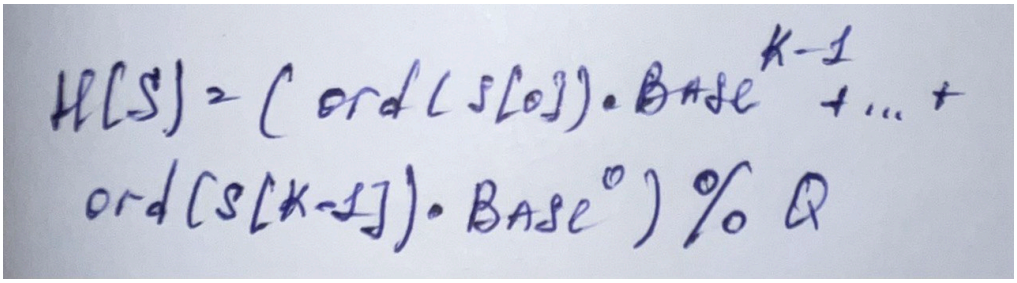
Строку рассматривают как число в позиционной системе счисления с основанием Base (в моем случае простое число = 37)

Для строки длины K: s_0…s_K-1 хэш выглядит так:



Где Q - большое простое число (у меня 10^9 + 7). Используется, чтобы ограничить значение хэша

Чтобы обновить состояние скользящего окна, нужно вычесть из хэша первый символ в окне и добавить последний. То есть:

1) Hash_new = Hash_old - ord(s[i]) * Base^(k-1) - **Удаление текущего первого символа в окне**

2) Hash_new = Hash_new * Base - **«Сдвиг» всех символов влево, т.к. 1й удалили**

3) Hash_new +=ord(s[i + m]) * Base^0 - **Добавление нового символа**

4) Hash_new %= Q

5) Если Hash_new < 0: Hash_new += Q

Итоговая сложность алгоритма поиска вхождения pattern в строку: O(N + M), т.к. за N проходим по строке, и O(M) времени потребуется на посимвольное сравнение, если хэши равны, но совпадения обычно происходят нечасто. Обновление хэша - O(1)

В худшем случае O(N * M), если хэши совпадают на каждом шаге, например:

S = «aaaaaaa», pattern = «aa»,

То тогда на каждой итерации в цикле придется проверять равенство посимвольно. Но такой случай на практике встречается редко.