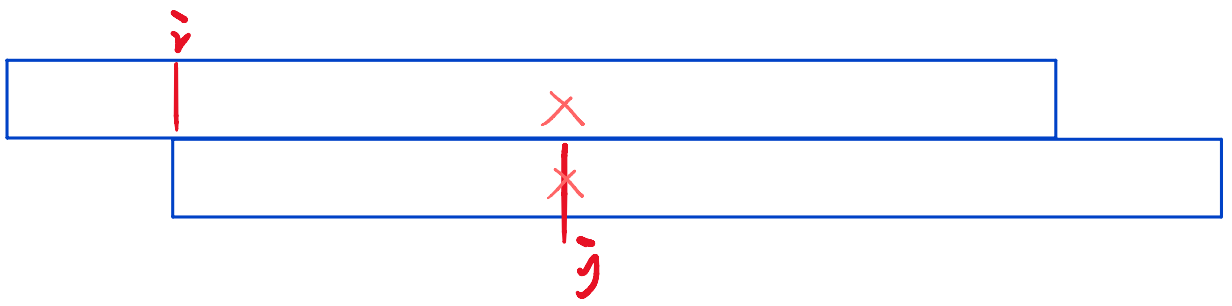
# KMP

2022年6月13日　18:34

暴力做法

```
s[N], p[N];

for (int i = 1; i <= n; i++) {
    bool flag = true;
    for (int j = 1; j <= m; j++) {
        if (s[i] != p[j]) {
            flag = false;
            break;
        }
    }
    if (flag == true) {
        return i;
    }
}
```




遇到不匹配的情况，不能按只让 j 向后移动一位

应该利用之前的信息

在模式串中，预处理以自己一个点为右端点，考虑和这个后缀相等的前缀的最大长度是多少


假设当前下标为 j，则此处的前缀的最后一个位置就是 j

对整个数组都做这个预处理，结果存在 next 数组中

即　　next(j) = j

如何使用 next 数组


在 i 发生不匹配，不用回到 0，开始

而是使用 next，将下一个匹配的字符变为 j+1

减少冗余计算

代码环节：

```cpp
#include <iostream>

using namespace std;

const int N = 1e5 + 10, M = 1e6 + 10;

char s[M];
char p[N];
int ne[N];

int n, m;

int main() {
    cin >> n >> p + 1 >> m >> s + 1;

    for (int i = 2, j = 0; i <= n; i++) {
        while (j && p[i] != p[j + 1]) {
            j = ne[j];
        }
        if (p[i] == p[j + 1]) {
            j++;
        }
        ne[i] = j;
    }

    for (int i = 1, j = 0; i <= m; i++) {
        while (j && s[i] != p[j + 1]) {
            j = ne[j];
        }
        if (s[i] == p[j + 1]) {
            j++;
        }
        if (j == n) {
            cout << i - n << ' ';
            j = ne[j];
        }
    }
}
```