Nama :adityah candra pratama

Kelas 2025 B

Nim;25091397046

Analogi (Dalam Kehidupan Sehari-hari)

Hash Table = Rak penyimpanan dengan banyak laci bernomor.

Key (misalnya "k0", "k1", dan seterusnya) = Label barang.

Fungsi hash = Cara menghitung laci pertama yang harus dicoba.

Linear Probing = Jika laci yang dituju penuh, maka barang akan mencoba laci berikutnya secara berurutan.

Collision = Dua barang ingin masuk ke laci yang sama.

Load Factor = Perbandingan antara jumlah laci yang sudah terisi dengan total laci yang tersedia.

Semakin tinggi load factor, semakin besar kemungkinan terjadi collision.

Struktur Fungsi Utama

1. plan_inserts()

Fungsi ini bertugas:

- Membuat hash table kosong
- Memproses semua key satu per satu
- Menyimpan setiap langkah ke dalam frame_data
- Menghitung load factor setiap selesai insert

Fungsi ini bisa diibaratkan sebagai “pembuat skenario animasi”.

2. main()

Fungsi ini bertugas:

- Mengatur tampilan grafik
- Membuat kotak-kotak bucket
- Menampilkan grafik load factor
- Mengatur animasi
- Mengatur kontrol keyboard

Fungsi ini adalah bagian visual dan interaksi pengguna.

Visualisasi

Tampilan terdiri dari dua bagian:

1. Bagian atas → Hash Table
    - Setiap kotak mewakili satu bucket
    - Bucket yang sedang dicek akan diberi highlight
    - Jika terjadi collision, kotak akan lebih gelap
2. Bagian bawah → Grafik Load Factor
    - Sumbu X = jumlah insert
    - Sumbu Y = load factor
    - Grafik menunjukkan seberapa penuh hash table

Di atas tabel terdapat informasi:

- Step ke berapa
- Key yang sedang diproses
- Index awal
- Index saat ini
- Jumlah probe
- Load factor
- Fase proses

Kontrol Keyboard

Saat animasi berjalan, pengguna bisa mengontrol dengan keyboard:

SPACE → Pause / lanjutkan animasi

Panah kanan → Jika pause, maju satu langkah

Panah kiri → Jika pause, mundur satu langkah

R → Restart dari awal

ESC / Q → Keluar dari program

Dengan fitur ini, proses collision bisa diamati secara detail langkah demi langkah.

Load Factor

Load factor dihitung dengan rumus:

Jumlah slot terisi / Total slot

Jika load factor mendekati 1 (100%), maka kemungkinan collision meningkat.

Secara teori:

- Jika load factor < 0.7 → performa masih baik
- Jika load factor mendekati 1 → probing menjadi lebih panjang

Kompleksitas Waktu

Dalam kondisi ideal (jarang collision):

Insert → O(1)

Dalam kondisi terburuk (banyak collision):

Insert → O(n)

Semakin tinggi load factor, semakin besar kemungkinan mendekati kasus terburuk.

Contoh Sederhana

Misal terdapat 5 slot (0–4).

Key pertama → masuk ke slot hasil hash → kosong → simpan.

Key kedua → masuk ke slot hasil hash → jika penuh → geser ke slot berikutnya.

Jika semua slot penuh, maka tidak bisa menambah data lagi.

Dalam program ini diasumsikan jumlah key tidak melebihi kapasitas tabel.

Inti Program

Program ini adalah simulasi interaktif untuk memahami:

- Cara kerja hash function
- Bagaimana collision terjadi
- Bagaimana linear probing menyelesaikan collision
- Bagaimana load factor memengaruhi performa

Dengan bantuan visualisasi dan kontrol keyboard, konsep yang biasanya abstrak menjadi lebih mudah dipahami.

Kesimpulan

Program ini merupakan alat bantu pembelajaran struktur data khususnya Hash Table dengan metode Linear Probing.

Melalui analogi sederhana dan visualisasi animasi, mahasiswa dapat memahami proses insert, collision, probing, serta pengaruh load factor terhadap performa hash table.

Program ini sangat efektif sebagai media pembelajaran interaktif untuk mata kuliah Struktur Data.