Nama : Restu Wibisono
NPM : 2340506061

Implementasi Hashing dalam Python

Hashing adalah teknik yang digunakan untuk memetakan data dari rentang nilai yang besar menjadi rentang nilai yang lebih kecil melalui fungsi hash. Dalam Python, implementasi hashing dapat dilakukan menggunakan beberapa struktur data bawaan seperti dictionary (kamus) dan set. Berikut adalah rangkuman tentang implementasi hashing dalam Python:

1. **Penggunaan Dictionary (Kamus)**

   Dictionary adalah struktur data yang umum digunakan dalam Python untuk menyimpan pasangan kunci-nilai yang bersifat tidak terurut. Dictionary menggunakan teknik hashing untuk memetakan kunci ke nilai-nilai yang sesuai. Dalam implementasi dictionary, kunci dipetakan ke dalam tabel hash menggunakan fungsi hash. Fungsi hash ini menghasilkan indeks di mana nilai-nilai akan disimpan. Setiap entri dalam tabel hash menunjuk ke daftar yang berisi semua nilai dengan hash yang sama. Saat dicari, kunci yang diberikan akan dihash dan indeks yang dihasilkan akan digunakan untuk mengakses nilai yang sesuai. Jika terdapat kollision hash (dua kunci menghasilkan indeks yang sama), dictionary akan menangani kasus ini dengan teknik chaining atau probing.

2. **Keuntungan dan Kekurangan**

   Keuntungan penggunaan hashing dalam implementasi dictionary adalah kemampuan untuk melakukan operasi penambahan, penghapusan, dan pencarian dengan kompleksitas waktu yang konstan, yaitu O(1) dalam kasus terbaik dan rata-rata. Namun, dalam kasus terburuk, kompleksitas waktu dapat menjadi O(n), terutama jika terjadi banyak kollision hash. Selain itu, dictionary tidak menyimpan urutan asli dari kunci-nilai yang dimasukkan. Ini berarti tidak ada jaminan bahwa urutan saat mengakses elemen akan tetap sama seperti saat elemen dimasukkan.

3. **Implementasi Set**

   Set adalah struktur data yang digunakan untuk menyimpan kumpulan elemen unik, di mana setiap elemen hanya muncul sekali. Dalam Python, set menggunakan teknik hashing untuk menyimpan elemen-elemennya. Implementasi set mirip dengan dictionary, di mana setiap elemen akan dihash dan ditempatkan pada indeks yang sesuai dalam tabel hash. Namun, dalam set, hanya ada satu nilai (tanpa kunci) yang disimpan dalam setiap

entri tabel hash. Jika terjadi kollision hash, teknik chaining atau probing juga digunakan untuk menangani kasus tersebut.

4. Istilah-istilah dalam Hashing

Dalam konteks hashing dalam Python, terdapat beberapa istilah yang penting untuk dipahami. Berikut adalah beberapa istilah yang umum digunakan dalam hashing dalam Python:

1) Hash Function (Fungsi Hash)

Hash function adalah fungsi yang digunakan untuk mengubah data menjadi nilai hash yang unik. Fungsi ini mengambil input data dan mengembalikan nilai hash yang merupakan representasi numerik dari data tersebut. Hash function harus menghasilkan nilai hash yang seragam untuk input yang berbeda-beda, dan sebaiknya memiliki kompleksitas waktu yang rendah.

2) Hash Value (Nilai Hash)

Hash value adalah hasil dari aplikasi fungsi hash pada data tertentu. Nilai hash biasanya berupa bilangan bulat atau string yang merepresentasikan data secara unik. Nilai hash digunakan untuk mengindeks dan menyimpan data dalam struktur data hashing seperti dictionary atau set.

3) Collision (Kolisi Hash)

Collision terjadi ketika dua atau lebih data menghasilkan nilai hash yang sama saat diaplikasikan pada fungsi hash. Ini adalah masalah umum dalam hashing, terutama ketika rentang nilai yang dimungkinkan sangat besar dibandingkan dengan jumlah nilai yang di-hash. Cara umum untuk menangani collision adalah dengan teknik chaining (menggunakan daftar terkait) atau probing (mencari slot kosong berikutnya dalam tabel hash).

4) Load Factor (Faktor Beban)

Load factor adalah rasio antara jumlah item yang disimpan dalam tabel hash dan jumlah sel yang tersedia dalam tabel. Faktor beban dapat memberikan indikasi tentang seberapa padat tabel hash dan seberapa sering collision terjadi. Load factor yang rendah menunjukkan tabel hash yang jarang diisi dan mungkin membuang memori, sedangkan load factor yang tinggi dapat menyebabkan peningkatan collision.

5) **Collision Resolution (Penyelesaian Kolisi)**

Collision resolution adalah proses menangani collision yang terjadi dalam struktur data hashing. Teknik-teknik penyelesaian kolisi mencakup chaining (menggunakan daftar terkait untuk menampung nilai-nilai yang berkolisi pada slot hash yang sama) dan probing (mencari slot kosong berikutnya dalam tabel hash untuk menempatkan nilai-nilai yang berkolisi).

6) **Open Addressing (Pendekatan Buka)**

Open addressing adalah pendekatan dalam penyelesaian kolisi di mana nilai-nilai yang berkolisi tidak disimpan dalam struktur data tambahan seperti daftar terkait, tetapi dicari lokasi kosong di dalam tabel hash untuk menempatkan nilai-nilai yang berkolisi.

7) **Cryptographic Hash Function (Fungsi Hash Kriptografis)**

Cryptographic hash function adalah jenis fungsi hash yang digunakan dalam kriptografi untuk menghasilkan nilai hash yang sangat sulit untuk dibalikkan atau dipalsukan. Fungsi hash kriptografis sering digunakan dalam aplikasi keamanan seperti penyandian data dan verifikasi integritas.

Memahami istilah-istilah ini penting dalam memahami konsep dan implementasi hashing dalam Python, serta dalam penggunaan struktur data hashing seperti dictionary dan set.

Kesimpulan

Implementasi hashing dalam Python, terutama menggunakan dictionary dan set, memungkinkan untuk pengelolaan data yang efisien dengan kompleksitas waktu yang konstan dalam banyak operasi. Namun, perlu diingat bahwa efisiensi ini tergantung pada kualitas fungsi hash yang digunakan dan seberapa baik manajemen kollision hashnya. Hashing adalah alat yang sangat berguna dalam pemrograman untuk mempercepat operasi pencarian, penambahan, dan penghapusan dalam struktur data seperti dictionary dan set. Dengan memahami konsep dan cara kerja hashing, programmer dapat menggunakan alat ini secara efektif dalam pengembangan perangkat lunak mereka.