Laporan Praktikum

Modul 10

Multi Linked List

Disususn Oleh:

Muhammad Ikhsan Al Hakim (2311104064)

S1SE07-02

Dosen:

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

1. Tujuan

a. Memahami penggunaan *Multi Linked List.*
b. Mengimplementasi *Multi Linked List* dalam beberapa studi kasus.

2. Landasan Teori

Multi Linked List

merupakan sekumpulan list yang berbeda yang memiliki suatu keterhubungan satu sama lain. Tiap elemen dalam multi link list dapat membentuk list sendiri.Biasanya ada yang bersifat sebagai list induk dan list anak.

3. Guided

Guided 1

```cpp
#include <iostream>
#include <string>

using namespace std;


struct Node {
    int data;
    Node* next;
    Node* child;

    Node(int val) : data(val), next(nullptr), child(nullptr) {}
};


class MultiLinkedList {
private:
    Node* head;

public:
    MultiLinkedList() : head(nullptr) {}


    void addParent(int data) {
        Node* newNode = new Node(data);
        newNode->next = head;
        head = newNode;
    }


    void addChild(int parentData, int childData) {
        Node* parent = head;
        while (parent != nullptr && parent->data != parentData) {
            parent = parent->next;
        }
        if (parent != nullptr) {
            Node* newChild = new Node(childData);
            newChild->next = parent->child;
            parent->child = newChild;
        } else {
            cout << "Parent not found!" << endl;
        }
    }


    void display() {
        Node* current = head;
        while (current != nullptr) {
            cout << "Parent: " << current->data << " -> ";
            Node* child = current->child;
            while (child != nullptr) {
                cout << child->data << " ";
                child = child->next;
            }
            cout << endl;
            current = current->next;
        }
    }

    ~MultiLinkedList() {

        while (head != nullptr) {
            Node* temp = head;
            head = head->next;


            while (temp->child != nullptr) {
                Node* childTemp = temp->child;
                temp->child = temp->child->next;
                delete childTemp;
            }
            delete temp;
        }
    }
};

int main() {
    MultiLinkedList mList;

    mList.addParent(1);
    mList.addParent(2);
    mList.addParent(3);

    mList.addChild(1, 10);
    mList.addChild(1, 11);
    mList.addChild(2, 20);
    mList.addChild(2, 20);
    mList.addChild(3, 30);
    mList.addChild(3, 30);
    mList.display();

    return 0;
}
```

Output:

```
Parent: 3 -> 30 30
Parent: 2 -> 20 20
Parent: 1 -> 11 10
PS D:\buat struktur data\pertemuan 10>
```

Guided 2

```cpp
#include <iostream>
#include <string>

using namespace std;


struct EmployeeNode {
    string name;
    EmployeeNode* next;
    EmployeeNode* subordinate;

    EmployeeNode(string empName) : name(empName), next(nullptr), subordinate(nullptr) {}
};


class EmployeeList {
private:
    EmployeeNode* head;

public:
    EmployeeList() : head(nullptr) {}


    void addEmployee(string name) {
        EmployeeNode* newEmployee = new EmployeeNode(name);
        newEmployee->next = head;
        head = newEmployee;
    }


    void addSubordinate(string managerName, string subordinateName) {
        EmployeeNode* manager = head;
        while (manager != nullptr && manager->name != managerName) {
            manager = manager->next;
        }
        if (manager != nullptr) {
            EmployeeNode* newSubordinate = new EmployeeNode(subordinateName);
            newSubordinate->next = manager->subordinate;
            manager->subordinate = newSubordinate;
        } else {
            cout << "Manager not found!" << endl;
        }
    }


    void display() {
        EmployeeNode* current = head;
        while (current != nullptr) {
            cout << "Manager: " << current->name << " -> ";
            EmployeeNode* sub = current->subordinate;
            while (sub != nullptr) {
                cout << sub->name << " ";
                sub = sub->next;
            }
            cout << endl;
            current = current->next;
        }
    }

    ~EmployeeList() {

        while (head != nullptr) {
            EmployeeNode* temp = head;
            head = head->next;


            while (temp->subordinate != nullptr) {
                EmployeeNode* subTemp = temp->subordinate;
                temp->subordinate = temp->subordinate->next;
                delete subTemp;
            }
            delete temp;
        }
    }
};

int main() {
    EmployeeList empList;

    empList.addEmployee("Alice");
    empList.addEmployee("Bob");
    empList.addEmployee("Charlie");

    empList.addSubordinate("Alice", "David");
    empList.addSubordinate("Alice", "Eve");
    empList.addSubordinate("Bob", "Frank");

    empList.addSubordinate("Charlie", "Frans");
    empList.addSubordinate("Charlie", "Brian");

    empList.display();

    return 0;
}
```

Output:

```
Manager: Charlie -> Brian Frans
Manager: Bob -> Frank
Manager: Alice -> Eve David
PS D:\buat struktur data\pertemuan 10>
```

Guided 3

Output:

```
Initial employee list:
Manager: Charlie ->
Manager: Bob -> Frank
Manager: Alice -> Eve David
Subordinate David deleted from Alice.
Employee Charlie deleted.

Updated employee list:
Manager: Bob -> Frank
Manager: Alice -> Eve
PS D:\buat struktur data\pertemuan 10>
```

4. Unguided

Unguided 1

```cpp
#include <iostream>
#include <string>
using namespace std;

struct Proyek {
    string namaProyek;
    int durasi;
    Proyek* nextProyek;
};

struct Pegawai {
    string namaPegawai;
    string idPegawai;
    Proyek* headProyek;
    Pegawai* nextPegawai;
};

Pegawai* headPegawai = nullptr;

void tambahPegawai(string nama, string id) {
    Pegawai* pegawaiBaru = new Pegawai;
    pegawaiBaru->namaPegawai = nama;
    pegawaiBaru->idPegawai = id;
    pegawaiBaru->headProyek = nullptr;
    pegawaiBaru->nextPegawai = headPegawai;
    headPegawai = pegawaiBaru;
}

void tambahProyek(string idPegawai, string namaProyek, int durasi) {
    Pegawai* currPegawai = headPegawai;
    while (currPegawai != nullptr) {
        if (currPegawai->idPegawai == idPegawai) {
            Proyek* proyekBaru = new Proyek;
            proyekBaru->namaProyek = namaProyek;
            proyekBaru->durasi = durasi;
            proyekBaru->nextProyek = currPegawai->headProyek;
            currPegawai->headProyek = proyekBaru;
            return;
        }
        currPegawai = currPegawai->nextPegawai;
    }
    cout << "Pegawai dengan ID " << idPegawai << " tidak ditemukan!\n";
}

void hapusProyek(string idPegawai, string namaProyek) {
    Pegawai* currPegawai = headPegawai;
    while (currPegawai != nullptr) {
        if (currPegawai->idPegawai == idPegawai) {
            Proyek* currProyek = currPegawai->headProyek;
            Proyek* prevProyek = nullptr;
            while (currProyek != nullptr) {
                if (currProyek->namaProyek == namaProyek) {
                    if (prevProyek == nullptr) {
                        currPegawai->headProyek = currProyek->nextProyek;
                    } else {
                        prevProyek->nextProyek = currProyek->nextProyek;
                    }
                    delete currProyek;
                    return;
                }
                prevProyek = currProyek;
                currProyek = currProyek->nextProyek;
            }
        }
        currPegawai = currPegawai->nextPegawai;
    }
    cout << "Proyek " << namaProyek << " tidak ditemukan untuk pegawai dengan ID " << idPegawai << "!\n";
}

void tampilkanData() {
    Pegawai* currPegawai = headPegawai;
    while (currPegawai != nullptr) {
        cout << "Pegawai: " << currPegawai->namaPegawai << " (ID: " << currPegawai->idPegawai << ")\n";
        Proyek* currProyek = currPegawai->headProyek;
        while (currProyek != nullptr) {
            cout << "  Proyek: " << currProyek->namaProyek << " (Durasi: " << currProyek->durasi << " bulan)\n";
            currProyek = currProyek->nextProyek;
        }
        currPegawai = currPegawai->nextPegawai;
    }
}

int main() {
    int pilihan;
    do {
        cout << "\nMenu:\n";
        cout << "1. Tambah Pegawai\n";
        cout << "2. Tambah Proyek\n";
        cout << "3. Hapus Proyek\n";
        cout << "4. Tampilkan Data\n";
        cout << "5. Keluar\n";
        cout << "Pilihan: ";
        cin >> pilihan;
        cin.ignore();

        if (pilihan == 1) {
            string nama, id;
            cout << "Masukkan Nama Pegawai: ";
            getline(cin, nama);
            cout << "Masukkan ID Pegawai: ";
            getline(cin, id);
            tambahPegawai(nama, id);
        } else if (pilihan == 2) {
            string id, namaProyek;
            int durasi;
            cout << "Masukkan ID Pegawai: ";
            getline(cin, id);
            cout << "Masukkan Nama Proyek: ";
            getline(cin, namaProyek);
            cout << "Masukkan Durasi Proyek (bulan): ";
            cin >> durasi;
            cin.ignore();
            tambahProyek(id, namaProyek, durasi);
        } else if (pilihan == 3) {
            string id, namaProyek;
            cout << "Masukkan ID Pegawai: ";
            getline(cin, id);
            cout << "Masukkan Nama Proyek yang akan dihapus: ";
            getline(cin, namaProyek);
            hapusProyek(id, namaProyek);
        } else if (pilihan == 4) {
            tampilkanData();
        } else if (pilihan != 5) {
            cout << "Pilihan tidak valid.\n";
        }
    } while (pilihan != 5);

    return 0;
}
```

Output:

```
Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 1
Masukkan Nama Pegawai: Andi
Masukkan ID Pegawai: P001

Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 1
Masukkan Nama Pegawai: Budi
Masukkan ID Pegawai: P002

Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 1
Masukkan Nama Pegawai: Citra
Masukkan ID Pegawai: P003
```

```
Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 2
Masukkan ID Pegawai: P001
Masukkan Nama Proyek: Aplikasi Mobile
Masukkan Durasi Proyek (bulan): 12

Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 2
Masukkan ID Pegawai: P002
Masukkan Nama Proyek: Sistem Akutansi
Masukkan Durasi Proyek (bulan): 8

Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 2
Masukkan ID Pegawai: P003
Masukkan Nama Proyek: E-Commerce
Masukkan Durasi Proyek (bulan): 10
```

```
Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 2
Masukkan ID Pegawai: P001
Masukkan Nama Proyek: Analisis Data
Masukkan Durasi Proyek (bulan): 6

Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 3
Masukkan ID Pegawai: P001
Masukkan Nama Proyek yang akan dihapus: Aplikasi Mobile

Menu:
1. Tambah Pegawai
2. Tambah Proyek
3. Hapus Proyek
4. Tampilkan Data
5. Keluar
Pilihan: 4
Pegawai: Citra (ID: P003)
  Proyek: E-Commerce (Durasi: 10 bulan)
Pegawai: Budi (ID: P002)
  Proyek: Sistem Akutansi (Durasi: 8 bulan)
Pegawai: Andi (ID: P001)
  Proyek: Analisis Data (Durasi: 6 bulan)
```

Unguided 2

```cpp
#include <iostream>
#include <string>
using namespace std;

struct Buku {
    string judulBuku;
    string tanggalPengembalian;
    Buku* nextBuku;
};

struct Anggota {
    string namaAnggota;
    string idAnggota;
    Buku* headBuku;
    Anggota* nextAnggota;
};

Anggota* headAnggota = nullptr;

void tambahAnggota(string nama, string id) {
    Anggota* anggotaBaru = new Anggota;
    anggotaBaru->namaAnggota = nama;
    anggotaBaru->idAnggota = id;
    anggotaBaru->headBuku = nullptr;
    anggotaBaru->nextAnggota = headAnggota;
    headAnggota = anggotaBaru;
}

void tambahBuku(string idAnggota, string judulBuku, string tanggalPengembalian) {
    Anggota* currAnggota = headAnggota;
    while (currAnggota != nullptr) {
        if (currAnggota->idAnggota == idAnggota) {
            Buku* bukuBaru = new Buku;
            bukuBaru->judulBuku = judulBuku;
            bukuBaru->tanggalPengembalian = tanggalPengembalian;
            bukuBaru->nextBuku = currAnggota->headBuku;
            currAnggota->headBuku = bukuBaru;
            return;
        }
        currAnggota = currAnggota->nextAnggota;
    }
    cout << "Anggota dengan ID " << idAnggota << " tidak ditemukan!\n";
}

void hapusAnggota(string idAnggota) {
    Anggota* currAnggota = headAnggota;
    Anggota* prevAnggota = nullptr;
    while (currAnggota != nullptr) {
        if (currAnggota->idAnggota == idAnggota) {
            if (prevAnggota == nullptr) {
                headAnggota = currAnggota->nextAnggota;
            } else {
                prevAnggota->nextAnggota = currAnggota->nextAnggota;
            }
            // Hapus semua buku yang dipinjam anggota
            Buku* currBuku = currAnggota->headBuku;
            while (currBuku != nullptr) {
                Buku* temp = currBuku;
                currBuku = currBuku->nextBuku;
                delete temp;
            }
            delete currAnggota;
            return;
        }
        prevAnggota = currAnggota;
        currAnggota = currAnggota->nextAnggota;
    }
    cout << "Anggota dengan ID " << idAnggota << " tidak ditemukan!\n";
}

void tampilkanData() {
    Anggota* currAnggota = headAnggota;
    while (currAnggota != nullptr) {
        cout << "Anggota: " << currAnggota->namaAnggota << " (ID: " << currAnggota->idAnggota << ")\n";
        Buku* currBuku = currAnggota->headBuku;
        while (currBuku != nullptr) {
            cout << "  Buku: " << currBuku->judulBuku << " (Pengembalian: " << currBuku->tanggalPengembalian << ")\n";
            currBuku = currBuku->nextBuku;
        }
        currAnggota = currAnggota->nextAnggota;
    }
}

int main() {
    int pilihan;
    do {
        cout << "\nMenu:\n";
        cout << "1. Tambah Anggota\n";
        cout << "2. Tambah Buku\n";
        cout << "3. Hapus Anggota\n";
        cout << "4. Tampilkan Data\n";
        cout << "5. Keluar\n";
        cout << "Pilihan: ";
        cin >> pilihan;
        cin.ignore();

        if (pilihan == 1) {
            string nama, id;
            cout << "Masukkan Nama Anggota: ";
            getline(cin, nama);
            cout << "Masukkan ID Anggota: ";
            getline(cin, id);
            tambahAnggota(nama, id);
        } else if (pilihan == 2) {
            string id, judulBuku, tanggalPengembalian;
            cout << "Masukkan ID Anggota: ";
            getline(cin, id);
            cout << "Masukkan Judul Buku: ";
            getline(cin, judulBuku);
            cout << "Masukkan Tanggal Pengembalian: ";
            getline(cin, tanggalPengembalian);
            tambahBuku(id, judulBuku, tanggalPengembalian);
        } else if (pilihan == 3) {
            string id;
            cout << "Masukkan ID Anggota yang akan dihapus: ";
            getline(cin, id);
            hapusAnggota(id);
        } else if (pilihan == 4) {
            tampilkanData();
        } else if (pilihan != 5) {
            cout << "Pilihan tidak valid.\n";
        }
    } while (pilihan != 5);

    return 0;
}
```

Output:

```
Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 1
Masukkan Nama Anggota: Rani
Masukkan ID Anggota: A001

Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 1
Masukkan Nama Anggota: Dito
Masukkan ID Anggota: A002

Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 1
Masukkan Nama Anggota: Vina
Masukkan ID Anggota: A003

Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 2
Masukkan ID Anggota: A001
Masukkan Judul Buku: Pemrograman C++
Masukkan Tanggal Pengembalian: 01/12/2024

Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 2
Masukkan ID Anggota: A002
Masukkan Judul Buku: Algoritma Pemrograman
Masukkan Tanggal Pengembalian: 15/12/2024

Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 2
Masukkan ID Anggota: A001
Masukkan Judul Buku: Struktur Data
Masukkan Tanggal Pengembalian: 10/12/2024
Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 3
Masukkan ID Anggota yang akan dihapus: A002

Menu:
1. Tambah Anggota
2. Tambah Buku
3. Hapus Anggota
4. Tampilkan Data
5. Keluar
Pilihan: 4
Anggota: Vina (ID: A003)
Anggota: Rani (ID: A001)
  Buku: Struktur Data (Pengembalian: 10/12/2024)
  Buku: Pemrograman C++ (Pengembalian: 01/12/2024)
```