STRUKTUR DATA
Tugas Kelompok Multilist

NAMA:
UHAMMAD LUTHFI AZIZ SUNARYA 140810230081
BAGAS DIATAMA WARDOYO 140810230061
MUHAMAD HISYAM AZ-ZAHRAN 140810230065

Dikumpulkan tanggal :
26 Mei 2024

Universitas Padjadjaran
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
Program Studi S-1 Teknik Informatika

2024

1. **Source Code Multilist**

```
/*
NAMA ANGGOTA 1      : MUHAMMAD LUTHFI AZIZ SUNARYA 140810230081
NAMA ANGGOTA 2      : BAGAS DIATAMA WARDOYO 140810230061
NAMA ANGGOTA 3      : MUHAMAD HISYAM AZ-ZAHRAN 140810230065
TANGGAL PEMBUATAN   : Minggu, 26 - 05 - 2024
NAMA PROGRRAM       : MULTILIST
DESKRIPSI           : INI ADALAH PROGRAM MULTILIST YANG BERISI DATA PEGAWAI
DENGAN SEMUA FUNGSI PRIMITIVE
*/
#include <iostream>
#include <iomanip>
using namespace std;

struct anak {
    string namaAnak;
    anak *nextAnak;
};

struct karyawan {
    string nip;
    string namaKaryawan;
    int golongan;
    anak *firstAnak;
    karyawan *nextKaryawan;
};

struct divisi {
    string namaDivisi;
    divisi *nextDivisi;
    karyawan *firstKaryawan;
};

typedef anak *ptrAnak;
typedef karyawan *ptrKaryawan;
typedef divisi *ptrDivisi;

typedef ptrDivisi first;

void tabel() {
    for (int i = 0; i < 20; i++) {
        cout << "=";
    }
    cout << endl;
}

void createList(first &list) {
    list = nullptr;
}
```

```
void createDivisi(ptrDivisi &newDivisi) {
    newDivisi = new divisi;
    cout << "masukan nama divisi : ";
    cin >> newDivisi->namaDivisi;
    newDivisi->nextDivisi = nullptr;
}

void createKaryawan(ptrKaryawan &newKaryawan) {
    newKaryawan = new karyawan;
    cout << "masukan nip karyawan : ";
    cin >> newKaryawan->nip;
    cout << "masukan nama karyawan : ";
    cin >> newKaryawan->namaKaryawan;
    newKaryawan->nextKaryawan = nullptr;
    newKaryawan->firstAnak = nullptr;
}

void createAnak(ptrAnak &newAnak) {
    newAnak = new anak;
    cout << "masukan nama anak : ";
    cin >> newAnak->namaAnak;
    newAnak->nextAnak = nullptr;
}

bool searchDivisi(first head, ptrDivisi &pCari, ptrDivisi &pCariDprev,
string key) {
    pCari = head;
    pCariDprev = nullptr;
    bool found = false;
    while (found == false && pCari != nullptr) {
        if (pCari->namaDivisi == key) {
            found = true;
        } else {
            pCariDprev = pCari;
            pCari = pCari->nextDivisi;
        }
    }
    return found;
}

bool searchKaryawan(first head, ptrKaryawan &pCariK, ptrKaryawan
&pCarikprev, string key) {
    ptrDivisi pBantu = head;
    pCarikprev = nullptr;
    bool found = false;
    while (found == false && pBantu != nullptr) {
        pCariK = pBantu->firstKaryawan;
        while (found == false && pCariK != nullptr) {
            if (pCariK->namaKaryawan == key) {
                found = true;
```

```
            } else {
                pCarikprev = pCariK;
                pCariK = pCariK->nextKaryawan;
            }
        }
        pBantu = pBantu->nextDivisi;
    }
    return found;
}

bool searchAnak(ptrKaryawan pCariK, ptrAnak &pCariA, ptrAnak &pCariAprev,
string key) {
    pCariA = pCariK->firstAnak;
    pCariAprev = nullptr;
    bool found = false;
    while (found == false && pCariA != nullptr) {
        if (pCariA->namaAnak == key) {
            found = true;
        } else {
            pCariAprev = pCariA;
            pCariA = pCariA->nextAnak;
        }
    }
    return found;
}

void insertFirstDivisi(ptrDivisi &newDivisi, first &head) {
    if (head == nullptr) {
        head = newDivisi;
    } else {
        newDivisi->nextDivisi = head;
        head = newDivisi;
    }
}

void insertFirstKaryawan(ptrKaryawan &newKaryawan, first pCariD) {
    if (pCariD->firstKaryawan == nullptr) {
        pCariD->firstKaryawan = newKaryawan;
    } else {
        newKaryawan->nextKaryawan = pCariD->firstKaryawan;
        pCariD->firstKaryawan = newKaryawan;
    }
}

void insertFirstAnak(ptrAnak &newAnak, ptrKaryawan pCariK) {
    if (pCariK->firstAnak == nullptr) {
        pCariK->firstAnak = newAnak;
    } else {
        newAnak->nextAnak = pCariK->firstAnak;
        pCariK->firstAnak = newAnak;
    }
```

```
}

void insertLastAnak(ptrAnak &newAnak, ptrKaryawan pCariK) {
    ptrAnak pBantu = pCariK->firstAnak;
    if (pCariK->firstAnak == nullptr) {
        pCariK->firstAnak = newAnak;
    } else {
        while (pBantu->nextAnak != nullptr) {
            pBantu = pBantu->nextAnak;
        }
        pBantu->nextAnak = newAnak;
    }
}

void insertLastKaryawan(ptrKaryawan &newKaryawan, ptrDivisi pCariD) {
    ptrKaryawan pBantu = pCariD->firstKaryawan;
    if (pCariD->firstKaryawan == nullptr) {
        pCariD->firstKaryawan = newKaryawan;
    } else {
        while (pBantu->nextKaryawan != nullptr) {
            pBantu = pBantu->nextKaryawan;
        }
        pBantu->nextKaryawan = newKaryawan;
    }
}

void insertLastDivisi(ptrDivisi &newDivisi, first &head) {
    ptrDivisi pBantu = head;
    if (head == nullptr) {
        head = newDivisi;
    } else {
        while (pBantu->nextDivisi != nullptr) {
            pBantu = pBantu->nextDivisi;
        }
        pBantu->nextDivisi = newDivisi;
    }
}

void insertAfterDivisi(ptrDivisi &newDivisi, first head, ptrDivisi pCariD)
{
    if (pCariD->nextDivisi == nullptr) {
        insertLastDivisi(newDivisi, head);
    } else {
        newDivisi->nextDivisi = pCariD->nextDivisi;
        pCariD->nextDivisi = newDivisi;
    }
}

void insertAfterKaryawan(ptrKaryawan &newKaryawan, first head, ptrKaryawan
pCariK) {
    if (pCariK->nextKaryawan == nullptr) {
```

```
        insertLastKaryawan(newKaryawan, head);
    } else {
        newKaryawan->nextKaryawan = pCariK->nextKaryawan;
        pCariK->nextKaryawan = newKaryawan;
    }
}

void insertAfterAnak(ptrAnak &newAnak, ptrKaryawan pCariK, ptrAnak pCariA)
{
    if (pCariA->nextAnak == nullptr) {
        insertLastAnak(newAnak, pCariK);
    } else {
        newAnak->nextAnak = pCariA->nextAnak;
        pCariA->nextAnak = newAnak;
    }
}

void deleteFirstDivisi(first &head, ptrDivisi &pHapus) {
    if (head == nullptr) {
        cout << "Tidak ada divisi untuk dihapus" << endl;
    }
    pHapus = head;
    head = head->nextDivisi;
    delete pHapus;
}

void deleteFirstKaryawan(ptrDivisi pCariD, ptrKaryawan &pHapus) {
    if (pCariD == nullptr) {
        cout << "Tidak ada karyawan untuk dihapus" << endl;
    }
    if (pCariD->firstKaryawan == nullptr) {
        cout << "Tidak ada karyawan untuk dihapus pada divisi ini" << endl;
    }
    pHapus = pCariD->firstKaryawan;
    pCariD->firstKaryawan = pCariD->firstKaryawan->nextKaryawan;
    delete pHapus;
}

void deleteFirstAnak(ptrKaryawan pCariK, ptrAnak &pHapus) {
    if (pCariK == nullptr) {
        cout << "Tidak ada anak untuk dihapus" << endl;
        return;
    }
    if (pCariK->firstAnak == nullptr) {
        cout << "Tidak ada anak untuk dihapus pada karyawan ini" << endl;
        return;
    }
    pHapus = pCariK->firstAnak;
    pCariK->firstAnak = pCariK->firstAnak->nextAnak;
    delete pHapus;
}
```

```
void deleteLastDivisi(first &head, ptrDivisi &pHapus) {
    if (head == nullptr) {
        cout << "Tidak ada divisi untuk dihapus" << endl;
        return;
    }
    if (head->nextDivisi == nullptr) {
        delete head;
        head = nullptr;
        return;
    }
    ptrDivisi prevDivisi = nullptr;
    pHapus = head;
    while (pHapus->nextDivisi != nullptr) {
        prevDivisi = pHapus;
        pHapus = pHapus->nextDivisi;
    }
    delete pHapus;
    prevDivisi->nextDivisi = nullptr;
}

void deleteLastKaryawan(ptrDivisi pCariD, ptrKaryawan &pHapus) {
    if (pCariD == nullptr) {
        cout << "Tidak ada karyawan untuk dihapus" << endl;
        return;
    }
    if (pCariD->firstKaryawan == nullptr) {
        cout << "Tidak ada karyawan untuk dihapus pada divisi ini" << endl;
        return;
    }
    if (pCariD->firstKaryawan->nextKaryawan == nullptr) {
        delete pCariD->firstKaryawan;
        pCariD->firstKaryawan = nullptr;
        return;
    }
    ptrKaryawan prevKaryawan = nullptr;
    pHapus = pCariD->firstKaryawan;
    while (pHapus->nextKaryawan != nullptr) {
        prevKaryawan = pHapus;
        pHapus = pHapus->nextKaryawan;
    }
    delete pHapus;
    prevKaryawan->nextKaryawan = nullptr;
}

void deleteLastAnak(ptrKaryawan pCariK, ptrAnak &pHapus) {
    if (pCariK == nullptr) {
        cout << "Tidak ada anak untuk dihapus" << endl;
        return;
    }
    if (pCariK->firstAnak == nullptr) {
```

```
        cout << "Tidak ada anak untuk dihapus pada karyawan ini" << endl;
        return;
    }
    if (pCariK->firstAnak->nextAnak == nullptr) {
        delete pCariK->firstAnak;
        pCariK->firstAnak = nullptr;
        return;
    }
    ptrAnak prevAnak = nullptr;
    pHapus = pCariK->firstAnak;
    while (pHapus->nextAnak != nullptr) {
        prevAnak = pHapus;
        pHapus = pHapus->nextAnak;
    }
    delete pHapus;
    prevAnak->nextAnak = nullptr;
}

void deleteAfterDivisi(ptrDivisi &pHapus, ptrDivisi prevDivisi) {
    if (prevDivisi->nextDivisi == nullptr) {
        delete prevDivisi->nextDivisi;
        prevDivisi->nextDivisi = nullptr;
        return;
    }
    pHapus = prevDivisi->nextDivisi;
    prevDivisi->nextDivisi = pHapus->nextDivisi;
    delete pHapus;
}

void deleteAfterKaryawan(ptrKaryawan &pHapus, ptrKaryawan prevKaryawan) {
    if (prevKaryawan->nextKaryawan == nullptr) {
        delete prevKaryawan->nextKaryawan;
        prevKaryawan->nextKaryawan = nullptr;
        return;
    }
    pHapus = prevKaryawan->nextKaryawan;
    prevKaryawan->nextKaryawan = pHapus->nextKaryawan;
    delete pHapus;
}

void deleteAfterAnak(ptrAnak &pHapus, ptrAnak prevAnak) {
    if (prevAnak->nextAnak == nullptr) {
        delete prevAnak->nextAnak;
        prevAnak->nextAnak = nullptr;
        return;
    }
    pHapus = prevAnak->nextAnak;
    prevAnak->nextAnak = pHapus->nextAnak;
    delete pHapus;
}
```

```
void showDivisi(first head) {
    ptrDivisi pBantu = head;
    cout << "Daftar divisi : " << endl;
    while (pBantu != nullptr) {
        cout << pBantu->namaDivisi << endl;
        pBantu = pBantu->nextDivisi;
    }
}

void showKaryawan(first head) {
    ptrDivisi pBantuD = head;
    ptrKaryawan pBantuK;
    while (pBantuD != nullptr) {
        pBantuK = pBantuD->firstKaryawan;
        cout << "Daftar karyawan pada divisi " << pBantuD->namaDivisi << "
: " << endl;
        while (pBantuK != nullptr) {
            cout << pBantuK->nip << " - " << pBantuK->namaKaryawan << " - "
<< pBantuK->golongan << endl;
            pBantuK = pBantuK->nextKaryawan;
        }
        pBantuD = pBantuD->nextDivisi;
    }
}

void showAnak(first head) {
    ptrDivisi pBantuD = head;
    ptrKaryawan pBantuK;
    ptrAnak pBantuA;
    while (pBantuD != nullptr) {
        pBantuK = pBantuD->firstKaryawan;
        while (pBantuK != nullptr) {
            pBantuA = pBantuK->firstAnak;
            cout << "Daftar anak pada karyawan " << pBantuK->namaKaryawan
<< " : " << endl;
            while (pBantuA != nullptr) {
                cout << pBantuA->namaAnak << endl;
                pBantuA = pBantuA->nextAnak;
            }
            pBantuK = pBantuK->nextKaryawan;
        }
        pBantuD = pBantuD->nextDivisi;
    }
}

void menu(first &head) {
    int pilih, pilih2, pilih3;
    ptrDivisi newDivisi, pCariD, pCariDprev;
    ptrKaryawan newKaryawan, pCariK, pCarikprev;
    ptrAnak newAnak, pCariA, pCariAprev;
    string key, key2, key3;
```

```
    bool found = false;
    do {
        tabel();
        cout << "PROGRAM MULTILIST" << endl;
        tabel();
        cout << "1. Insert" << endl;
        cout << "2. Delete" << endl;
        cout << "3. Search" << endl;
        cout << "4. Tampilkan" << endl;
        cout << "0. Keluar" << endl;
        tabel();
        cout << "Masukan pilihan : ";
        cin >> pilih;
        tabel();
        switch (pilih) {
            case 1:
                cout << "Insert" << endl;
                tabel();
                cout << "1. Divisi" << endl;
                cout << "2. Karyawan" << endl;
                cout << "3. Anak" << endl;
                tabel();
                cout << "Masukan pilihan : ";
                cin >> pilih2;
                tabel();
                switch (pilih2) {
                    case 1:
                        createDivisi(newDivisi);
                        insertLastDivisi(newDivisi, head);
                        break;
                    case 2:
                        cout << "Masukan nama divisi : ";
                        cin >> key;
                        found = searchDivisi(head, pCariD, pCariDprev,
key);
                        if (found) {
                            createKaryawan(newKaryawan);
                            insertLastKaryawan(newKaryawan, pCariD);
                        } else {
                            cout << "Divisi tidak ditemukan" << endl;
                        }
                        break;
                    case 3:
                        cout << "Masukan nama karyawan : ";
                        cin >> key2;
                        found = searchKaryawan(head, pCariK, pCarikprev,
key2);
                        if (found) {
                            createAnak(newAnak);
                            insertLastAnak(newAnak, pCariK);
                        } else {
```

```
                        cout << "Karyawan tidak ditemukan" << endl;
                    }
                    break;
                default:
                    cout << "Pilihan tidak ada" << endl;
                    break;
            }
            break;
        case 2:
            cout << "Delete" << endl;
            tabel();
            cout << "1. Divisi" << endl;
            cout << "2. Karyawan" << endl;
            cout << "3. Anak" << endl;
            tabel();
            cout << "Masukan pilihan : ";
            cin >> pilih2;
            tabel();
            switch (pilih2) {
                case 1:
                    cout << "Masukan nama divisi : ";
                    cin >> key;
                    found = searchDivisi(head, pCariD, pCariDprev,
key);
                    if (found) {
                        if (pCariD == head) {
                            deleteFirstDivisi(head, pCariD);
                        } else {
                            deleteAfterDivisi(pCariD, pCariDprev);
                        }
                    } else {
                        cout << "Divisi tidak ditemukan" << endl;
                    }
                    break;
                case 2:
                    cout << "Masukan nama karyawan : ";
                    cin >> key2;
                    found = searchKaryawan(head, pCariK, pCarikprev,
key2);
                    if (found) {
                        if (pCariK == pCariD->firstKaryawan) {
                            deleteFirstKaryawan(pCariD, pCariK);
                        } else {
                            deleteAfterKaryawan(pCariK, pCarikprev);
                        }
                    } else {
                        cout << "Karyawan tidak ditemukan" << endl;
                    }
                    break;
                case 3:
                    cout << "Masukan nama anak : ";
```

```
                        cin >> key3;
                        found = searchKaryawan(head, pCariK, pCarikprev,
key2);
                        if (found) {
                            found = searchAnak(pCariK, pCariA, pCariAprev,
key3);
                            if (found) {
                                if (pCariA == pCariK->firstAnak) {
                                    deleteFirstAnak(pCariK, pCariA);
                                } else {
                                    deleteAfterAnak(pCariA, pCariAprev);
                                }
                            } else {
                                cout << "Anak tidak ditemukan" << endl;
                            }
                        } else {
                            cout << "Karyawan tidak ditemukan" << endl;
                        }
                        break;
                    default:
                        cout << "Pilihan tidak ada" << endl;
                        break;
                }
                break;
            case 3:
                cout << "Search" << endl;
                tabel();
                cout << "1. Divisi" << endl;
                cout << "2. Karyawan" << endl;
                cout << "3. Anak" << endl;
                tabel();
                cout << "Masukan pilihan : ";
                cin >> pilih2;
                tabel();
                switch (pilih2) {
                    case 1:
                        cout << "Masukan nama divisi : ";
                        cin >> key;
                        found = searchDivisi(head, pCariD, pCariDprev,
key);
                        if (found) {
                            cout << "Divisi ditemukan : " << pCariD-
>namaDivisi << endl;
                        } else {
                            cout << "Divisi tidak ditemukan" << endl;
                        }
                        break;
                    case 2:
                        cout << "Masukan nama karyawan : ";
                        cin >> key2;
```

```
                        found = searchKaryawan(head, pCariK, pCarikprev,
key2);
                        if (found) {
                            cout << "Karyawan ditemukan : " << pCariK->nip
<< " - " << pCariK->namaKaryawan << " - " << pCariK->golongan << endl;
                        } else {
                            cout << "Karyawan tidak ditemukan" << endl;
                        }
                        break;
                    case 3:
                        cout << "Masukan nama anak : ";
                        cin >> key3;
                        found = searchKaryawan(head, pCariK, pCarikprev,
key2);
                        if (found) {
                            found = searchAnak(pCariK, pCariA, pCariAprev,
key3);
                            if (found) {
                                cout << "Anak ditemukan : " << pCariA-
>namaAnak << endl;
                            } else {
                                cout << "Anak tidak ditemukan" << endl;
                            }
                        } else {
                            cout << "Karyawan tidak ditemukan" << endl;
                        }
                        break;
                    default:
                        cout << "Pilihan tidak ada" << endl;
                        break;
                }
                break;
            case 4:
                cout << "Tampilkan" << endl;
                tabel();
                cout << "1. Divisi" << endl;
                cout << "2. Karyawan" << endl;
                cout << "3. Anak" << endl;
                tabel();
                cout << "Masukan pilihan : ";
                cin >> pilih2;
                tabel();
                switch (pilih2) {
                    case 1:
                        showDivisi(head);
                        break;
                    case 2:
                        showKaryawan(head);
                        break;
                    case 3:
                        showAnak(head);
```

```
                        break;
                    default:
                        cout << "Pilihan tidak ada" << endl;
                        break;
                }
                break;
            case 0:
                cout << "Keluar" << endl;
                break;
            default:
                cout << "Pilihan tidak ada" << endl;
                break;
        }
    } while (pilih != 0);
}

int main() {
    first head;
    createList(head);
    menu(head);
    return 0;
}
```

Hasil Run

```
====================
PROGRAM MULTILIST
====================
1. Insert
2. Delete
3. Search
4. Tampilkan
0. Keluar
====================
Masukan pilihan : 1
====================
Insert
====================
1. Divisi
2. Karyawan
3. Anak
====================
Masukan pilihan : 1
====================
masukan nama divisi : HONKAI
--------------------
```

```
--------------------
PROGRAM MULTILIST
====================
1. Insert
2. Delete
3. Search
4. Tampilkan
0. Keluar
====================
Masukan pilihan : 1
====================
Insert
====================
1. Divisi
2. Karyawan
3. Anak
====================
Masukan pilihan : 1
====================
masukan nama divisi : GOTEI
```

```
====================
PROGRAM MULTILIST
====================
1. Insert
2. Delete
3. Search
4. Tampilkan
0. Keluar
====================
Masukan pilihan : 2
====================
Delete
====================
1. Divisi
2. Karyawan
3. Anak
====================
Masukan pilihan : 1
====================
Masukan nama divisi : HONKAI
```

```
====================
PROGRAM MULTILIST
====================
1. Insert
2. Delete
3. Search
4. Tampilkan
0. Keluar
====================
Masukan pilihan : 3
====================
Search
====================
1. Divisi
2. Karyawan
3. Anak
====================
Masukan pilihan : 1
====================
Masukan nama divisi : GOTEI
Divisi ditemukan : GOTEI
```

```
====================
PROGRAM MULTILIST
====================
1. Insert
2. Delete
3. Search
4. Tampilkan
0. Keluar
====================
Masukan pilihan : 4
====================
Tampilkan
====================
1. Divisi
2. Karyawan
3. Anak
====================
Masukan pilihan : 1
====================
Daftar divisi :
GOTEI
====================
PROGRAM MULTILIST
====================
1. Insert
2. Delete
3. Search
4. Tampilkan
0. Keluar
====================
Masukan pilihan : 0
====================
Keluar
```