-2x} - \frac{1}{1-x}\right) \\ &= x\left(2\sum_{i\geq 0} 2^i x^i - \sum_{i\geq 0} x^i\right) \\ &= \sum_{i\geq 0}(2^{i+1} - 1)x^{i+1}. \end{aligned}$$

So $g_i = 2^i - 1$.

# Theoretical Computer Science Cheat Sheet

$\pi \approx 3.14159$, $e \approx 2.71828$, $\gamma \approx 0.57721$, $\phi = \frac{1+\sqrt{5}}{2} \approx 1.61803$, $\hat{\phi} = \frac{1-\sqrt{5}}{2} \approx -.61803$

| $i$ | $2^i$ | $p_i$ |
|---|---|---|
| 1 | 2 | 2 |
| 2 | 4 | 3 |
| 3 | 8 | 5 |
| 4 | 16 | 7 |
| 5 | 32 | 11 |
| 6 | 64 | 13 |
| 7 | 128 | 17 |
| 8 | 256 | 19 |
| 9 | 512 | 23 |
| 10 | 1,024 | 29 |
| 11 | 2,048 | 31 |
| 12 | 4,096 | 37 |
| 13 | 8,192 | 41 |
| 14 | 16,384 | 43 |
| 15 | 32,768 | 47 |
| 16 | 65,536 | 53 |
| 17 | 131,072 | 59 |
| 18 | 262,144 | 61 |
| 19 | 524,288 | 67 |
| 20 | 1,048,576 | 71 |
| 21 | 2,097,152 | 73 |
| 22 | 4,194,304 | 79 |
| 23 | 8,388,608 | 83 |
| 24 | 16,777,216 | 89 |
| 25 | 33,554,432 | 97 |
| 26 | 67,108,864 | 101 |
| 27 | 134,217,728 | 103 |
| 28 | 268,435,456 | 107 |
| 29 | 536,870,912 | 109 |
| 30 | 1,073,741,824 | 113 |
| 31 | 2,147,483,648 | 127 |
| 32 | 4,294,967,296 | 131 |

## Pascal's Triangle

1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1
1 9 36 84 126 126 84 36 9 1
1 10 45 120 210 252 210 120 45 10 1

## General

Bernoulli Numbers ($B_i = 0$, odd $i \neq 1$):

$$B_0 = 1,\ B_1 = -\tfrac{1}{2},\ B_2 = \tfrac{1}{6},\ B_4 = -\tfrac{1}{30},$$
$$B_6 = \tfrac{1}{42},\ B_8 = -\tfrac{1}{30},\ B_{10} = \tfrac{5}{66}.$$

Change of base, quadratic formula:

$$\log_b x = \frac{\log_a x}{\log_a b}, \qquad \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

Euler's number $e$:

$$e = 1 + \tfrac{1}{2} + \tfrac{1}{6} + \tfrac{1}{24} + \tfrac{1}{120} + \cdots$$
$$\lim_{n \to \infty} \left(1 + \frac{x}{n}\right)^n = e^x.$$
$$\left(1 + \tfrac{1}{n}\right)^n < e < \left(1 + \tfrac{1}{n}\right)^{n+1}.$$
$$\left(1 + \tfrac{1}{n}\right)^n = e - \frac{e}{2n} + \frac{11e}{24n^2} - O\left(\frac{1}{n^3}\right).$$

Harmonic numbers:

$$1,\ \tfrac{3}{2},\ \tfrac{11}{6},\ \tfrac{25}{12},\ \tfrac{137}{60},\ \tfrac{49}{20},\ \tfrac{363}{140},\ \tfrac{761}{280},\ \tfrac{7129}{2520},\ \ldots$$

$$\ln n < H_n < \ln n + 1,$$
$$H_n = \ln n + \gamma + O\left(\frac{1}{n}\right).$$

Factorial, Stirling's approximation:

$$1,\ 2,\ 6,\ 24,\ 120,\ 720,\ 5040,\ 40320,\ 362880,\ \ldots$$

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right).$$

Ackermann's function and inverse:

$$a(i,j) = \begin{cases} 2^j & i = 1 \\ a(i-1, 2) & j = 1 \\ a(i-1, a(i, j-1)) & i, j \geq 2 \end{cases}$$
$$\alpha(i) = \min\{j \mid a(j,j) \geq i\}.$$

Binomial distribution:

$$\Pr[X = k] = \binom{n}{k} p^k q^{n-k}, \qquad q = 1 - p,$$
$$\mathrm{E}[X] = \sum_{k=1}^{n} k \binom{n}{k} p^k q^{n-k} = np.$$

Poisson distribution:

$$\Pr[X = k] = \frac{e^{-\lambda}\lambda^k}{k!}, \quad \mathrm{E}[X] = \lambda.$$

Normal (Gaussian) distribution:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}, \quad \mathrm{E}[X] = \mu.$$

The "coupon collector": We are given a random coupon each day, and there are $n$ different types of coupons. The distribution of coupons is uniform. The expected number of days to pass before we to collect all $n$ types is

$$nH_n.$$

## Probability

Continuous distributions: If

$$\Pr[a < X < b] = \int_a^b p(x)\,dx,$$

then $p$ is the probability density function of $X$. If

$$\Pr[X < a] = P(a),$$

then $P$ is the distribution function of $X$. If $P$ and $p$ both exist then

$$P(a) = \int_{-\infty}^{a} p(x)\,dx.$$

Expectation: If $X$ is discrete

$$\mathrm{E}[g(X)] = \sum_x g(x) \Pr[X = x].$$

If $X$ continuous then

$$\mathrm{E}[g(X)] = \int_{-\infty}^{\infty} g(x)p(x)\,dx = \int_{-\infty}^{\infty} g(x)\,dP(x).$$

Variance, standard deviation:

$$\mathrm{VAR}[X] = \mathrm{E}[X^2] - \mathrm{E}[X]^2,$$
$$\sigma = \sqrt{\mathrm{VAR}[X]}.$$

For events $A$ and $B$:

$$\Pr[A \vee B] = \Pr[A] + \Pr[B] - \Pr[A \wedge B]$$
$$\Pr[A \wedge B] = \Pr[A] \cdot \Pr[B],$$

iff $A$ and $B$ are independent.

$$\Pr[A|B] = \frac{\Pr[A \wedge B]}{\Pr[B]}$$

For random variables $X$ and $Y$:

$$\mathrm{E}[X \cdot Y] = \mathrm{E}[X] \cdot \mathrm{E}[Y],$$

if $X$ and $Y$ are independent.

$$\mathrm{E}[X + Y] = \mathrm{E}[X] + \mathrm{E}[Y],$$
$$\mathrm{E}[cX] = c\,\mathrm{E}[X].$$

Bayes' theorem:

$$\Pr[A_i|B] = \frac{\Pr[B|A_i]\Pr[A_i]}{\sum_{j=1}^{n} \Pr[A_j]\Pr[B|A_j]}.$$

Inclusion-exclusion:

$$\Pr\left[\bigvee_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \Pr[X_i] + \sum_{k=2}^{n} (-1)^{k+1} \sum_{i_i < \cdots < i_k} \Pr\left[\bigwedge_{j=1}^{k} X_{i_j}\right].$$

Moment inequalities:

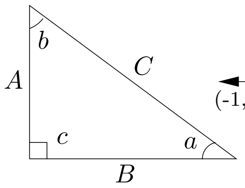
$$\Pr\left[|X| \geq \lambda\,\mathrm{E}[X]\right] \leq \frac{1}{\lambda},$$
$$\Pr\left[\left|X - \mathrm{E}[X]\right| \geq \lambda \cdot \sigma\right] \leq \frac{1}{\lambda^2}.$$

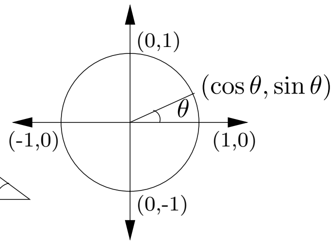
Geometric distribution:

$$\Pr[X = k] = pq^{k-1}, \qquad q = 1 - p,$$
$$\mathrm{E}[X] = \sum_{k=1}^{\infty} kpq^{k-1} = \frac{1}{p}.$$

# Theoretical Computer Science Cheat Sheet

## Trigonometry




Pythagorean theorem:

$$C^2 = A^2 + B^2.$$

Definitions:

$$\sin a = A/C, \quad \cos a = B/C,$$
$$\csc a = C/A, \quad \sec a = C/B,$$
$$\tan a = \frac{\sin a}{\cos a} = \frac{A}{B}, \quad \cot a = \frac{\cos a}{\sin a} = \frac{B}{A}.$$

Area, radius of inscribed circle:

$$\tfrac{1}{2}AB, \quad \frac{AB}{A+B+C}.$$

Identities:

$$\sin x = \frac{1}{\csc x}, \qquad \cos x = \frac{1}{\sec x},$$
$$\tan x = \frac{1}{\cot x}, \qquad \sin^2 x + \cos^2 x = 1,$$
$$1 + \tan^2 x = \sec^2 x, \qquad 1 + \cot^2 x = \csc^2 x,$$
$$\sin x = \cos\left(\tfrac{\pi}{2} - x\right), \qquad \sin x = \sin(\pi - x),$$
$$\cos x = -\cos(\pi - x), \qquad \tan x = \cot\left(\tfrac{\pi}{2} - x\right),$$
$$\cot x = -\cot(\pi - x), \qquad \csc x = \cot\tfrac{x}{2} - \cot x,$$
$$\sin(x \pm y) = \sin x \cos y \pm \cos x \sin y,$$
$$\cos(x \pm y) = \cos x \cos y \mp \sin x \sin y,$$
$$\tan(x \pm y) = \frac{\tan x \pm \tan y}{1 \mp \tan x \tan y},$$
$$\cot(x \pm y) = \frac{\cot x \cot y \mp 1}{\cot x \pm \cot y},$$
$$\sin 2x = 2 \sin x \cos x, \qquad \sin 2x = \frac{2 \tan x}{1 + \tan^2 x},$$
$$\cos 2x = \cos^2 x - \sin^2 x, \qquad \cos 2x = 2\cos^2 x - 1,$$
$$\cos 2x = 1 - 2\sin^2 x, \qquad \cos 2x = \frac{1 - \tan^2 x}{1 + \tan^2 x},$$
$$\tan 2x = \frac{2 \tan x}{1 - \tan^2 x}, \qquad \cot 2x = \frac{\cot^2 x - 1}{2 \cot x},$$
$$\sin(x+y)\sin(x-y) = \sin^2 x - \sin^2 y,$$
$$\cos(x+y)\cos(x-y) = \cos^2 x - \sin^2 y.$$

Euler's equation:

$$e^{ix} = \cos x + i \sin x, \qquad e^{i\pi} = -1.$$


v2.02 ©1994 by Steve Seiden

sseiden@acm.org

http://www.csc.lsu.edu/~seiden


## Matrices

Multiplication:

$$C = A \cdot B, \quad c_{i,j} = \sum_{k=1}^{n} a_{i,k} b_{k,j}.$$

Determinants: $\det A \neq 0$ iff $A$ is non-singular.

$$\det A \cdot B = \det A \cdot \det B,$$
$$\det A = \sum_{\pi} \prod_{i=1}^{n} \operatorname{sign}(\pi) a_{i,\pi(i)}.$$

$2 \times 2$ and $3 \times 3$ determinant:

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc,$$
$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = g \begin{vmatrix} b & c \\ e & f \end{vmatrix} - h \begin{vmatrix} a & c \\ d & f \end{vmatrix} + i \begin{vmatrix} a & b \\ d & e \end{vmatrix}$$
$$= \begin{matrix} aei + bfg + cdh \\ - ceg - fha - ibd. \end{matrix}$$

Permanents:

$$\operatorname{perm} A = \sum_{\pi} \prod_{i=1}^{n} a_{i,\pi(i)}.$$

## Hyperbolic Functions

Definitions:

$$\sinh x = \frac{e^x - e^{-x}}{2}, \qquad \cosh x = \frac{e^x + e^{-x}}{2},$$
$$\tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \qquad \operatorname{csch} x = \frac{1}{\sinh x},$$
$$\operatorname{sech} x = \frac{1}{\cosh x}, \qquad \coth x = \frac{1}{\tanh x}.$$

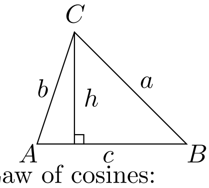
Identities:

$$\cosh^2 x - \sinh^2 x = 1, \qquad \tanh^2 x + \operatorname{sech}^2 x = 1,$$
$$\coth^2 x - \operatorname{csch}^2 x = 1, \qquad \sinh(-x) = -\sinh x,$$
$$\cosh(-x) = \cosh x, \qquad \tanh(-x) = -\tanh x,$$
$$\sinh(x+y) = \sinh x \cosh y + \cosh x \sinh y,$$
$$\cosh(x+y) = \cosh x \cosh y + \sinh x \sinh y,$$
$$\sinh 2x = 2 \sinh x \cosh x,$$
$$\cosh 2x = \cosh^2 x + \sinh^2 x,$$
$$\cosh x + \sinh x = e^x, \qquad \cosh x - \sinh x = e^{-x},$$
$$(\cosh x + \sinh x)^n = \cosh nx + \sinh nx, \quad n \in \mathbb{Z},$$
$$2\sinh^2 \tfrac{x}{2} = \cosh x - 1, \qquad 2\cosh^2 \tfrac{x}{2} = \cosh x + 1.$$

| $\theta$ | $\sin\theta$ | $\cos\theta$ | $\tan\theta$ |
|---|---|---|---|
| $0$ | $0$ | $1$ | $0$ |
| $\frac{\pi}{6}$ | $\frac{1}{2}$ | $\frac{\sqrt{3}}{2}$ | $\frac{\sqrt{3}}{3}$ |
| $\frac{\pi}{4}$ | $\frac{\sqrt{2}}{2}$ | $\frac{\sqrt{2}}{2}$ | $1$ |
| $\frac{\pi}{3}$ | $\frac{\sqrt{3}}{2}$ | $\frac{1}{2}$ | $\sqrt{3}$ |
| $\frac{\pi}{2}$ | $1$ | $0$ | $\infty$ |

... in mathematics you don't understand things, you just get used to them.
– J. von Neumann

## More Trig.




Law of cosines:

$$c^2 = a^2 + b^2 - 2ab\cos C.$$

Area:

$$A = \tfrac{1}{2}hc,$$
$$= \tfrac{1}{2}ab\sin C,$$
$$= \frac{c^2 \sin A \sin B}{2 \sin C}.$$

Heron's formula:

$$A = \sqrt{s \cdot s_a \cdot s_b \cdot s_c},$$
$$s = \tfrac{1}{2}(a + b + c),$$
$$s_a = s - a,$$
$$s_b = s - b,$$
$$s_c = s - c.$$

More identities:

$$\sin\tfrac{x}{2} = \sqrt{\frac{1 - \cos x}{2}},$$
$$\cos\tfrac{x}{2} = \sqrt{\frac{1 + \cos x}{2}},$$
$$\tan\tfrac{x}{2} = \sqrt{\frac{1 - \cos x}{1 + \cos x}},$$
$$= \frac{1 - \cos x}{\sin x},$$
$$= \frac{\sin x}{1 + \cos x},$$
$$\cot\tfrac{x}{2} = \sqrt{\frac{1 + \cos x}{1 - \cos x}},$$
$$= \frac{1 + \cos x}{\sin x},$$
$$= \frac{\sin x}{1 - \cos x},$$
$$\sin x = \frac{e^{ix} - e^{-ix}}{2i},$$
$$\cos x = \frac{e^{ix} + e^{-ix}}{2},$$
$$\tan x = -i\frac{e^{ix} - e^{-ix}}{e^{ix} + e^{-ix}},$$
$$= -i\frac{e^{2ix} - 1}{e^{2ix} + 1},$$
$$\sin x = \frac{\sinh ix}{i},$$
$$\cos x = \cosh ix,$$
$$\tan x = \frac{\tanh ix}{i}.$$

## Theoretical Computer Science Cheat Sheet

### Number Theory

The Chinese remainder theorem: There exists a number $C$ such that:

$$\begin{aligned} C &\equiv r_1 \bmod m_1 \\ \vdots\ &\ \vdots\ \ \vdots \\ C &\equiv r_n \bmod m_n \end{aligned}$$

if $m_i$ and $m_j$ are relatively prime for $i \neq j$.

Euler's function: $\phi(x)$ is the number of positive integers less than $x$ relatively prime to $x$. If $\prod_{i=1}^{n} p_i^{e_i}$ is the prime factorization of $x$ then

$$\phi(x) = \prod_{i=1}^{n} p_i^{e_i-1}(p_i - 1).$$

Euler's theorem: If $a$ and $b$ are relatively prime then

$$1 \equiv a^{\phi(b)} \bmod b.$$

Fermat's theorem:

$$1 \equiv a^{p-1} \bmod p.$$

The Euclidean algorithm: if $a > b$ are integers then

$$\gcd(a, b) = \gcd(a \bmod b, b).$$

If $\prod_{i=1}^{n} p_i^{e_i}$ is the prime factorization of $x$ then

$$S(x) = \sum_{d|x} d = \prod_{i=1}^{n} \frac{p_i^{e_i+1} - 1}{p_i - 1}.$$

Perfect Numbers: $x$ is an even perfect number iff $x = 2^{n-1}(2^n - 1)$ and $2^n - 1$ is prime.
Wilson's theorem: $n$ is a prime iff

$$(n-1)! \equiv -1 \bmod n.$$

Möbius inversion:

$$\mu(i) = \begin{cases} 1 & \text{if } i = 1. \\ 0 & \text{if } i \text{ is not square-free.} \\ (-1)^r & \text{if } i \text{ is the product of} \\ & r \text{ distinct primes.} \end{cases}$$

If

$$G(a) = \sum_{d|a} F(d),$$

then

$$F(a) = \sum_{d|a} \mu(d) G\Big(\frac{a}{d}\Big).$$

Prime numbers:

$$p_n = n \ln n + n \ln\ln n - n + n \frac{\ln\ln n}{\ln n} + O\Big(\frac{n}{\ln n}\Big),$$

$$\pi(n) = \frac{n}{\ln n} + \frac{n}{(\ln n)^2} + \frac{2!n}{(\ln n)^3} + O\Big(\frac{n}{(\ln n)^4}\Big).$$

### Graph Theory

Definitions:

| | |
|---|---|
| *Loop* | An edge connecting a vertex to itself. |
| *Directed* | Each edge has a direction. |
| *Simple* | Graph with no loops or multi-edges. |
| *Walk* | A sequence $v_0 e_1 v_1 \ldots e_\ell v_\ell$. |
| *Trail* | A walk with distinct edges. |
| *Path* | A trail with distinct vertices. |
| *Connected* | A graph where there exists a path between any two vertices. |
| *Component* | A maximal connected subgraph. |
| *Tree* | A connected acyclic graph. |
| *Free tree* | A tree with no root. |
| *DAG* | Directed acyclic graph. |
| *Eulerian* | Graph with a trail visiting each edge exactly once. |
| *Hamiltonian* | Graph with a cycle visiting each vertex exactly once. |
| *Cut* | A set of edges whose removal increases the number of components. |
| *Cut-set* | A minimal cut. |
| *Cut edge* | A size 1 cut. |
| *k-Connected* | A graph connected with the removal of any $k-1$ vertices. |
| *k-Tough* | $\forall S \subseteq V, S \neq \emptyset$ we have $k \cdot c(G - S) \leq \lvert S \rvert$. |
| *k-Regular* | A graph where all vertices have degree $k$. |
| *k-Factor* | A $k$-regular spanning subgraph. |
| *Matching* | A set of edges, no two of which are adjacent. |
| *Clique* | A set of vertices, all of which are adjacent. |
| *Ind. set* | A set of vertices, none of which are adjacent. |
| *Vertex cover* | A set of vertices which cover all edges. |
| *Planar graph* | A graph which can be embeded in the plane. |
| *Plane graph* | An embedding of a planar graph. |

$$\sum_{v \in V} \deg(v) = 2m.$$

If $G$ is planar then $n - m + f = 2$, so

$$f \leq 2n - 4, \quad m \leq 3n - 6.$$

Any planar graph has a vertex with degree $\leq 5$.

Notation:

| | |
|---|---|
| $E(G)$ | Edge set |
| $V(G)$ | Vertex set |
| $c(G)$ | Number of components |
| $G[S]$ | Induced subgraph |
| $\deg(v)$ | Degree of $v$ |
| $\Delta(G)$ | Maximum degree |
| $\delta(G)$ | Minimum degree |
| $\chi(G)$ | Chromatic number |
| $\chi_E(G)$ | Edge chromatic number |
| $G^c$ | Complement graph |
| $K_n$ | Complete graph |
| $K_{n_1,n_2}$ | Complete bipartite graph |
| $\mathrm{r}(k, \ell)$ | Ramsey number |

### Geometry

Projective coordinates: triples $(x, y, z)$, not all $x$, $y$ and $z$ zero.

$$(x, y, z) = (cx, cy, cz) \quad \forall c \neq 0.$$

| Cartesian | Projective |
|---|---|
| $(x, y)$ | $(x, y, 1)$ |
| $y = mx + b$ | $(m, -1, b)$ |
| $x = c$ | $(1, 0, -c)$ |

Distance formula, $L_p$ and $L_\infty$ metric:

$$\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2},$$

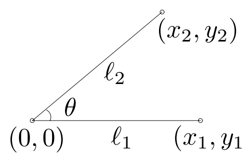
$$\big[|x_1 - x_0|^p + |y_1 - y_0|^p\big]^{1/p},$$

$$\lim_{p \to \infty} \big[|x_1 - x_0|^p + |y_1 - y_0|^p\big]^{1/p}.$$

Area of triangle $(x_0, y_0)$, $(x_1, y_1)$ and $(x_2, y_2)$:

$$\tfrac{1}{2} \operatorname{abs} \begin{vmatrix} x_1 - x_0 & y_1 - y_0 \\ x_2 - x_0 & y_2 - y_0 \end{vmatrix}.$$

Angle formed by three points:




$$\cos\theta = \frac{(x_1, y_1) \cdot (x_2, y_2)}{\ell_1 \ell_2}.$$

Line through two points $(x_0, y_0)$ and $(x_1, y_1)$:

$$\begin{vmatrix} x & y & 1 \\ x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \end{vmatrix} = 0.$$

Area of circle, volume of sphere:

$$A = \pi r^2, \qquad V = \tfrac{4}{3}\pi r^3.$$

If I have seen farther than others, it is because I have stood on the shoulders of giants.
– Issac Newton

## Theoretical Computer Science Cheat Sheet

### $\pi$

Wallis' identity:
$$\pi = 2 \cdot \frac{2 \cdot 2 \cdot 4 \cdot 4 \cdot 6 \cdot 6 \cdots}{1 \cdot 3 \cdot 3 \cdot 5 \cdot 5 \cdot 7 \cdots}$$

Brouncker's continued fraction expansion:
$$\tfrac{\pi}{4} = 1 + \cfrac{1^2}{2 + \cfrac{3^2}{2 + \cfrac{5^2}{2 + \cfrac{7^2}{2 + \cdots}}}}$$

Gregrory's series:
$$\tfrac{\pi}{4} = 1 - \tfrac{1}{3} + \tfrac{1}{5} - \tfrac{1}{7} + \tfrac{1}{9} - \cdots$$

Newton's series:
$$\tfrac{\pi}{6} = \frac{1}{2} + \frac{1}{2 \cdot 3 \cdot 2^3} + \frac{1 \cdot 3}{2 \cdot 4 \cdot 5 \cdot 2^5} + \cdots$$

Sharp's series:
$$\tfrac{\pi}{6} = \frac{1}{\sqrt{3}}\left(1 - \frac{1}{3^1 \cdot 3} + \frac{1}{3^2 \cdot 5} - \frac{1}{3^3 \cdot 7} + \cdots\right)$$

Euler's series:
$$\tfrac{\pi^2}{6} = \tfrac{1}{1^2} + \tfrac{1}{2^2} + \tfrac{1}{3^2} + \tfrac{1}{4^2} + \tfrac{1}{5^2} + \cdots$$
$$\tfrac{\pi^2}{8} = \tfrac{1}{1^2} + \tfrac{1}{3^2} + \tfrac{1}{5^2} + \tfrac{1}{7^2} + \tfrac{1}{9^2} + \cdots$$
$$\tfrac{\pi^2}{12} = \tfrac{1}{1^2} - \tfrac{1}{2^2} + \tfrac{1}{3^2} - \tfrac{1}{4^2} + \tfrac{1}{5^2} - \cdots$$

### Partial Fractions

Let $N(x)$ and $D(x)$ be polynomial functions of $x$. We can break down $N(x)/D(x)$ using partial fraction expansion. First, if the degree of $N$ is greater than or equal to the degree of $D$, divide $N$ by $D$, obtaining
$$\frac{N(x)}{D(x)} = Q(x) + \frac{N'(x)}{D(x)},$$
where the degree of $N'$ is less than that of $D$. Second, factor $D(x)$. Use the following rules: For a non-repeated factor:
$$\frac{N(x)}{(x-a)D(x)} = \frac{A}{x-a} + \frac{N'(x)}{D(x)},$$
where
$$A = \left[\frac{N(x)}{D(x)}\right]_{x=a}.$$
For a repeated factor:
$$\frac{N(x)}{(x-a)^m D(x)} = \sum_{k=0}^{m-1} \frac{A_k}{(x-a)^{m-k}} + \frac{N'(x)}{D(x)},$$
where
$$A_k = \frac{1}{k!}\left[\frac{d^k}{dx^k}\left(\frac{N(x)}{D(x)}\right)\right]_{x=a}.$$

The reasonable man adapts himself to the world; the unreasonable persists in trying to adapt the world to himself. Therefore all progress depends on the unreasonable.
– George Bernard Shaw

### Calculus

Derivatives:

**1.** $\dfrac{d(cu)}{dx} = c\dfrac{du}{dx}$, **2.** $\dfrac{d(u+v)}{dx} = \dfrac{du}{dx} + \dfrac{dv}{dx}$, **3.** $\dfrac{d(uv)}{dx} = u\dfrac{dv}{dx} + v\dfrac{du}{dx}$,

**4.** $\dfrac{d(u^n)}{dx} = nu^{n-1}\dfrac{du}{dx}$, **5.** $\dfrac{d(u/v)}{dx} = \dfrac{v\left(\frac{du}{dx}\right) - u\left(\frac{dv}{dx}\right)}{v^2}$, **6.** $\dfrac{d(e^{cu})}{dx} = ce^{cu}\dfrac{du}{dx}$,

**7.** $\dfrac{d(c^u)}{dx} = (\ln c)c^u\dfrac{du}{dx}$, **8.** $\dfrac{d(\ln u)}{dx} = \dfrac{1}{u}\dfrac{du}{dx}$,

**9.** $\dfrac{d(\sin u)}{dx} = \cos u\dfrac{du}{dx}$, **10.** $\dfrac{d(\cos u)}{dx} = -\sin u\dfrac{du}{dx}$,

**11.** $\dfrac{d(\tan u)}{dx} = \sec^2 u\dfrac{du}{dx}$, **12.** $\dfrac{d(\cot u)}{dx} = \csc^2 u\dfrac{du}{dx}$,

**13.** $\dfrac{d(\sec u)}{dx} = \tan u \sec u\dfrac{du}{dx}$, **14.** $\dfrac{d(\csc u)}{dx} = -\cot u \csc u\dfrac{du}{dx}$,

**15.** $\dfrac{d(\arcsin u)}{dx} = \dfrac{1}{\sqrt{1-u^2}}\dfrac{du}{dx}$, **16.** $\dfrac{d(\arccos u)}{dx} = \dfrac{-1}{\sqrt{1-u^2}}\dfrac{du}{dx}$,

**17.** $\dfrac{d(\arctan u)}{dx} = \dfrac{1}{1+u^2}\dfrac{du}{dx}$, **18.** $\dfrac{d(\operatorname{arccot} u)}{dx} = \dfrac{-1}{1+u^2}\dfrac{du}{dx}$,

**19.** $\dfrac{d(\operatorname{arcsec} u)}{dx} = \dfrac{1}{u\sqrt{1-u^2}}\dfrac{du}{dx}$, **20.** $\dfrac{d(\operatorname{arccsc} u)}{dx} = \dfrac{-1}{u\sqrt{1-u^2}}\dfrac{du}{dx}$,

**21.** $\dfrac{d(\sinh u)}{dx} = \cosh u\dfrac{du}{dx}$, **22.** $\dfrac{d(\cosh u)}{dx} = \sinh u\dfrac{du}{dx}$,

**23.** $\dfrac{d(\tanh u)}{dx} = \operatorname{sech}^2 u\dfrac{du}{dx}$, **24.** $\dfrac{d(\coth u)}{dx} = -\operatorname{csch}^2 u\dfrac{du}{dx}$,

**25.** $\dfrac{d(\operatorname{sech} u)}{dx} = -\operatorname{sech} u \tanh u\dfrac{du}{dx}$, **26.** $\dfrac{d(\operatorname{csch} u)}{dx} = -\operatorname{csch} u \coth u\dfrac{du}{dx}$,

**27.** $\dfrac{d(\operatorname{arcsinh} u)}{dx} = \dfrac{1}{\sqrt{1+u^2}}\dfrac{du}{dx}$, **28.** $\dfrac{d(\operatorname{arccosh} u)}{dx} = \dfrac{1}{\sqrt{u^2-1}}\dfrac{du}{dx}$,

**29.** $\dfrac{d(\operatorname{arctanh} u)}{dx} = \dfrac{1}{1-u^2}\dfrac{du}{dx}$, **30.** $\dfrac{d(\operatorname{arccoth} u)}{dx} = \dfrac{1}{u^2-1}\dfrac{du}{dx}$,

**31.** $\dfrac{d(\operatorname{arcsech} u)}{dx} = \dfrac{-1}{u\sqrt{1-u^2}}\dfrac{du}{dx}$, **32.** $\dfrac{d(\operatorname{arccsch} u)}{dx} = \dfrac{-1}{|u|\sqrt{1+u^2}}\dfrac{du}{dx}$.

Integrals:

**1.** $\displaystyle\int cu\,dx = c\int u\,dx$, **2.** $\displaystyle\int (u+v)\,dx = \int u\,dx + \int v\,dx$,

**3.** $\displaystyle\int x^n\,dx = \frac{1}{n+1}x^{n+1}, \quad n \neq -1$, **4.** $\displaystyle\int \frac{1}{x}dx = \ln x$, **5.** $\displaystyle\int e^x\,dx = e^x$,

**6.** $\displaystyle\int \frac{dx}{1+x^2} = \arctan x$, **7.** $\displaystyle\int u\frac{dv}{dx}dx = uv - \int v\frac{du}{dx}dx$,

**8.** $\displaystyle\int \sin x\,dx = -\cos x$, **9.** $\displaystyle\int \cos x\,dx = \sin x$,

**10.** $\displaystyle\int \tan x\,dx = -\ln|\cos x|$, **11.** $\displaystyle\int \cot x\,dx = \ln|\cos x|$,

**12.** $\displaystyle\int \sec x\,dx = \ln|\sec x + \tan x|$, **13.** $\displaystyle\int \csc x\,dx = \ln|\csc x + \cot x|$,

**14.** $\displaystyle\int \arcsin\tfrac{x}{a}dx = \arcsin\tfrac{x}{a} + \sqrt{a^2-x^2}, \quad a > 0$,

| Theoretical Computer Science Cheat Sheet |
|---|
| Calculus Cont. |

**15.** $\int \arccos \frac{x}{a} dx = \arccos \frac{x}{a} - \sqrt{a^2 - x^2}, \quad a > 0,$

**16.** $\int \arctan \frac{x}{a} dx = x \arctan \frac{x}{a} - \frac{a}{2} \ln(a^2 + x^2), \quad a > 0,$

**17.** $\int \sin^2(ax) dx = \frac{1}{2a}\big(ax - \sin(ax)\cos(ax)\big),$

**18.** $\int \cos^2(ax) dx = \frac{1}{2a}\big(ax + \sin(ax)\cos(ax)\big),$

**19.** $\int \sec^2 x \, dx = \tan x,$

**20.** $\int \csc^2 x \, dx = -\cot x,$

**21.** $\int \sin^n x \, dx = -\frac{\sin^{n-1} x \cos x}{n} + \frac{n-1}{n} \int \sin^{n-2} x \, dx,$

**22.** $\int \cos^n x \, dx = \frac{\cos^{n-1} x \sin x}{n} + \frac{n-1}{n} \int \cos^{n-2} x \, dx,$

**23.** $\int \tan^n x \, dx = \frac{\tan^{n-1} x}{n-1} - \int \tan^{n-2} x \, dx, \quad n \neq 1,$

**24.** $\int \cot^n x \, dx = -\frac{\cot^{n-1} x}{n-1} - \int \cot^{n-2} x \, dx, \quad n \neq 1,$

**25.** $\int \sec^n x \, dx = \frac{\tan x \sec^{n-1} x}{n-1} + \frac{n-2}{n-1} \int \sec^{n-2} x \, dx, \quad n \neq 1,$

**26.** $\int \csc^n x \, dx = -\frac{\cot x \csc^{n-1} x}{n-1} + \frac{n-2}{n-1} \int \csc^{n-2} x \, dx, \quad n \neq 1,$

**27.** $\int \sinh x \, dx = \cosh x,$

**28.** $\int \cosh x \, dx = \sinh x,$

**29.** $\int \tanh x \, dx = \ln|\cosh x|,$

**30.** $\int \coth x \, dx = \ln|\sinh x|,$

**31.** $\int \operatorname{sech} x \, dx = \arctan \sinh x,$

**32.** $\int \operatorname{csch} x \, dx = \ln\left|\tanh \frac{x}{2}\right|,$

**33.** $\int \sinh^2 x \, dx = \frac{1}{4}\sinh(2x) - \frac{1}{2}x,$

**34.** $\int \cosh^2 x \, dx = \frac{1}{4}\sinh(2x) + \frac{1}{2}x,$

**35.** $\int \operatorname{sech}^2 x \, dx = \tanh x,$

**36.** $\int \operatorname{arcsinh} \frac{x}{a} dx = x \operatorname{arcsinh} \frac{x}{a} - \sqrt{x^2 + a^2}, \quad a > 0,$

**37.** $\int \operatorname{arctanh} \frac{x}{a} dx = x \operatorname{arctanh} \frac{x}{a} + \frac{a}{2} \ln|a^2 - x^2|,$

**38.** $$\int \operatorname{arccosh} \frac{x}{a} dx = \begin{cases} x \operatorname{arccosh} \dfrac{x}{a} - \sqrt{x^2 + a^2}, & \text{if } \operatorname{arccosh} \frac{x}{a} > 0 \text{ and } a > 0, \\ x \operatorname{arccosh} \dfrac{x}{a} + \sqrt{x^2 + a^2}, & \text{if } \operatorname{arccosh} \frac{x}{a} < 0 \text{ and } a > 0, \end{cases}$$

**39.** $\int \frac{dx}{\sqrt{a^2 + x^2}} = \ln\left(x + \sqrt{a^2 + x^2}\right), \quad a > 0,$

**40.** $\int \frac{dx}{a^2 + x^2} = \frac{1}{a} \arctan \frac{x}{a}, \quad a > 0,$

**41.** $\int \sqrt{a^2 - x^2} \, dx = \frac{x}{2}\sqrt{a^2 - x^2} + \frac{a^2}{2} \arcsin \frac{x}{a}, \quad a > 0,$

**42.** $\int (a^2 - x^2)^{3/2} dx = \frac{x}{8}(5a^2 - 2x^2)\sqrt{a^2 - x^2} + \frac{3a^4}{8} \arcsin \frac{x}{a}, \quad a > 0,$

**43.** $\int \frac{dx}{\sqrt{a^2 - x^2}} = \arcsin \frac{x}{a}, \quad a > 0,$

**44.** $\int \frac{dx}{a^2 - x^2} = \frac{1}{2a} \ln\left|\frac{a + x}{a - x}\right|,$

**45.** $\int \frac{dx}{(a^2 - x^2)^{3/2}} = \frac{x}{a^2\sqrt{a^2 - x^2}},$

**46.** $\int \sqrt{a^2 \pm x^2} \, dx = \frac{x}{2}\sqrt{a^2 \pm x^2} \pm \frac{a^2}{2} \ln\left|x + \sqrt{a^2 \pm x^2}\right|,$

**47.** $\int \frac{dx}{\sqrt{x^2 - a^2}} = \ln\left|x + \sqrt{x^2 - a^2}\right|, \quad a > 0,$

**48.** $\int \frac{dx}{ax^2 + bx} = \frac{1}{a} \ln\left|\frac{x}{a + bx}\right|,$

**49.** $\int x\sqrt{a + bx} \, dx = \frac{2(3bx - 2a)(a + bx)^{3/2}}{15b^2},$

**50.** $\int \frac{\sqrt{a + bx}}{x} dx = 2\sqrt{a + bx} + a \int \frac{1}{x\sqrt{a + bx}} dx,$

**51.** $\int \frac{x}{\sqrt{a + bx}} dx = \frac{1}{\sqrt{2}} \ln\left|\frac{\sqrt{a + bx} - \sqrt{a}}{\sqrt{a + bx} + \sqrt{a}}\right|, \quad a > 0,$

**52.** $\int \frac{\sqrt{a^2 - x^2}}{x} dx = \sqrt{a^2 - x^2} - a \ln\left|\frac{a + \sqrt{a^2 - x^2}}{x}\right|,$

**53.** $\int x\sqrt{a^2 - x^2} \, dx = -\frac{1}{3}(a^2 - x^2)^{3/2},$

**54.** $\int x^2\sqrt{a^2 - x^2} \, dx = \frac{x}{8}(2x^2 - a^2)\sqrt{a^2 - x^2} + \frac{a^4}{8} \arcsin \frac{x}{a}, \quad a > 0,$

**55.** $\int \frac{dx}{\sqrt{a^2 - x^2}} = -\frac{1}{a} \ln\left|\frac{a + \sqrt{a^2 - x^2}}{x}\right|,$

**56.** $\int \frac{x \, dx}{\sqrt{a^2 - x^2}} = -\sqrt{a^2 - x^2},$

**57.** $\int \frac{x^2 \, dx}{\sqrt{a^2 - x^2}} = -\frac{x}{2}\sqrt{a^2 - x^2} + \frac{a^2}{2} \arcsin \frac{x}{a}, \quad a > 0,$

**58.** $\int \frac{\sqrt{a^2 + x^2}}{x} dx = \sqrt{a^2 + x^2} - a \ln\left|\frac{a + \sqrt{a^2 + x^2}}{x}\right|,$

**59.** $\int \frac{\sqrt{x^2 - a^2}}{x} dx = \sqrt{x^2 - a^2} - a \arccos \frac{a}{|x|}, \quad a > 0,$

**60.** $\int x\sqrt{x^2 \pm a^2} \, dx = \frac{1}{3}(x^2 \pm a^2)^{3/2},$

**61.** $\int \frac{dx}{x\sqrt{x^2 + a^2}} = \frac{1}{a} \ln\left|\frac{x}{a + \sqrt{a^2 + x^2}}\right|,$

# Theoretical Computer Science Cheat Sheet

## Calculus Cont.

**62.** $\displaystyle\int \frac{dx}{x\sqrt{x^2-a^2}} = \tfrac{1}{a}\arccos\tfrac{a}{|x|}, \quad a>0,$

**63.** $\displaystyle\int \frac{dx}{x^2\sqrt{x^2\pm a^2}} = \mp\frac{\sqrt{x^2\pm a^2}}{a^2x},$

**64.** $\displaystyle\int \frac{x\,dx}{\sqrt{x^2\pm a^2}} = \sqrt{x^2\pm a^2},$

**65.** $\displaystyle\int \frac{\sqrt{x^2\pm a^2}}{x^4}\,dx = \mp\frac{(x^2+a^2)^{3/2}}{3a^2x^3},$

**66.** $$\int \frac{dx}{ax^2+bx+c} = \begin{cases} \dfrac{1}{\sqrt{b^2-4ac}}\ln\left|\dfrac{2ax+b-\sqrt{b^2-4ac}}{2ax+b+\sqrt{b^2-4ac}}\right|, & \text{if } b^2>4ac, \\ \dfrac{2}{\sqrt{4ac-b^2}}\arctan\dfrac{2ax+b}{\sqrt{4ac-b^2}}, & \text{if } b^2<4ac, \end{cases}$$

**67.** $$\int \frac{dx}{\sqrt{ax^2+bx+c}} = \begin{cases} \dfrac{1}{\sqrt{a}}\ln\left|2ax+b+2\sqrt{a}\sqrt{ax^2+bx+c}\right|, & \text{if } a>0, \\ \dfrac{1}{\sqrt{-a}}\arcsin\dfrac{-2ax-b}{\sqrt{b^2-4ac}}, & \text{if } a<0, \end{cases}$$

**68.** $$\int \sqrt{ax^2+bx+c}\,dx = \frac{2ax+b}{4a}\sqrt{ax^2+bx+c} + \frac{4ax-b^2}{8a}\int\frac{dx}{\sqrt{ax^2+bx+c}},$$

**69.** $$\int \frac{x\,dx}{\sqrt{ax^2+bx+c}} = \frac{\sqrt{ax^2+bx+c}}{a} - \frac{b}{2a}\int\frac{dx}{\sqrt{ax^2+bx+c}},$$

**70.** $$\int \frac{dx}{x\sqrt{ax^2+bx+c}} = \begin{cases} \dfrac{-1}{\sqrt{c}}\ln\left|\dfrac{2\sqrt{c}\sqrt{ax^2+bx+c}+bx+2c}{x}\right|, & \text{if } c>0, \\ \dfrac{1}{\sqrt{-c}}\arcsin\dfrac{bx+2c}{|x|\sqrt{b^2-4ac}}, & \text{if } c<0, \end{cases}$$

**71.** $\displaystyle\int x^3\sqrt{x^2+a^2}\,dx = (\tfrac{1}{3}x^2-\tfrac{2}{15}a^2)(x^2+a^2)^{3/2},$

**72.** $\displaystyle\int x^n\sin(ax)\,dx = -\tfrac{1}{a}x^n\cos(ax) + \tfrac{n}{a}\int x^{n-1}\cos(ax)\,dx,$

**73.** $\displaystyle\int x^n\cos(ax)\,dx = \tfrac{1}{a}x^n\sin(ax) - \tfrac{n}{a}\int x^{n-1}\sin(ax)\,dx,$

**74.** $\displaystyle\int x^ne^{ax}\,dx = \frac{x^ne^{ax}}{a} - \frac{n}{a}\int x^{n-1}e^{ax}\,dx,$

**75.** $\displaystyle\int x^n\ln(ax)\,dx = x^{n+1}\left(\frac{\ln(ax)}{n+1} - \frac{1}{(n+1)^2}\right),$

**76.** $\displaystyle\int x^n(\ln ax)^m\,dx = \frac{x^{n+1}}{n+1}(\ln ax)^m - \frac{m}{n+1}\int x^n(\ln ax)^{m-1}\,dx.$

$$\begin{aligned}
x^1 &= x^{\underline{1}} &&= x^{\overline{1}} \\
x^2 &= x^{\underline{2}}+x^{\underline{1}} &&= x^{\overline{2}}-x^{\overline{1}} \\
x^3 &= x^{\underline{3}}+3x^{\underline{2}}+x^{\underline{1}} &&= x^{\overline{3}}-3x^{\overline{2}}+x^{\overline{1}} \\
x^4 &= x^{\underline{4}}+6x^{\underline{3}}+7x^{\underline{2}}+x^{\underline{1}} &&= x^{\overline{4}}-6x^{\overline{3}}+7x^{\overline{2}}-x^{\overline{1}} \\
x^5 &= x^{\underline{5}}+15x^{\underline{4}}+25x^{\underline{3}}+10x^{\underline{2}}+x^{\underline{1}} &&= x^{\underline{5}}-15x^{\overline{4}}+25x^{\overline{3}}-10x^{\overline{2}}+x^{\overline{1}}
\end{aligned}$$

$$\begin{aligned}
x^{\overline{1}} &= x^1 & x^{\underline{1}} &= x^1 \\
x^{\overline{2}} &= x^2+x^1 & x^{\underline{2}} &= x^2-x^1 \\
x^{\overline{3}} &= x^3+3x^2+2x^1 & x^{\underline{3}} &= x^3-3x^2+2x^1 \\
x^{\overline{4}} &= x^4+6x^3+11x^2+6x^1 & x^{\underline{4}} &= x^4-6x^3+11x^2-6x^1 \\
x^{\overline{5}} &= x^5+10x^4+35x^3+50x^2+24x^1 & x^{\underline{5}} &= x^5-10x^4+35x^3-50x^2+24x^1
\end{aligned}$$

## Finite Calculus

Difference, shift operators:

$$\Delta f(x) = f(x+1) - f(x),$$
$$\mathrm{E}\,f(x) = f(x+1).$$

Fundamental Theorem:

$$f(x) = \Delta F(x) \Leftrightarrow \sum f(x)\delta x = F(x) + C.$$
$$\sum_a^b f(x)\delta x = \sum_{i=a}^{b-1} f(i).$$

Differences:

$\Delta(cu) = c\Delta u, \qquad \Delta(u+v) = \Delta u + \Delta v,$

$\Delta(uv) = u\Delta v + \mathrm{E}\,v\Delta u,$

$\Delta(x^{\underline{n}}) = nx^{\underline{n-1}},$

$\Delta(H_x) = x^{\underline{-1}}, \qquad \Delta(2^x) = 2^x,$

$\Delta(c^x) = (c-1)c^x, \qquad \Delta\binom{x}{m} = \binom{x}{m-1}.$

Sums:

$\sum cu\,\delta x = c\sum u\,\delta x,$

$\sum(u+v)\,\delta x = \sum u\,\delta x + \sum v\,\delta x,$

$\sum u\Delta v\,\delta x = uv - \sum \mathrm{E}\,v\Delta u\,\delta x,$

$\sum x^{\underline{n}}\,\delta x = \frac{x^{\underline{n+1}}}{m+1}, \qquad \sum x^{\underline{-1}}\,\delta x = H_x,$

$\sum c^x\,\delta x = \frac{c^x}{c-1}, \qquad \sum\binom{x}{m}\,\delta x = \binom{x}{m+1}.$

Falling Factorial Powers:

$x^{\underline{n}} = x(x-1)\cdots(x-n+1), \quad n>0,$

$x^{\underline{0}} = 1,$

$x^{\underline{n}} = \dfrac{1}{(x+1)\cdots(x+|n|)}, \quad n<0,$

$x^{\underline{n+m}} = x^{\underline{m}}(x-m)^{\underline{n}}.$

Rising Factorial Powers:

$x^{\overline{n}} = x(x+1)\cdots(x+n-1), \quad n>0,$

$x^{\overline{0}} = 1,$

$x^{\overline{n}} = \dfrac{1}{(x-1)\cdots(x-|n|)}, \quad n<0,$

$x^{\overline{n+m}} = x^{\overline{m}}(x+m)^{\overline{n}}.$

Conversion:

$x^{\underline{n}} = (-1)^n(-x)^{\overline{n}} = (x-n+1)^{\overline{n}}$
$\quad = 1/(x+1)^{\overline{-n}},$

$x^{\overline{n}} = (-1)^n(-x)^{\underline{n}} = (x+n-1)^{\underline{n}}$
$\quad = 1/(x-1)^{\underline{-n}},$

$$x^n = \sum_{k=1}^n \left\{ n \atop k \right\} x^{\underline{k}} = \sum_{k=1}^n \left\{ n \atop k \right\}(-1)^{n-k}x^{\overline{k}},$$

$$x^{\underline{n}} = \sum_{k=1}^n \left[ n \atop k \right](-1)^{n-k}x^k,$$

$$x^{\overline{n}} = \sum_{k=1}^n \left[ n \atop k \right]x^k.$$

## Theoretical Computer Science Cheat Sheet

### Series

Taylor's series:

$$f(x) = f(a) + (x-a)f'(a) + \frac{(x-a)^2}{2}f''(a) + \cdots = \sum_{i=0}^{\infty} \frac{(x-a)^i}{i!} f^{(i)}(a).$$

Expansions:

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + x^4 + \cdots = \sum_{i=0}^{\infty} x^i,$$

$$\frac{1}{1-cx} = 1 + cx + c^2x^2 + c^3x^3 + \cdots = \sum_{i=0}^{\infty} c^i x^i,$$

$$\frac{1}{1-x^n} = 1 + x^n + x^{2n} + x^{3n} + \cdots = \sum_{i=0}^{\infty} x^{ni},$$

$$\frac{x}{(1-x)^2} = x + 2x^2 + 3x^3 + 4x^4 + \cdots = \sum_{i=0}^{\infty} i x^i,$$

$$x^k \frac{d^n}{dx^n}\left(\frac{1}{1-x}\right) = x + 2^n x^2 + 3^n x^3 + 4^n x^4 + \cdots = \sum_{i=0}^{\infty} i^n x^i,$$

$$e^x = 1 + x + \tfrac{1}{2}x^2 + \tfrac{1}{6}x^3 + \cdots = \sum_{i=0}^{\infty} \frac{x^i}{i!},$$

$$\ln(1+x) = x - \tfrac{1}{2}x^2 + \tfrac{1}{3}x^3 - \tfrac{1}{4}x^4 - \cdots = \sum_{i=1}^{\infty} (-1)^{i+1}\frac{x^i}{i},$$

$$\ln\frac{1}{1-x} = x + \tfrac{1}{2}x^2 + \tfrac{1}{3}x^3 + \tfrac{1}{4}x^4 + \cdots = \sum_{i=1}^{\infty} \frac{x^i}{i},$$

$$\sin x = x - \tfrac{1}{3!}x^3 + \tfrac{1}{5!}x^5 - \tfrac{1}{7!}x^7 + \cdots = \sum_{i=0}^{\infty} (-1)^i \frac{x^{2i+1}}{(2i+1)!},$$

$$\cos x = 1 - \tfrac{1}{2!}x^2 + \tfrac{1}{4!}x^4 - \tfrac{1}{6!}x^6 + \cdots = \sum_{i=0}^{\infty} (-1)^i \frac{x^{2i}}{(2i)!},$$

$$\tan^{-1} x = x - \tfrac{1}{3}x^3 + \tfrac{1}{5}x^5 - \tfrac{1}{7}x^7 + \cdots = \sum_{i=0}^{\infty} (-1)^i \frac{x^{2i+1}}{(2i+1)},$$

$$(1+x)^n = 1 + nx + \tfrac{n(n-1)}{2}x^2 + \cdots = \sum_{i=0}^{\infty} \binom{n}{i} x^i,$$

$$\frac{1}{(1-x)^{n+1}} = 1 + (n+1)x + \tbinom{n+2}{2}x^2 + \cdots = \sum_{i=0}^{\infty} \binom{i+n}{i} x^i,$$

$$\frac{x}{e^x - 1} = 1 - \tfrac{1}{2}x + \tfrac{1}{12}x^2 - \tfrac{1}{720}x^4 + \cdots = \sum_{i=0}^{\infty} \frac{B_i x^i}{i!},$$

$$\frac{1}{2x}(1 - \sqrt{1-4x}) = 1 + x + 2x^2 + 5x^3 + \cdots = \sum_{i=0}^{\infty} \frac{1}{i+1}\binom{2i}{i} x^i,$$

$$\frac{1}{\sqrt{1-4x}} = 1 + x + 2x^2 + 6x^3 + \cdots = \sum_{i=0}^{\infty} \binom{2i}{i} x^i,$$

$$\frac{1}{\sqrt{1-4x}}\left(\frac{1-\sqrt{1-4x}}{2x}\right)^n = 1 + (2+n)x + \tbinom{4+n}{2}x^2 + \cdots = \sum_{i=0}^{\infty} \binom{2i+n}{i} x^i,$$

$$\frac{1}{1-x}\ln\frac{1}{1-x} = x + \tfrac{3}{2}x^2 + \tfrac{11}{6}x^3 + \tfrac{25}{12}x^4 + \cdots = \sum_{i=1}^{\infty} H_i x^i,$$

$$\frac{1}{2}\left(\ln\frac{1}{1-x}\right)^2 = \tfrac{1}{2}x^2 + \tfrac{3}{4}x^3 + \tfrac{11}{24}x^4 + \cdots = \sum_{i=2}^{\infty} \frac{H_{i-1}x^i}{i},$$

$$\frac{x}{1-x-x^2} = x + x^2 + 2x^3 + 3x^4 + \cdots = \sum_{i=0}^{\infty} F_i x^i,$$

$$\frac{F_n x}{1 - (F_{n-1} + F_{n+1})x - (-1)^n x^2} = F_n x + F_{2n}x^2 + F_{3n}x^3 + \cdots = \sum_{i=0}^{\infty} F_{ni} x^i.$$

Ordinary power series:

$$A(x) = \sum_{i=0}^{\infty} a_i x^i.$$

Exponential power series:

$$A(x) = \sum_{i=0}^{\infty} a_i \frac{x^i}{i!}.$$

Dirichlet power series:

$$A(x) = \sum_{i=1}^{\infty} \frac{a_i}{i^x}.$$

Binomial theorem:

$$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k.$$

Difference of like powers:

$$x^n - y^n = (x-y)\sum_{k=0}^{n-1} x^{n-1-k} y^k.$$

For ordinary power series:

$$\alpha A(x) + \beta B(x) = \sum_{i=0}^{\infty} (\alpha a_i + \beta b_i) x^i,$$

$$x^k A(x) = \sum_{i=k}^{\infty} a_{i-k} x^i,$$

$$\frac{A(x) - \sum_{i=0}^{k-1} a_i x^i}{x^k} = \sum_{i=0}^{\infty} a_{i+k} x^i,$$

$$A(cx) = \sum_{i=0}^{\infty} c^i a_i x^i,$$

$$A'(x) = \sum_{i=0}^{\infty} (i+1) a_{i+1} x^i,$$

$$xA'(x) = \sum_{i=1}^{\infty} i a_i x^i,$$

$$\int A(x)\,dx = \sum_{i=1}^{\infty} \frac{a_{i-1}}{i} x^i,$$

$$\frac{A(x) + A(-x)}{2} = \sum_{i=0}^{\infty} a_{2i} x^{2i},$$

$$\frac{A(x) - A(-x)}{2} = \sum_{i=0}^{\infty} a_{2i+1} x^{2i+1}.$$

Summation: If $b_i = \sum_{j=0}^{i} a_i$ then

$$B(x) = \frac{1}{1-x} A(x).$$

Convolution:

$$A(x)B(x) = \sum_{i=0}^{\infty} \left(\sum_{j=0}^{i} a_j b_{i-j}\right) x^i.$$

God made the natural numbers;
all the rest is the work of man.
– Leopold Kronecker

# Theoretical Computer Science Cheat Sheet

## Series

Expansions:

$$\frac{1}{(1-x)^{n+1}}\ln\frac{1}{1-x} = \sum_{i=0}^{\infty}(H_{n+i}-H_n)\binom{n+i}{i}x^i, \qquad \left(\frac{1}{x}\right)^{\overline{-n}} = \sum_{i=0}^{\infty}\left\{ {i \atop n} \right\}x^i,$$

$$x^{\overline{n}} = \sum_{i=0}^{\infty}\left[ {n \atop i} \right]x^i, \qquad (e^x-1)^n = \sum_{i=0}^{\infty}\left\{ {i \atop n} \right\}\frac{n!x^i}{i!},$$

$$\left(\ln\frac{1}{1-x}\right)^n = \sum_{i=0}^{\infty}\left[ {i \atop n} \right]\frac{n!x^i}{i!}, \qquad x\cot x = \sum_{i=0}^{\infty}\frac{(-4)^iB_{2i}x^{2i}}{(2i)!},$$

$$\tan x = \sum_{i=1}^{\infty}(-1)^{i-1}\frac{2^{2i}(2^{2i}-1)B_{2i}x^{2i-1}}{(2i)!}, \qquad \zeta(x) = \sum_{i=1}^{\infty}\frac{1}{i^x},$$

$$\frac{1}{\zeta(x)} = \sum_{i=1}^{\infty}\frac{\mu(i)}{i^x}, \qquad \frac{\zeta(x-1)}{\zeta(x)} = \sum_{i=1}^{\infty}\frac{\phi(i)}{i^x},$$

$$\zeta(x) = \prod_p \frac{1}{1-p^{-x}},$$

$$\zeta^2(x) = \sum_{i=1}^{\infty}\frac{d(i)}{x^i} \quad \text{where } d(n) = \textstyle\sum_{d|n} 1,$$

$$\zeta(x)\zeta(x-1) = \sum_{i=1}^{\infty}\frac{S(i)}{x^i} \quad \text{where } S(n) = \textstyle\sum_{d|n} d,$$

$$\zeta(2n) = \frac{2^{2n-1}|B_{2n}|}{(2n)!}\pi^{2n}, \quad n \in \mathbb{N},$$

$$\frac{x}{\sin x} = \sum_{i=0}^{\infty}(-1)^{i-1}\frac{(4^i-2)B_{2i}x^{2i}}{(2i)!},$$

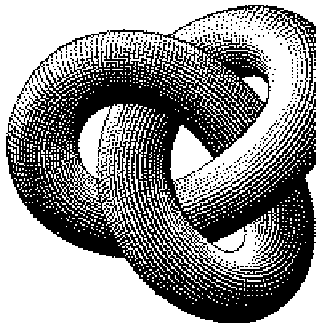
$$\left(\frac{1-\sqrt{1-4x}}{2x}\right)^n = \sum_{i=0}^{\infty}\frac{n(2i+n-1)!}{i!(n+i)!}x^i,$$

$$e^x\sin x = \sum_{i=1}^{\infty}\frac{2^{i/2}\sin\frac{i\pi}{4}}{i!}x^i,$$

$$\sqrt{\frac{1-\sqrt{1-x}}{x}} = \sum_{i=0}^{\infty}\frac{(4i)!}{16^i\sqrt{2}(2i)!(2i+1)!}x^i,$$

$$\left(\frac{\arcsin x}{x}\right)^2 = \sum_{i=0}^{\infty}\frac{4^ii!^2}{(i+1)(2i+1)!}x^{2i}.$$

## Escher's Knot



## Stieltjes Integration

If $G$ is continuous in the interval $[a,b]$ and $F$ is nondecreasing then

$$\int_a^b G(x)\,dF(x)$$

exists. If $a \le b \le c$ then

$$\int_a^c G(x)\,dF(x) = \int_a^b G(x)\,dF(x) + \int_b^c G(x)\,dF(x).$$

If the integrals involved exist

$$\int_a^b \big(G(x)+H(x)\big)\,dF(x) = \int_a^b G(x)\,dF(x) + \int_a^b H(x)\,dF(x),$$

$$\int_a^b G(x)\,d\big(F(x)+H(x)\big) = \int_a^b G(x)\,dF(x) + \int_a^b G(x)\,dH(x),$$

$$\int_a^b c\cdot G(x)\,dF(x) = \int_a^b G(x)\,d\big(c\cdot F(x)\big) = c\int_a^b G(x)\,dF(x),$$

$$\int_a^b G(x)\,dF(x) = G(b)F(b) - G(a)F(a) - \int_a^b F(x)\,dG(x).$$

If the integrals involved exist, and $F$ possesses a derivative $F'$ at every point in $[a,b]$ then

$$\int_a^b G(x)\,dF(x) = \int_a^b G(x)F'(x)\,dx.$$

## Cramer's Rule

If we have equations:

$$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n &= b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n &= b_2 \\ \vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \quad & \\ a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,n}x_n &= b_n \end{aligned}$$

Let $A = (a_{i,j})$ and $B$ be the column matrix $(b_i)$. Then there is a unique solution iff $\det A \neq 0$. Let $A_i$ be $A$ with column $i$ replaced by $B$. Then

$$x_i = \frac{\det A_i}{\det A}.$$

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 00 | 47 | 18 | 76 | 29 | 93 | 85 | 34 | 61 | 52 |
| 86 | 11 | 57 | 28 | 70 | 39 | 94 | 45 | 02 | 63 |
| 95 | 80 | 22 | 67 | 38 | 71 | 49 | 56 | 13 | 04 |
| 59 | 96 | 81 | 33 | 07 | 48 | 72 | 60 | 24 | 15 |
| 73 | 69 | 90 | 82 | 44 | 17 | 58 | 01 | 35 | 26 |
| 68 | 74 | 09 | 91 | 83 | 55 | 27 | 12 | 46 | 30 |
| 37 | 08 | 75 | 19 | 92 | 84 | 66 | 23 | 50 | 41 |
| 14 | 25 | 36 | 40 | 51 | 62 | 03 | 77 | 88 | 99 |
| 21 | 32 | 43 | 54 | 65 | 06 | 10 | 89 | 97 | 78 |
| 42 | 53 | 64 | 05 | 16 | 20 | 31 | 98 | 79 | 87 |

## Fibonacci Numbers

$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$

Definitions:

$$F_i = F_{i-1}+F_{i-2}, \quad F_0 = F_1 = 1,$$

$$F_{-i} = (-1)^{i-1}F_i,$$

$$F_i = \tfrac{1}{\sqrt{5}}\left(\phi^i - \hat{\phi}^i\right),$$

Cassini's identity: for $i > 0$:

$$F_{i+1}F_{i-1} - F_i^2 = (-1)^i.$$

Additive rule:

$$F_{n+k} = F_kF_{n+1} + F_{k-1}F_n,$$

$$F_{2n} = F_nF_{n+1} + F_{n-1}F_n.$$

Calculation by matrices:

$$\begin{pmatrix} F_{n-2} & F_{n-1} \\ F_{n-1} & F_n \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}^n.$$

The Fibonacci number system: Every integer $n$ has a unique representation

$$n = F_{k_1} + F_{k_2} + \cdots + F_{k_m},$$

where $k_i \ge k_{i+1} + 2$ for all $i$, $1 \le i < m$ and $k_m \ge 2$.

Improvement makes strait roads, but the crooked roads without Improvement, are roads of Genius.
– William Blake (The Marriage of Heaven and Hell)