[illegible] - *KMP* [illegible]

[illegible]

[illegible] 408 [illegible] KMP [illegible]KMP [illegible]

KMP [illegible]

[illegible] `next` [illegible]

[illegible] +1[illegible] `next` [illegible]

```cpp
std::optional<size_t> kmpMatch(const std::string& str, const std::string& pattern) {
    if (pattern.empty()) {
        return 0;
    }
    std::vector<size_t> next = generateNext(pattern);
    // pattern[illegible]+1[illegible]
    size_t str_ptr = 0, pattern_ptr = 1;
    while (str_ptr < str.size() && pattern_ptr <= pattern.size()) {
        if (pattern_ptr == 0 || str[str_ptr] == pattern[pattern_ptr - 1]) {
            // [illegible]
            str_ptr++;
            pattern_ptr++;
        } else {
            // [illegible]next[illegible]
            pattern_ptr = next[pattern_ptr - 1];
        }
    }
    if (pattern_ptr > pattern.size()) {
        // [illegible]
        return str_ptr - (pattern_ptr - 1);
    } else {
        // [illegible]
        return {};
    }
}
```

KMP next [illegible]

[illegible]

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

[illegible] MATLAB [illegible] `str[a:b]` [illegible] `str.substr(a, b - a)` [illegible]

[illegible] `next` [illegible]`next[i]` [illegible] `pattern[0:i - 1]` [illegible] +1[illegible] i[illegible] j [illegible] `pattern[0:i]` [illegible] i++ [illegible]j [illegible] `pattern[0:i - 1]` [illegible] +1([illegible] j [illegible])[illegible]

[illegible] i++ [illegible] j [illegible] `pattern[0:i]` [illegible] `pattern[i]` [illegible] `pattern[0:i - 1]` [illegible] `pattern[j - 1]` [illegible]

[illegible] `pattern[0:i]` [illegible] `pattern[0:j - 1]` [illegible] `pattern[i - j:i]`[illegible] `pattern[0:i - 1]` [illegible] `pattern[0:j - 2]` [illegible] `pattern[i - j:i - 1]`[illegible]

[illegible] j[illegible] `pattern[0:j - 2] == pattern[i - j:i - 1]`[illegible] `next[j - 1]` [illegible] `pattern[0:j - 2]` [illegible] +1[illegible] `pattern[i - j:i - 1]` [illegible] +1[illegible] j [illegible] `next[j - 1]`[illegible]

[illegible] j [illegible] `pattern[0]`[illegible] i[illegible]

KMP nextval [illegible]

[illegible] KMP [illegible] `pattern[i]` [illegible] `next[i] - 1` [illegible] `pattern[i] = pattern[next[i] - 1]`[illegible]

`nextval` [illegible] `next` [illegible] `next` [illegible] `pattern[i] = pattern[next[i] - 1]`[illegible] `next[i] = next[next[i] - 1]`[illegible]

[illegible]

```cpp
void generateNextval(const std::string& pattern, std::vector<size_t>& next) {
    if (next.size() == 0) {
        return;
    }
    for (size_t i = 1; i < pattern.size(); ++i) {
        if (pattern[i] == pattern[next[i] - 1]) {
            next[i] = next[next[i] - 1];
        }
    }
    return;
}
```