String Manipulation and Algorithms

Cracking Coding Interview @ Ostad

- Partharaj Deb -

Introduction to Strings

- Definition: A sequence of characters.
  *“Hello”, “World”, “Hello World”, “1234”, “$%&*_”, “<space>“*
- Importance in coding interviews and real-world applications.

String Operations

- Basic operations: concatenation, substring, length.
- Common manipulations: reversing, changing case, trimming.

String Manipulation Techniques

- Exploring different kinds of string manipulations:
  - Swapping characters
  - Replacing characters
  - Removing duplicates
  - Palindrome detection
  - Anagram detection
  - Hashing
  - Encoding/Decoding

Hashing

Hashing is a process that converts an input (or 'message') into a fixed-size string of bytes. The output is typically a hash code or hash value.

- **Hash Function**: A mathematical algorithm that maps data of arbitrary size to a hash of a fixed size.
- **Purpose**: Ensures data integrity, speeds up data retrieval, and secures data via cryptographic hash functions.

Example

Input: "Hello, World!"
MD5 Hash: fc3ff98e8c6a0d3087d515c0473f8677
SHA-1 Hash: 2ef7bde608ce5404e97d5f042f95f89f1c232871
SHA-256 Hash: a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e

Applications of Hashing

- **Data Structures**: Used in hash tables, hash maps, and dictionaries for efficient data retrieval.
- **Data Integrity**: Ensures that data has not been altered by using checksums and digital signatures.
- **Cryptography**: Secures data by creating digital fingerprints for data integrity and authentication.
- **Password Storage**: Hashes passwords to store them securely in databases.

Sliding Window Technique

A method used to efficiently solve problems involving arrays or lists by maintaining a window that slides over the data.

Purpose: Reduces the complexity of problems that involve examining contiguous subarrays, thus improving performance and reducing unnecessary computations.

Rabin-Karp Algorithm

The Rabin-Karp algorithm is a string-searching algorithm that uses hashing to find a pattern in a text efficiently.

- **Purpose**: Designed to search for multiple patterns simultaneously, making it effective for plagiarism detection and DNA sequencing.

Key Concepts

- **Hash Function**: Converts a string into a numerical value (hash) to quickly compare substrings.
- **Rolling Hash**: A hash function that updates the hash value as the window slides over the text, allowing efficient recalculations.

How Rabin-Karp Algorithm works

Preprocessing:

- Compute the hash value of the pattern P.
- Compute the hash value for the first window of the text T.

Sliding Window:

- Slide the window one position at a time over the text.
- Update the hash value of the current window using the rolling hash technique.
- Compare the hash value of the current window with the pattern's hash value.

Verification:

- If the hash values match, compare the actual substring with the pattern to confirm a match (due to potential hash collisions).

Rolling Hash Formula

For a pattern of length m:

- Initial Hash: H(P) = P[0]*d^(m-1) + P[1]*d^(m-2) + ... + P[m-1]*d^0 % q
- Sliding Hash: H(T[i+1:i+m+1]) = (d * (H(T[i:i+m]) - T[i]*d^(m-1)) + T[i+m]) % q

Where:

- d is the number of possible characters (e.g., 256 for ASCII).
- q is a large prime number to reduce collisions.

Regular Expression

A regular expression (RegEx) is a powerful tool for text pattern matching and manipulation. It is a sequence of characters that defines a search pattern.

Common Use Cases:

- String validation (email addresses, phone numbers, etc.).
- Text searching and extraction.
- Find and replace operations in text editors and IDEs.

Regular Expression Syntax

- Characters and Metacharacters:
  - Regular expressions consist of literal characters and metacharacters.
  - Metacharacters include ., *, +, ?, [], etc.
  - Example: a*b matches "ab," "aab," "aaab," and so on.
- Quantifiers:
  - Specify the number of occurrences of a character or group.
  - * (zero or more), + (one or more), ? (zero or one).
  - Example: \d{3} matches exactly three digits.
- Character Classes and Escape Characters:
  - [ ] defines a character class.
  - \ escapes a metacharacter.
  - Example: [aeiou] matches any vowel, \d matches any digit.

Regular Expression Syntax (Cont.)

- Anchors:
  - ^ (caret) matches the start of a string.
  - $ (dollar sign) matches the end of a string.
  - Example: ^\d{3}$ matches exactly three digits from the start to end.
- Groups and Capturing:
  - () defines a group.
  - Capturing groups for extracting matched portions.
  - Example: (\d{3})-(\d{4}) captures area code and phone number parts.
- Lookahead and Lookbehind:
  - Positive and negative lookahead ((?=...) and (?!...)).
  - Positive and negative lookbehind ((?<=...) and (?<!...)).
  - Example: (?<=@)\w+ matches the domain part of an email address.

Regular Expression Syntax (Cont.)

- Modifiers:
  - Case insensitivity (i), global matching (g), multiline matching (m).
  - Example: /pattern/i matches "Pattern," "pattern," etc.
- Escape Sequences:
  - Represent special characters using escape sequences.
  - Example: \n for newline, \t for tab.
- Wildcard and Alternation:
  - . (dot) matches any character except newline.
  - | (pipe) denotes alternation between choices.
  - Example: (cat|dog) matches either "cat" or "dog."

Practice here- https://regex101.com

Best Practices

Tips for optimizing string algorithms:

- Choosing the right data structures.
- Efficient string comparison techniques.

Q&A...