

LAPORAN PRAKTIKUM

MODUL 5

HASH TABLE



Disusun oleh:

Arjun Werdho Kumoro

NIM : 2311102009

Dosen Pengampu:

Wahyu Andi Saputra, S.Pd., M.Eng.

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

BAB I

TUJUAN PRAKTIKUM

1. Mahasiswa mampu menjelaskan definisi dan konsep dari Hash Code
2. Mahasiswa mampu menerapkan Hash Code kedalam pemrograman

BAB II

DASAR TEORI

Hash table merupakan struktur data yang secara asosiatif menyimpan data. Dalam hal ini, data disimpan dalam format array, di mana setiap nilai data memiliki nilai indeks uniknya sendiri. Akses data akan menjadi sangat cepat jika Anda mengetahui indeks dari data yang diinginkan. Dengan demikian, hash table menjadi struktur data di mana operasi penyisipan dan pencarian data terjadi sangat cepat terlepas dari ukuran data tersebut. Hash table menggunakan array sebagai media penyimpanan dan tekniknya untuk menghasilkan indeks suatu elemen yang dimasukkan atau ditempatkan.

Fungsi utamanya pada data adalah mempercepat proses akses data. Hal ini berkaitan dengan peningkatan data dalam jumlah besar yang diproses oleh jaringan data global dan lokal. Hash table adalah solusi untuk membuat proses akses data lebih cepat dan memastikan bahwa data dapat dipertukarkan dengan aman. Di dalam banyak bidang, hash table dikembangkan dan digunakan karena menawarkan kelebihan dalam efisiensi waktu operasi, mulai dari pengarsipan hingga pencarian data. Contohnya adalah bidang jaringan komputer yang mengembangkannya menjadi hybrid open hash table, yang kemudian dipakai untuk memproses jaringan komputer. Untuk membuat hash table, sepotong memori perlu diblokir dengan cara yang sama seperti saat membuat array. Anda perlu membuat indeks yang didasarkan pada kunci dengan menggunakan fungsi hash karena indeks yang dihasilkan harus sesuai dengan potongan memori. Ada dua pemeriksaan yang dibutuhkan saat menempatkan data baru pada hash table, yaitu nilai hash dari kunci dan bagaimana nilainya dibandingkan dengan objek lain. Pemeriksaan ini diperlukan saat membuatnya dengan Python karena saat data dimasukkan, kunci akan di-hash dan di-mask agar diubah menjadi larik atau indeks yang efisien.

BAB III

GUIDED

1. GUIDED 1

SOURCE CODE

```
#include <iostream>
using namespace std;
const int MAX_SIZE = 10;
// Fungsi hash sederhana
int hash_func(int key)
{
    return key % MAX_SIZE;
}
// Struktur data untuk setiap node
struct Node
{
    int key;
    int value;
    Node *next;
    Node(int key, int value) : key(key), value(value),
                                next(nullptr) {}
};
// Class hash table
class HashTable
{
private:
    Node **table;

public:
    HashTable()
    {
        table = new Node *[MAX_SIZE]();
    }
};
```

```

~HashTable()
{
    for (int i = 0; i < MAX_SIZE; i++)
    {
        Node *current = table[i];
        while (current != nullptr)
        {
            Node *temp = current;
            current = current->next;
            delete temp;
        }
    }
    delete[] table;
}

// Insertion
void insert(int key, int value)
{
    int index = hash_func(key);
    Node *current = table[index];
    while (current != nullptr)
    {
        if (current->key == key)
        {
            current->value = value;
            return;
        }
        current = current->next;
    }
    Node *node = new Node(key, value);
    node->next = table[index];
    table[index] = node;
}

// Searching
int get(int key)

```

```

{
    int index = hash_func(key);
    Node *current = table[index];
    while (current != nullptr)
    {
        if (current->key == key)
        {
            return current->value;
        }
        current = current->next;
    }
    return -1;
}

// Deletion
void remove(int key)
{
    int index = hash_func(key);
    Node *current = table[index];
    Node *prev = nullptr;
    while (current != nullptr)
    {
        if (current->key == key)
        {
            if (prev == nullptr)
            {
                table[index] = current->next;
            }
            else
            {
                prev->next = current->next;
            }
            delete current;
            return;
        }
    }
}

```

```

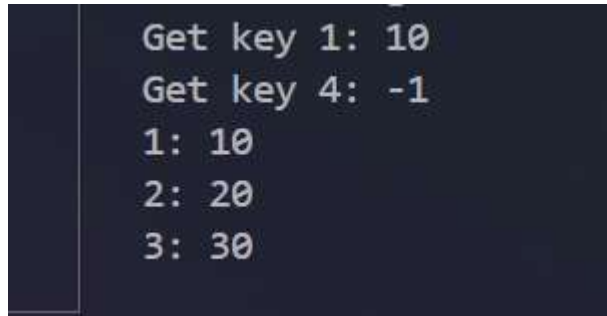
        prev = current;
        current = current->next;

    }
}
// Traversal
void traverse()
{
    for (int i = 0; i < MAX_SIZE; i++)
    {
        Node *current = table[i];
        while (current != nullptr)
        {
            cout << current->key << ": " << current->value
                << endl;
            current = current->next;
        }
    }
}
};
int main()
{
    HashTable ht;
    // Insertion
    ht.insert(1, 10);
    ht.insert(2, 20);
    ht.insert(3, 30);
    // Searching
    cout << "Get key 1: " << ht.get(1) << endl;
    cout << "Get key 4: " << ht.get(4) << endl;
    // Deletion
    ht.remove(4);
    // Traversal
    ht.traverse();
    return 0;
}

```

```
}
```

SCREENSHOOT PROGRAM

A screenshot of a terminal window with a dark background. The text displayed is as follows:

```
Get key 1: 10
Get key 4: -1
1: 10
2: 20
3: 30
```

DESKRIPSI PROGRAM

Node: Setiap elemen dalam tabel hash disimpan dalam bentuk node. Setiap node memiliki dua atribut, yaitu kunci dan nilai. Selain itu, setiap node juga memiliki pointer ke node berikutnya dalam kasus terjadi tabrakan (kunci yang sama).

2. GUIDED 2

SOURCE CODE

```
#include <iostream>
#include <string>
#include <vector>
using namespace std;
const int TABLE_SIZE = 11;
string name;
string phone_number;
class HashNode
{
public:
    string name;
    string phone_number;
    HashNode(string name, string phone_number)
    {
```



```

        this->name = name;
        this->phone_number = phone_number;
    }
};

class HashMap
{
private:
    vector<HashNode *> table[TABLE_SIZE];

public:
    int hashFunc(string key)
    {
        int hash_val = 0;
        for (char c : key)
        {
            hash_val += c;
        }
        return hash_val % TABLE_SIZE;
    }
    void insert(string name, string phone_number)
    {
        int hash_val = hashFunc(name);
        for (auto node : table[hash_val])
        {
            if (node->name == name)
            {
                node->phone_number = phone_number;
                return;
            }
        }
        table[hash_val].push_back(new HashNode(name,
                                                    phone_number));
    }
    void remove(string name)

```

```

    {
        int hash_val = hashFunc(name);
        for (auto it = table[hash_val].begin(); it !=
table[hash_val].end();
            it++)
        {
            if ((*it)->name == name)
            {
                table[hash_val].erase(it);
                return;
            }
        }
    }
string searchByName(string name)
{
    int hash_val = hashFunc(name);
    for (auto node : table[hash_val])
    {
        if (node->name == name)
        {
            return node->phone_number;
        }
    }
    return "";
}
void print()
{
    for (int i = 0; i < TABLE_SIZE; i++)
    {
        cout << i << ": ";
        for (auto pair : table[i])
        {
            if (pair != nullptr)

```

```

        {
            cout << "[" << pair->name << ", " << pair-
>phone_number << "];";
        }
    }
    cout << endl;
}

};

int main()
{
    HashMap employee_map;
    employee_map.insert("Mistah", "1234");
    employee_map.insert("Pastah", "5678");
    employee_map.insert("Ghana", "91011");
    cout << "Nomer Hp Mistah : "
        << employee_map.searchByName("Mistah") << endl;
    cout << "Phone Hp Pastah : "
        << employee_map.searchByName("Pastah") << endl;
    employee_map.remove("Mistah");
    cout << "Nomer Hp Mistah setelah dihapus : "
        << employee_map.searchByName("Mistah") << endl
        << endl;
    cout << "Hash Table : " << endl;
    employee_map.print();
    return 0;
}

```

SCREENSHOOT PROGRAM

```
Nomer Hp Mistah : 1234
Phone Hp Pastah : 5678
Nomer Hp Mistah setelah dihapus :

Hash Table :
0:
1:
2:
3:
4: [Pastah, 5678]
5:
6: [Ghana, 91011]
7:
8:
9:
10:
```

DESKRIPSI PROGRAM

HashMap: Ini adalah kelas yang mewakili struktur data HashMap. Ini memiliki atribut table yang merupakan array dari vektor HashNode. Fungsi-fungsi utama dalam kelas ini adalah: hashFunc: Fungsi ini digunakan untuk menghitung nilai hash dari kunci yang diberikan. Ini dilakukan dengan menjumlahkan nilai ASCII dari setiap karakter dalam kunci dan mengembalikan hasil modulo dengan ukuran tabel. insert: Fungsi ini digunakan untuk memasukkan pasangan kunci-nilai baru ke dalam HashMap. Jika kunci sudah ada, nilai yang ada akan diperbarui..

UNGUIDED

1. UNGUIDED 1

Implementasikan hash table untuk menyimpan data mahasiswa. Setiap mahasiswa memiliki NIM dan nilai. Implementasikan fungsi untuk menambahkan data baru, menghapus data, mencari data berdasarkan NIM, dan mencari data berdasarkan nilai. Dengan ketentuan :

- a. Setiap mahasiswa memiliki NIM dan nilai.
- b. Program memiliki tampilan pilihan menu berisi poin C.
- c. Implementasikan fungsi untuk menambahkan data baru, menghapus data, mencari data berdasarkan NIM, dan mencari data berdasarkan rentang nilai (80 – 90).

SOURCE CODE

```
#include <iostream>
#include <vector>
#include <list>
using namespace std;

struct Mahasiswa {
    string nim;
    int nilai;
};

const int hashTableSize = 100;

class HashTable {
private:
    vector<list<Mahasiswa>> table;

    int hashFunction(const string& key) {
```

```

        int sum = 0;
        for (char c : key) {
            sum += c;
        }
        return sum % hashTableSize;
    }

public:
    HashTable() {
        table.resize(hashTableSize);
    }

    // Fungsi untuk menambahkan data baru
    void tambahData(const string& nim, int nilai) {
        Mahasiswa mahasiswa;
        mahasiswa.nim = nim;
        mahasiswa.nilai = nilai;
        int index = hashFunction(nim);
        table[index].push_back(mahasiswa);
    }

    // Fungsi untuk menghapus data
    void hapusData(const string& nim) {
        int index = hashFunction(nim);
        for (auto it = table[index].begin(); it !=
table[index].end(); ++it) {
            if ((*it).nim == nim) {
                table[index].erase(it);
                break;
            }
        }
    }

    // Fungsi untuk mencari data berdasarkan NIM

```

```

        void cariByNIM(const string& nim) {
            int index = hashFunction(nim);
            for (auto it = table[index].begin(); it !=
table[index].end(); ++it) {
                if ((*it).nim == nim) {
                    cout << "Data ditemukan - NIM: " << (*it).nim <<
", Nilai: " << (*it).nilai << endl;
                    return;
                }
            }
            cout << "Data tidak ditemukan" << endl;
        }

// Fungsi untuk mencari data berdasarkan rentang nilai 80-90
void cariByRange() {
    for (int i = 0; i < hashTableSize; ++i) {
        for (auto it = table[i].begin(); it !=
table[i].end(); ++it) {
            if ((*it).nilai >= 80 && (*it).nilai <= 90) {
                cout << "NIM: " << (*it).nim << ", Nilai: "
<< (*it).nilai << endl;
            }
        }
    }
}

};

int main() {
    HashTable hashTable;
    int choice, nilai;
    string nim;

    do {
        cout << "\nMenu:\n";

```

```

        cout << "1. Add Data\n";
        cout << "2. Delete Data\n";
        cout << "3. Search data by NIM\n";
        cout << "4. Search data by score 80 - 90\n";
        cout << "Input Choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                cout << "Masukkan NIM: ";
                cin >> nim;
                cout << "Masukkan Nilai: ";
                cin >> nilai;
                hashTable.tambahData(nim, nilai);
                break;
            case 2:
                cout << "Masukkan NIM untuk dihapus: ";
                cin >> nim;
                hashTable.hapusData(nim);
                break;
            case 3:
                cout << "Masukkan NIM yang ingin dicari: ";
                cin >> nim;
                hashTable.cariByNIM(nim);
                break;
            case 4:
                cout << "Mahasiswa dengan nilai antara 80 -
90:\n";

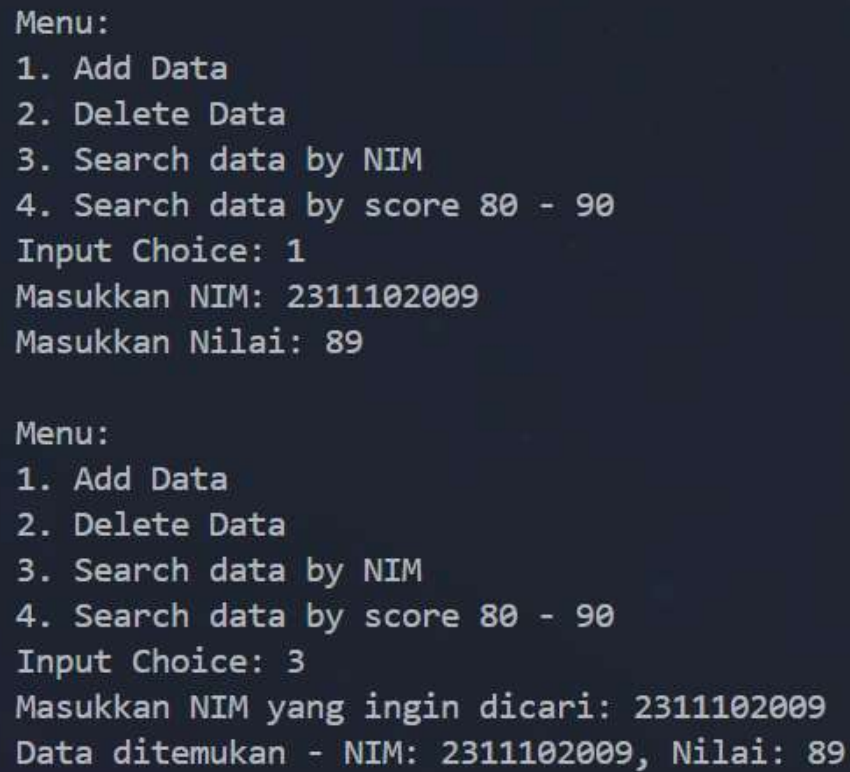
                hashTable.cariByRange();
                break;
            default:
                cout << "Pilihan tidak ada.\n";
        }
    } while (choice != 5);

```



```
    return 0;  
}
```

SCREENSHOOT PROGRAM



```
Menu:  
1. Add Data  
2. Delete Data  
3. Search data by NIM  
4. Search data by score 80 - 90  
Input Choice: 1  
Masukkan NIM: 2311102009  
Masukkan Nilai: 89  
  
Menu:  
1. Add Data  
2. Delete Data  
3. Search data by NIM  
4. Search data by score 80 - 90  
Input Choice: 3  
Masukkan NIM yang ingin dicari: 2311102009  
Data ditemukan - NIM: 2311102009, Nilai: 89
```

Menu menambah dan mencari data dengan nim

```
Menu:
1. Add Data
2. Delete Data
3. Search data by NIM
4. Search data by score 80 - 90
Input Choice: 4
Mahasiswa dengan nilai antara 80 - 90:
NIM: 2311102009, Nilai: 89

Menu:
1. Add Data
2. Delete Data
3. Search data by NIM
4. Search data by score 80 - 90
Input Choice: 2
Masukkan NIM untuk dihapus: 2311102009
```

Menu berdasarkan nilai 80 – 90 dan menghapus data

DESKRIPSI PROGRAM

Kelas HashTable: Kelas HashTable mengimplementasikan tabel hash menggunakan vektor dari daftar berantai. Setiap elemen vektor adalah daftar berantai yang berisi objek Mahasiswa. Kelas ini memiliki beberapa metode, seperti tambahData untuk menambahkan data baru, hapusData untuk menghapus data, cariByNIM untuk mencari data berdasarkan NIM, dan cariByRange untuk mencari data dalam rentang nilai tertentu.

Fungsi Utama: Fungsi main adalah titik masuk program. Di dalamnya, kita membuat objek HashTable dan menggunakan loop do-while untuk menampilkan menu pilihan kepada pengguna. Pengguna dapat memilih untuk menambahkan data, menghapus data, mencari data berdasarkan NIM, atau mencari data dalam rentang nilai tertentu.

BAB IV

KESIMPULAN

Setelah melakukan pembelajaran mengenai Hash Table di Bahasa Pemrograman C++ berikut poin utama yang telah dipelajari :

1. Keuntungan utama dari hash table dibandingkan struktur data lainnya adalah efisiensi dan kecepatan.
2. Waktu yang dibutuhkan untuk mengakses sebuah elemen cukup cepat sehingga bisa lebih diandalkan.
3. Jadi, tidak perlu memakan waktu atau usaha besar untuk menyimpan dan mencari data yang diperlukan.

DAFTAR PUSTAKA

- PT. Algoritma Data Indonesia. (2022, 22 Maret) Apa itu Hash Table dan Bagaimana Penggunaannya?. diakses pada 11 Mei 2024 dari <https://algorit.ma/blog/hash-table-adalah-2022/>