```cpp
#include <iostream>

#include <vector>

#include <string>


class RabinKarp {

public:

    RabinKarp(const std::string& pattern, int prime = 101)

        : pattern(pattern), prime(prime), d(256), m(pattern.size()), pHash(0), h(1) {

        // Precompute h = pow(d, m-1) % prime

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

            h = (h * d) % prime;

        }


        // Compute the hash value of the pattern

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

            pHash = (d * pHash + pattern[i]) % prime;

        }

    }


    std::vector<int> search(const std::string& text) {

        int n = text.size();

        int tHash = 0;

        std::vector<int> result;


        // Compute the hash value of the first window of the text

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

            tHash = (d * tHash + text[i]) % prime;
        }

        // Slide the pattern over text one by one
        for (int s = 0; s <= n - m; ++s) {
            if (pHash == tHash) {
                // Check the actual characters if hash values match
                if (text.substr(s, m) == pattern) {
                    result.push_back(s);
                }
            }

            // Calculate hash value for the next window of text
            if (s < n - m) {
                tHash = (d * (tHash - text[s] * h) + text[s + m]) % prime;
                if (tHash < 0) {
                    tHash += prime;
                }
            }
        }

        return result;
    }

private:
    std::string pattern;
    int prime;
    int d;
    int m;
    int pHash;
    int h;
};

int main() {
    std::string text = "ABCCDDAEFGABCD";
    std::string pattern = "ABCD";
    RabinKarp rk(pattern);

    std::vector<int> result = rk.search(text);
    for (int index : result) {
        std::cout << "Pattern found at index: " << index << std::endl;
    }

    return 0;
}
```

DRy Run

Text = "ABAB ABC"
Pattern = "ABAD"

Step	Window	Hash Calculation	tHash	pHash	Match?
1	'ABAB"	Initial tHash Calculation	13	13	Yes, at index 0
2	"BABA"	(256*(13-'A'*5) +(A')%101	29	13	NO
3	"ABAB"	(256)*(29-'B'*5)+'B')%101	13	13	Yes, at Index 2
4	"BABC"	(256*(~~bb~~ 13-'A'*5)+C)%101	30	13	NO

The matchs at indices 0 & 2 are correctly identified.