# 字符串

## kmp pfunc

```
vector<int> pfunc(string s) {
   int n = s.size();
   vector<int> p(n);
   for (int i = 1; i < n; i ++) {
      int j = p[i - 1];
      while (j && s[j] != s[i]) {
         j = p[j - 1];
      }
      j += s[j] == s[i];
      p[i] = j;
   }
   return p;
}</pre>
```

### 马拉车

```
std::vector<int> manacher(std::string s) {
    std::string t = "#";
    for (auto c : s) {
        t += c;
        t += '#';
    }
    int n = t.size();
    std::vector<int> r(n);
    for (int i = 0, j = 0; i < n; i++) {
        if (2 * j - i >= 0 && j + r[j] > i) {
            r[i] = std::min(r[2 * j - i], j + r[j] - i);
        }
        while (i - r[i] >= 0 && i + r[i] < n && t[i - r[i]] == t[i + r[i]]) {
            r[i] += 1;
        }
        if (i + r[i] > j + r[j]) {
            j = i;
        }
        reconstants;
        reconstants;
```

```
}
return r;
}
```

#### zfunc

```
std::vector<int> zFunction(std::string s) {
    int n = s.size();
    std::vector<int> z(n + 1);
    z[0] = n;
    for (int i = 1, j = 1; i < n; i++) {
        z[i] = std::max(0ll, std::min(j + z[j] - i, z[i - j]));
        while (i + z[i] < n && s[z[i]] == s[i + z[i]]) {
            z[i]++;
        }
        if (i + z[i] > j + z[j]) {
            j = i;
        }
    }
    return z;
}
```

#### 字符串hash

```
std::mt19937 rng(std::chrono::steady_clock::now().time_since_epoch().count());

bool isprime(int n) {
    if (n <= 1)
        return false;
    for (int i = 2; i * i <= n; i++)
        if (n % i == 0)
            return false;
    return true;
}</pre>
```

```
int findPrime(int n) {
    while (!isprime(n))
       n++;
    return n;
template <int N>
struct StringHash {
    static array<int, N> mod;
   static array<int, N> base;
   vector<array<i64, N>> p, h;
    StringHash() = default;
    StringHash(const string& s) {
        int n = s.size();
        p.resize(n);
       h.resize(n);
        fill(p[0].begin(), p[0].end(), 1);
        fill(h[0].begin(), h[0].end(), 1);
        for (int j = 0; j < N; j++) {
           h[0][j] = s[0] \% mod[j];
            p[0][j] = 1;
        for (int i = 1; i < n; i++)
            for (int j = 0; j < N; j++) {
                p[i][j] = 111 * p[i - 1][j] * base[j] % mod[j];
                h[i][j] = ((111 * h[i - 1][j]) * base[j] + s[i]) % mod[j];
    array<int, N> query(int 1, int r) {
        array<int, N> ans{};
       if (1 > r)
            return array<int, N>{};
        for (int i = 0; i < N; i++) {
            ans[i] =
```

```
(1LL * h[r][i] -
                 1LL * (1 == 0 ? OLL : h[1 - 1][i]) * p[r - 1 + 1][i] % mod[i] +
                 mod[i]) %
                mod[i];
        return ans;
constexpr int HN = 2;
template <int N>
array<int, N> StringHash<N>::mod = []() {
    array<int, N> res;
   for (int i = 0; i < N; i++) {
        res[i] = findPrime(rng() % 900000000 + 100000000);
    return res;
}();
template <int N>
array<int, N> StringHash<N>::base = []() {
    array<int, N> res;
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < 2 * i + 1; j++) {
        res[i] = c;
    return res;
}();
using Hashing = StringHash<HN>;
```