Hash Functions and Types of Hash functions - GeeksforGeeks

Source: https://www.geeksforgeeks.org/hash-functions-and-list-types-of-hash-functions/

Courses Tutorials Practice Jobs DSA Tutorial Interview Questions Quizzes Must Do Advanced DSA System Design Aptitude Puzzles Interview Corner DSA Python Technical Scripter 2026 Explore DSA Fundamentals Logic Building Problems Analysis of Algorithms Data Structures Array Data Structure String in Data Structure Hashing in Data Structure Linked List Data Structure Stack Data Structure Queue Data Structure Tree Data Structure Graph Data Structure Trie Data Structure Algorithms Searching Algorithms Sorting Algorithms Introduction to Recursion Greedy Algorithms Tutorial Graph Algorithms Dynamic Programming or DP Bitwise Algorithms Advanced Segment Tree Binary Indexed Tree or Fenwick Tree Square Root (Sqrt) Decomposition Algorithm Binary Lifting Geometry Interview Preparation Interview Corner GfG160 Practice Problem GeeksforGeeks Practice - Leading Online Coding Platform Problem of The Day - Develop the Habit of Coding DSA Course 90% Refund Hash Functions and Types of Hash functions Last Updated : 23 Jul, 2025 Hash functions are a fundamental concept in computer science and play a crucial role in various applications such as data storage, retrieval, and cryptography. A hash function creates a mapping from an input key to an index in hash table. Below are few examples. Phone numbers as input keys : Consider a hash table of size 100. A simple example hash function is to consider the last two digits of phone numbers so that we have valid hash table indexes as output. This is mainly taking remainder when input phone number is divided by 100. Please note that taking first two digits of a phone number would not be a good idea for a hash function as there would be many phone number having same first two digits. Lowercase English Strings as Keys : Consider a hash table of size 100. A simple way to hash the strings would be add their codes (1 for a, 2 for b, ... 26 for z) and take remainder of the sum when divided by 100. This hash function may not be a good idea as strings "ad" and "bc" would have the same hash value. A better idea would be to do weighted sum of characters and then find remainder. Please refer an example string hashing function for details. Key Properties of Hash Functions Deterministic : A hash function must consistently produce the same output for the same input. Fixed Output Size : The output of a hash function should have a fixed size, regardless of the size of the input. Efficiency : The hash function should be able to process input quickly. Uniformity : The hash function should distribute the hash values uniformly across the output space to avoid clustering. Pre-image Resistance : It should be computationally infeasible to reverse the hash function, i.e., to find the original input given a hash value. Collision Resistance : It should be difficult to find two different inputs that produce the same hash value. Avalanche Effect : A small change in the input should produce a significantly different hash value. Applications of Hash Functions Hash Tables : The most common use of hash functions in DSA is in hash tables, which provide an efficient way to store and retrieve data. Data Integrity : Hash functions are used to ensure the integrity of data by generating checksums. Cryptography : In cryptographic applications, hash functions are used to create secure hash algorithms like SHA-256. Data Structures : Hash functions are utilized in various data structures such as Bloom filters and hash sets. Types of Hash Functions There are many hash functions that use numeric or alphanumeric keys. This article focuses on discussing different hash functions: Division Method. Multiplication Method Mid-Square Method Folding Method Cryptographic Hash Functions Universal Hashing Perfect Hashing Let's begin discussing these methods in detail. 1. Division Method The division method involves dividing the key by a prime number and using the remainder as the hash value. h (k)= k mod m Where k is the key and ? m is a prime number. Advantages : Simple to implement. Works well when ? m is a prime number. Disadvantages : Poor distribution if ? m is not chosen wisely. 2. Multiplication Method In the multiplication method, a constant ? A (0 < A < 1) is used to multiply the key. The fractional part of the product is then multiplied by ? m to get the hash value. h (k)=■ m (kA mod1)■ Where ■ ■ denotes the floor function. Advantages : Less sensitive to the choice of ? m . Disadvantages : More complex than the division method. 3. Mid-Square Method In the mid-square method, the key is squared, and the middle digits of the result are taken as the hash value. Steps : Square the key. Extract the middle digits of the squared value. Advantages : Produces a good distribution of hash values. Disadvantages : May require more computational effort. 4. Folding Method The folding method involves dividing the key into equal parts, summing the parts, and then taking the modulo with respect to ? m . Steps : Divide the key into parts. Sum the parts. Take the modulo ? m of the sum. Advantages : Simple and easy to implement. Disadvantages : Depends on the choice of partitioning scheme. 5. Cryptographic Hash Functions Cryptographic hash functions are designed to be secure and are used in cryptography. Examples include MD5, SHA-1, and SHA-256. Characteristics : Pre-image resistance. Second pre-image resistance. Collision resistance. Advantages : High security. Disadvantages : Computationally intensive. 6. Universal Hashing Universal hashing uses a family of hash functions to minimize the chance of collision for any given set of inputs. $h(k)=((a \cdot k+b) \bmod p) \bmod m$ Where a and b are randomly chosen constants, p is a prime number greater than m , and k is the key. Advantages : Reduces the probability of collisions. Disadvantages : Requires more computation and storage. 7. Perfect Hashing Perfect hashing aims to create a collision-free hash function for a static set of keys. It guarantees that no two keys will hash to the same value. Types : Minimal Perfect Hashing: Ensures that the range of the hash function is equal to the number of keys. Non-minimal Perfect Hashing: The range may be larger than the number of keys. Advantages : No collisions. Disadvantages : Complex to construct. Conclusion In conclusion, hash functions are very important tools that help store and find data quickly. Knowing the different types of hash functions and how to use them correctly is key to making software work better and more securely. By choosing the right hash function for the job, developers can greatly improve the efficiency and reliability of their systems. Comment Article Tags: Article Tags: DSA Data Structures-Hash