字符串

kmp

```cpp
std::vector<int> get_next(std::string& t) {
    std::vector<int> next(t.size());
    next[0] = -1;
    for (int i = 0, j = -1; i < (int)t.size();) {
        if (j == -1 || t[i] == t[j]) {
            ++i, ++j;
            next[i] = j;
        }
        else
            j = next[j];
    }
    return next;
}
```

kmp

```cpp
bool kmp(std::string& s, std::string& t) {
    if (t.length() > s.length())return false;
    auto next = get_next(t);

    for (int i = 0, j = 0; i < (int)s.size() && j < (int)t.size();) {
        if (j == -1 || s[i] == t[j]) {
            ++i, ++j;
        }
        else
            j = next[j];
        if (j == (int)t.size())return true;
    }
    return false;
}
```

z函数

```cpp
std::vector<int> z_function(std::string s) {
    int n = (int)s.length();
    std::vector<int> z(n);
    for (int i = 1, l = 0, r = 0; i < n; ++i) {
        if (i <= r && z[i - l] < r - i + 1) {
            z[i] = z[i - l];
        }
        else {
            z[i] = std::max(0, r - i + 1);
            while (i + z[i] < n && s[z[i]] == s[i + z[i]]) ++z[i];
        }
        if (i + z[i] - 1 > r) l = i, r = i + z[i] - 1;
    }
    return z;
}
```

AC自动机

```cpp
struct AhoCorasick {
    static constexpr int ALPHABET = 26;
    struct Node {
        int len;
        int link;
        std::array<int, ALPHABET> next;
        Node() : len{ 0 }, link{ 0 }, next{} {}
    };

    std::vector<Node> t;

    AhoCorasick() {
        init();
    }

    void init() {
        t.assign(2, Node());
        t[0].next.fill(1);
        t[0].len = -1;
    }

    int newNode() {
        t.emplace_back();
        return t.size() - 1;
    }

    int add(const std::string& a) {
        int p = 1;
        for (auto c : a) {
            int x = c - 'a';
            if (t[p].next[x] == 0) {
                t[p].next[x] = newNode();
                t[t[p].next[x]].len = t[p].len + 1;
            }
            p = t[p].next[x];
        }
        return p;
    }

    void get_fail() {
```

```
    std::queue<int> q;
    q.push(1);

    while (!q.empty()) {
        int x = q.front();
        q.pop();

        for (int i = 0; i < ALPHABET; i++) {
            if (t[x].next[i] == 0) {
                t[x].next[i] = t[t[x].link].next[i];
            }
            else {
                t[t[x].next[i]].link = t[t[x].link].next[i];
                q.push(t[x].next[i]);
            }
        }
    }
}

std::vector<int> work(std::string s) {
    get_fail();
    int p = 1;
    std::vector<int> f(t.size());
    for (auto c : s) {
        p = next(p, c - 'a');
        f[p]++;
    }

    std::vector<std::vector<int>> adj(t.size());
    for (int i = 2; i < t.size(); i++) {
        adj[link(i)].push_back(i);
    }

    std::function<void(int)> dfs = [&](int x) -> void {
        for (auto y : adj[x]) {
            dfs(y);
            f[x] += f[y];
        }
        };
    dfs(1);
    return f;
}
```

```cpp
    int next(int p, int x) {
        return t[p].next[x];
    }

    int link(int p) {
        return t[p].link;
    }

    int len(int p) {
        return t[p].len;
    }

    int size() {
        return t.size();
    }
};
```

马拉车

```cpp
struct Manachar {
    std::vector<int> d1, d2;
    Manachar(std::string s) {
        int n = s.length();
        d1.assign(n, 0);
        d2.assign(n, 0);
        for (int i = 0, l = 0, r = -1;i < n;++i) {
            int k = (i > r) ? 1 : std::min(d1[l + r - i], r - i + 1);
            while (i + k < n && i - k >= 0 && s[i + k] == s[i - k])k++;
            d1[i] = k--;
            if (i + k > r) {
                r = i + k;
                l = i - k;
            }
        }
        for (int i = 0, l = 0, r = -1;i < n;++i) {
            int k = (i > r) ? 0 : std::min(d2[l + r - i + 1], r - i + 1);
            while (i + k < n && i - k - 1 >= 0 && s[i + k] == s[i - k - 1])k++;
            d2[i] = k--;
            if (i + k > r) {
                r = i + k;
                l = i - k - 1;
            }
        }
    }
    bool check(int l, int r) {
        if (r < l)return false;
        int len = r - l + 1;
        if (len % 2) {
            return d1[l + len / 2] * 2 - 1 < len;
        }
        else {
            return d2[l + len / 2] * 2 < len;
        }
    }
};
```

双模哈希 哈希字符串 树上哈希 取随机数

```cpp
#define M1 998244853
#define M2 1000000009
#define N 500000

i64 qpow(i64 x, i64 p, i64 mod) {
    i64 ret = 1;
    while (p) {
        if (p & 1)ret = ret * x % mod;
        p >>= 1;
        x = x * x % mod;
    }
    return ret;
}

struct hsh {
    i64 w1, w2;
    hsh operator * (const int w) {
        return { w1 * w % M1,w2 * w % M2 };
    }
    hsh operator * (const hsh w) {
        return { w1 * w.w1 % M1,w2 * w.w2 % M2 };
    }
    hsh operator + (const hsh w) {
        return { (w1 + w.w1) % M1,(w2 + w.w2) % M2 };
    }
    hsh operator - (const hsh w) {
        return { (w1 + M1 - w.w1) % M1,(w2 + M2 - w.w2) % M2 };
    }
    bool operator == (const hsh w) {
        return (w1 == w.w1) && (w2 == w.w2);
    }
    i64 wt() {
        return M2 * w1 + w2;
    }
    void show() { std::cout << w1 << ' ' << w2 << '\n'; }
}pw[N + 50], inv[N + 50];

std::mt19937_64 rng(std::chrono::steady_clock::now().time_since_epoch().count());

int dep[MAXN], lg[MAXN], p[MAXN][30];
```

```cpp
void init() {
    int b1 = rng() % M1 + 1, b2 = rng() % M2 + 1;
    pw[0] = inv[0] = { 1,1 };
    pw[1] = { b1,b2 };
    inv[1] = { qpow(b1,M1 - 2,M1),qpow(b2,M2 - 2,M2) };
    for (int i = 2;i <= N;i++) {
        pw[i] = pw[i - 1] * pw[1];
        inv[i] = inv[i - 1] * inv[1];
    }
    for (int i = 1;i <= N;++i)
        lg[i] = lg[i >> 1] + 1;
}

int lca(int x, int y) {
    if (dep[x] < dep[y])std::swap(x, y);
    while (dep[x] > dep[y])
        x = p[x][lg[dep[x] - dep[y]] - 1];
    if (x == y)return x;
    for (int k = lg[dep[x]] - 1;k >= 0;--k)
        if (p[x][k] != p[y][k])
            x = p[x][k], y = p[y][k];
    return p[x][0];
}

hsh h1[MAXN], h2[MAXN];
int Fa[MAXN];
std::string s;
void dfs(int x, int par) {
    Fa[x] = par;
    p[x][0] = par;
    dep[x] = dep[par] + 1;
    h1[x] = h1[par] + pw[dep[x]] * s[x];
    h2[x] = h2[par] + inv[dep[x]] * s[x];
    for (int i = 1;i <= lg[dep[x]];++i)
        p[x][i] = p[p[x][i - 1]][i - 1];
    for (int nxt : adj[x])if (nxt != par)dfs(nxt, x);
}
```