

**Скользящий хеш**



Материал из Википедии — свободной энциклопедии

**Скользящий хеш** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA)  *rolling hash*, также  *кольцевой хеш*) — [хеш-функция](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F), обрабатывающая вход в рамках некоторого окна. Получение значения хеш-функции для сдвинутого окна в таких функциях является дешевой операцией. Для пересчета значения требуется знать лишь предыдущее значение хеша, значение входных данных, которые остались за пределами окна, и значение данных, которые попали в окно. Другими словами, если представляет собой хеш последовательности , то хеш



для «сдвинутой» последовательности может быть получен с помощью легко вычислимой функции .



Возможность быстрого «сдвига» хеша накладывает некоторые ограничения на теоретические [гарантии. В частности, показано[1], что семейства кольцевых хешей не могут быть 3-](https://ru.wikipedia.org/w/index.php?title=K-%D0%BD%D0%B5%D0%B7%D0%B0%D0%B2%D0%B8%D1%81%D0%B8%D0%BC%D0%BE%D0%B5_%D1%85%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1)

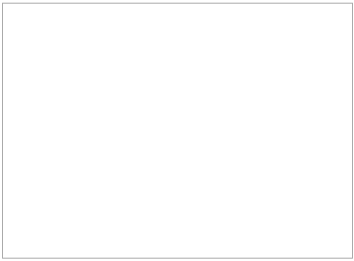
[независимыми; максимум —](https://ru.wikipedia.org/w/index.php?title=K-%D0%BD%D0%B5%D0%B7%D0%B0%D0%B2%D0%B8%D1%81%D0%B8%D0%BC%D0%BE%D0%B5_%D1%85%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) [универсальными](https://ru.wikipedia.org/wiki/%D0%A3%D0%BD%D0%B8%D0%B2%D0%B5%D1%80%D1%81%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%85%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) [или 2-независимыми. Впрочем, для](https://ru.wikipedia.org/w/index.php?title=K-%D0%BD%D0%B5%D0%B7%D0%B0%D0%B2%D0%B8%D1%81%D0%B8%D0%BC%D0%BE%D0%B5_%D1%85%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1) большинства приложений достаточно универсальности (даже приблизительной).

Кольцевой хеш применяется для поиска подстроки в алгоритме [Рабина — Карпа](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A0%D0%B0%D0%B1%D0%B8%D0%BD%D0%B0_%E2%80%94_%D0%9A%D0%B0%D1%80%D0%BF%D0%B0), для вычисления хешей [N-грамм](https://ru.wikipedia.org/wiki/N-%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0) в тексте[[2]](#_page4_x53.75_y75.50), а также в программе [rsync](https://ru.wikipedia.org/wiki/Rsync) для сравнения двоичных

файлов (используется кольцевая версия [adler-32](https://ru.wikipedia.org/wiki/Adler-32)).



**Содержание**



[**Полиномиальный хеш**](#_page0_x36.50_y623.75)

[**Полиномиальный хеш над полем GF(2L)**](#_page1_x36.50_y467.00) [**Хеш циклическими полиномами (Buzhash)**](#_page2_x36.50_y143.00) [**Хеш Рабина**](#_page2_x36.50_y597.50)

[**Ссылки**](#_page3_x36.50_y645.50)

[**Примечания**](#_page4_x36.50_y30.50)

[**Литература**](#_page4_x36.50_y223.25)

**Полиномиальный хеш**



В [алгоритме Рабина — Карпа](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A0%D0%B0%D0%B1%D0%B8%D0%BD%D0%B0_%E2%80%94_%D0%9A%D0%B0%D1%80%D0%BF%D0%B0) часто используется простой полиномиальный кольцевой хеш, построенный на операциях умножения и сложения[[3]](#_page4_x53.75_y92.00)[[4]](#_page4_x53.75_y108.50):

.



Чтобы избежать использования целочисленной арифметики произвольной точности, используется арифметика в [кольце вычетов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D1%8C%D1%86%D0%BE_%D0%B2%D1%8B%D1%87%D0%B5%D1%82%D0%BE%D0%B2) по модулю , умещающемуся в одно машинное слово. Выбор констант и очень важен для получения качественного хеша. В исходном



варианте хеша предполагалось, что должно быть случайно выбранным простым числом, а .[[3]](#_page4_x53.75_y92.00) Но ввиду того, что алгоритм выбора случайного простого числа не такой простой,



предпочитают использовать вариант хеша, в котором является фиксированным простым числом, а выбирается случайно из диапазона . Дитзфелбингер и др.[[4]](#_page4_x53.75_y108.50)



показали, что такой вариант хеша имеет те же теоретические характеристики, что и исходный. В частности, вероятность совпадения значений хешей двух различных строк и не превосходит , если и представляют собой целые числа



из диапазона , и выбирается действительно случайно.



Удаление старых входных символов и добавление новых производится путём прибавления или вычитания первого или последнего члена формулы (по модулю ). Для удаления члена



хранят заранее посчитанное значение . Сдвиг окна производится путём домножения всего многочлена на либо делением на (если простое, то в кольце вычетов возможно вместо деления производить умножение на обратную величину). На



практике удобнее всего полагать или для, соответственно, 32- и 64- битовых машинных слов (это так называемые [простые числа Мерсенна](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%BE_%D0%9C%D0%B5%D1%80%D1%81%D0%B5%D0%BD%D0%BD%D0%B0)). В таком случае операция взятия модуля может быть выполнена на многих компьютерах с помощью быстрых



операций побитового сдвига и сложения[[5]](#_page4_x53.75_y125.00). Другой возможный выбор — значения или , для которых тоже существуют быстрые алгоритмы взятия остатка от деления



на (при этом диапазон допустимых значений немного сужают)[[6]](#_page4_x53.75_y141.50). Частое заблуждение —



полагать . Существуют семейства строк, на которых хеш с будет всегда давать множество [коллизий](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BB%D0%BB%D0%B8%D0%B7%D0%B8%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8), независимо от выбора .[[7]](#_page4_x53.75_y158.00) Эти и другие дальнейшие детали реализации



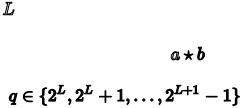
[и теоретического анализ полиномиального хеша можно найти в статье об алгоритме Рабина — Карпа.](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A0%D0%B0%D0%B1%D0%B8%D0%BD%D0%B0_%E2%80%94_%D0%9A%D0%B0%D1%80%D0%BF%D0%B0#%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D1%83%D0%B5%D0%BC%D0%B0%D1%8F_%D1%85%D0%B5%D1%88-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F)



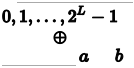
**Полиномиальный хеш над полем GF(2****L)**



Данный хеш похож на обычный полиномиальный хеш, но все вычисления в нём производятся в [конечном поле](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%B5%D1%87%D0%BD%D0%BE%D0%B5_%D0%BF%D0%BE%D0%BB%D0%B5) . Обычно выбирается равным 64. Элементы поля — это числа

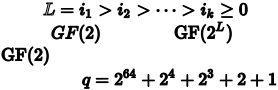


* [Сложение в поле реализуется с помощью операции побитового исключающего](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BF%D0%BE_%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8E_2)



[«или» , а умножение выполняется с помощью операции , которая сначала беспереносно умножает на , а потом берёт остаток от «беспереносного» деления результата на некоторый](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%B5%D1%81%D0%BF%D0%B5%D1%80%D0%B5%D0%BD%D0%BE%D1%81%D0%BD%D0%BE%D0%B5_%D1%83%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5&action=edit&redlink=1)

выбранный фиксированный элемент (беспереносным делением здесь названа операция обратная беспереносному умножению). Элемент должен быть выбран так, что и



— это [неприводимый многочлен](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D1%80%D0%B8%D0%B2%D0%BE%D0%B4%D0%B8%D0%BC%D1%8B%D0%B9_%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%D1%87%D0%BB%D0%B5%D0%BD) над полем (на поле часто смотрят как на множество многочленов над полем по модулю произвольного

неприводимого многочлена степени ). Например, можно положить [[8]](#_page4_x53.75_y174.50). Тогда хеш вычисляется следующим образом[[4]](#_page4_x53.75_y108.50):



,



где — это случайно выбранное на этапе инициализации хеша число из диапазона



, а — это короткая запись для , где повторён раз. С помощью [основной теоремы алгебры](https://ru.wikipedia.org/wiki/%D0%9E%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D0%B0%D1%8F_%D1%82%D0%B5%D0%BE%D1%80%D0%B5%D0%BC%D0%B0_%D0%B0%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D1%8B) можно показать, что вероятность коллизии хешей двух различных строк длины не превосходит . Показано[[8]](#_page4_x53.75_y174.50), что на современных процессорах [Intel](https://ru.wikipedia.org/wiki/Intel) и [AMD](https://ru.wikipedia.org/wiki/AMD) вся необходимая для хеша арифметика над полем может быть эффективно вычислена с помощью инструкций из расширения [CLMUL](https://ru.wikipedia.org/w/index.php?title=%D0%9D%D0%B0%D0%B1%D0%BE%D1%80_%D0%B8%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D0%B9_CLMUL&action=edit&redlink=1).



**Хеш циклическими полиномами (Buzhash)**



Пусть — какой-то хеш, который отображает символы хешируемой строки в - битовые числа (обычно или ). Хеш циклическими полиномами определяется следующим образом[[2]](#_page4_x53.75_y75.50):



[где — это операция побитового исключающего «или», а — это операция циклического сдвига -битового числа на битов влево. Несложно показать, что данный хеш кольцевой:](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%81%D0%B4%D0%B2%D0%B8%D0%B3#%D0%A6%D0%B8%D0%BA%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%81%D0%B4%D0%B2%D0%B8%D0%B3)



Главное преимущество этого хеша в том, что он использует только быстрые побитовые операции доступные на многих современных компьютерах. Качество хеша напрямую зависит

от выбора функции . Лемире и Касер[[1]](#_page4_x53.75_y59.00) доказали, что если функция выбирается случайно из семейства [независимых хеш-функций](https://ru.wikipedia.org/w/index.php?title=K-%D0%BD%D0%B5%D0%B7%D0%B0%D0%B2%D0%B8%D1%81%D0%B8%D0%BC%D0%BE%D0%B5_%D1%85%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5&action=edit&redlink=1), то вероятность совпадения хешей двух различных



строк длины не превосходит . Это накладывает определённые ограничения на диапазон задач, в которых данный хеш может использоваться. Во-первых, длина хешируемых строк должна быть меньше . Для алгоритмов хеширования общего назначения это условие может быть проблемой, но, например, для хеширования  [-грамм](https://ru.wikipedia.org/wiki/N-%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), где обычно не превосходит 16, такое ограничение является естественным (в случае -грамм роль символов играют



отдельные [лексемы](https://ru.wikipedia.org/wiki/%D0%9B%D0%B5%D0%BA%D1%81%D0%B5%D0%BC%D0%B0_(%D0%BB%D0%B8%D0%BD%D0%B3%D0%B2%D0%B8%D1%81%D1%82%D0%B8%D0%BA%D0%B0)) текста). Во-вторых, выбор семейства *независимых* функций в некоторых случаях тоже может быть проблемой. Для байтового алфавита свойством независимости



обладает семейство функций , закодированных таблицей из 256-и различных случайных - битовых чисел (выбор функции — это заполнение таблицы). Для хеширования -грамм можно присваивать различные случайные -битовые числа различным лексемам (обычно число



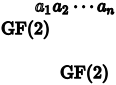
разных лексем в таких задачах относительно невелико) и такое семейство хеш-функций тоже имеет свойство независимости.



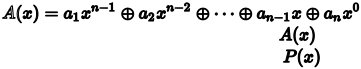
**Хеш Рабина**



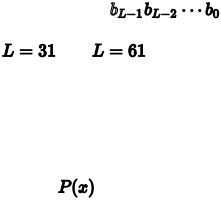
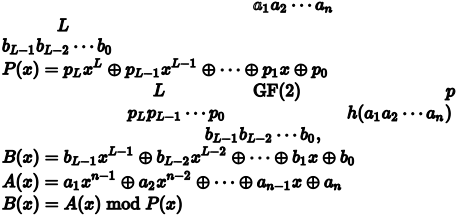
Данный хеш применим только в специальном случае, когда символы хешируемой строки суть числа 0 и 1. Идея хеша в том, чтобы смотреть на входную строку как



на многочлен над полем , а сам хеш представляет собой взятие остатка от деления на случайно выбранный на этапе инициализации хеша неприводимый многочлен степени над полем . По существу это та же процедура, что используется в [CRC](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D0%BA%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%B7%D0%B1%D1%8B%D1%82%D0%BE%D1%87%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4). Рассмотрим её более подробно.



Результат хеширования строки — это последовательность битов . Число выбирается простым[[9]](#_page4_x53.75_y191.00) и достаточно большим, но так чтобы последовательность умещалась в одно машинное слово (обычно берут или [[9]](#_page4_x53.75_y191.00)). Пусть

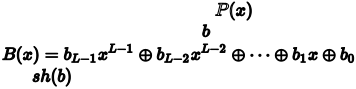


[представляет собой некоторый неприводимый многочлен степени над полем . Обозначим через соответствующее число с битовым](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D1%80%D0%B8%D0%B2%D0%BE%D0%B4%D0%B8%D0%BC%D1%8B%D0%B9_%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE%D1%87%D0%BB%D0%B5%D0%BD) представлением . Хеш-функция определяется как число с битовым представлением таким что многочлен является остатком от деления многочлена

на многочлен , то есть

.

Несмотря на весьма запутанное определение, хеш Рабина довольно просто реализуем (если неприводимый многочлен уже найден). Вычисления опираются на такое несложное



наблюдение: если число с битовым представлением кодирует многочлен , то число кодирует многочлен ,



где обозначает операцию [побитового сдвига](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%81%D0%B4%D0%B2%D0%B8%D0%B3) числа на один бит влево с замещением младшего бита нулём (не путать с циклическим сдвигом , определённым выше!). Пусть



, и — это битовое представление . Тогда вычисляется следующим образом:



если



если

Хеш является кольцевым. Пусть и — это битовое представление . Хеш вычисляется следующим образом[[9]](#_page4_x53.75_y191.00):



если



если

где — это -битовое число, битовое представление которого соответствует многочлену



* Число вычисляют заранее при инициализации хеша строки длины .



Главная сложность — случайным образом выбрать неприводимый многочлен степени . Рабин[[9]](#_page4_x53.75_y191.00) описал эффективный алгоритм, позволяющий это сделать, и доказал, что вероятность



коллизии хешей двух различных строк длины при случайном выборе не превосходит



.



Отметим, что данный хеш часто путают с полиномиальным хешем из-за схожей области применения, рассмотрения многочленов и общего автора.

**Ссылки**



* [ngramhashing (https://code.google.com/p/ngramhashing/)](https://code.google.com/p/ngramhashing/) — [свободная](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) C++-реализация нескольких кольцевых хеш-функций
* [rollinghashjava (https://code.google.com/p/rollinghashjava/)](https://code.google.com/p/rollinghashjava/) — Java-реализация кольцевых хеш-функций под [лицензией Apache](https://ru.wikipedia.org/wiki/%D0%9B%D0%B8%D1%86%D0%B5%D0%BD%D0%B7%D0%B8%D1%8F_Apache)

**Примечания**



1. [Lemire, Kaser, 2010](#_page4_x55.25_y413.75).
2. [Cohen, 1997](#_page4_x55.25_y256.25).
3. [Rabin, Karp, 1987](#_page4_x55.25_y557.75).
4. [Dietzfelbinger, Gil, Matias, Pippinger, 1992](#_page4_x55.25_y299.75).
5. S. E. Anderson. [Bit twiddling hacks. (https://graphics.stanford.edu/~seander/bithacks.html)](https://graphics.stanford.edu/~seander/bithacks.html)
6. [Krovetz, Rogaway, 2000](#_page4_x55.25_y356.75).
7. [Pachocki, Radoszewski, 2013](#_page4_x55.25_y601.25).
8. [Lemire, Kaser, 2016](#_page4_x55.25_y457.25).
9. [Rabin, 1981](#_page4_x55.25_y514.25).

**Литература**



* [*Cohen J. D.* Recursive hashing functions for n-grams // ACM Transactions on Information Systems. — New York, USA: ACM, 1997. — Т. 15, № 3. — С. 291–320. —](https://ru.wikipedia.org/w/index.php?title=ACM_Transactions_on_Information_Systems&action=edit&redlink=1) [doi](https://ru.wikipedia.org/wiki/Doi):[10.1145/256163.256168 (https://dx.doi.org/10.1145%2F256163.256168)](https://dx.doi.org/10.1145%2F256163.256168).
  + *Dietzfelbinger* *M., Gil J., Matias Y.,* [*Pippenger N.*](https://ru.wikipedia.org/w/index.php?title=%D0%9F%D0%B8%D0%BF%D0%BF%D0%B5%D0%BD%D0%B4%D0%B6%D0%B5%D1%80,_%D0%9D%D0%B8%D0%BA%D0%BE%D0%BB%D0%B0%D1%81_%D0%94%D0%B6%D0%BE%D0%BD&action=edit&redlink=1) Polynomial hash functions are reliable // Proceedings of the 19th [International Colloquium on Automata, Languages and Programming](https://ru.wikipedia.org/w/index.php?title=International_Colloquium_on_Automata,_Languages_and_Programming&action=edit&redlink=1) [(ICALP'92). — Berlin, Germany: Springer-Verlag, 1992. — С. 235–246. — doi:10.1007/3-540- 55719-9\_77 (https://dx.doi.org/10.1007%2F3-540-55719-9\_77).](https://dx.doi.org/10.1007%2F3-540-55719-9_77)
  + *Krovetz* *T., Rogaway P.* Fast universal hashing with small keys and no preprocessing: the PolyR construction // Proceedings of the International Conference on Information Security and [Cryptology. — Berlin, Germany: Springer-Verlag, 2000. — С. 73–89. — doi:10.1007/3-540-45247- 8\_7 (https://dx.doi.org/10.1007%2F3-540-45247-8\_7).](https://dx.doi.org/10.1007%2F3-540-45247-8_7)
  + *Lemire* *D., Kaser O.* Recursive n-gram hashing is pairwise independent, at best // Journal Computer Speech and Language. — London, UK: Academic Press Ltd., 2010. — Т. 24, № 4. — С. 698–710. — [doi](https://ru.wikipedia.org/wiki/Doi):[10.1016/j.csl.2009.12.001 (https://dx.doi.org/10.1016%2Fj.csl.2009.12.001)](https://dx.doi.org/10.1016%2Fj.csl.2009.12.001).
* *Lemire* *D., Kaser O.* Faster 64-bit universal hashing using carry-less multiplications // Journal of Cryptographic Engineering. — Berlin, Germany: Springer-Verlag, 2016. — Т. 6, № 3. — [С. 171–185. — doi:10.1007/s13389-015-0110-5 (https://dx.doi.org/10.1007%2Fs13389-015-0110- 5).](https://dx.doi.org/10.1007%2Fs13389-015-0110-5)
* [*Рабин М. О.*](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B1%D0%B8%D0%BD,_%D0%9C%D0%B8%D1%85%D0%B0%D1%8D%D0%BB%D1%8C_%D0%9E%D1%88%D0%B5%D1%80) Fingerprinting by random polynomials // Tech Report TR-CSE-03-01. — Center for [Research in Computing Technology, Harvard University, 1981. — С. 1–14. Архивировано (http://ww w.xmailserver.org/rabin.pdf) 29 апреля 2018 года.](http://www.xmailserver.org/rabin.pdf)



* + [*Рабин М. О., Карп Р. М.* Efficient randomized pattern-matching algorithms // IBM Journal of Research and Development. — IBM, 1987. — Т. 31, № 2. — С. 249–260. —](https://ru.wikipedia.org/w/index.php?title=IBM_Journal_of_Research_and_Development&action=edit&redlink=1) [doi](https://ru.wikipedia.org/wiki/Doi):[10.1147/rd.312.0249 (https://dx.doi.org/10.1147%2Frd.312.0249)](https://dx.doi.org/10.1147%2Frd.312.0249).
* [*Pachocki J., Radoszewski J.* Where to use and how not to use polynomial string hashing (https:// www.mii.lt/olympiads\_in\_informatics/pdf/INFOL119.pdf) // Olympiads in Informatics. — Vilnus,](https://www.mii.lt/olympiads_in_informatics/pdf/INFOL119.pdf) Lithuania: Vilnus University, 2013. — Т. 7. — С. 90–100.



Источник — [https://ru.wikipedia.org/w/index.php?title=Скользящий\_хеш&oldid=110767329](https://ru.wikipedia.org/w/index.php?title=%D0%A1%D0%BA%D0%BE%D0%BB%D1%8C%D0%B7%D1%8F%D1%89%D0%B8%D0%B9_%D1%85%D0%B5%D1%88&oldid=110767329) **Эта страница в последний раз была отредактирована 29 ноября 2020 в 11:54.**



Текст доступен по [лицензии Creative Commons Attribution-ShareAlike](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%BA%D0%B8%D0%BF%D0%B5%D0%B4%D0%B8%D1%8F:%D0%A2%D0%B5%D0%BA%D1%81%D1%82_%D0%BB%D0%B8%D1%86%D0%B5%D0%BD%D0%B7%D0%B8%D0%B8_Creative_Commons_Attribution-ShareAlike_3.0_Unported); в отдельных случаях могут действовать дополнительные условия.

Wikipedia® — зарегистрированный товарный знак некоммерческой организации [Wikimedia Foundation, Inc.](https://wikimediafoundation.org/)