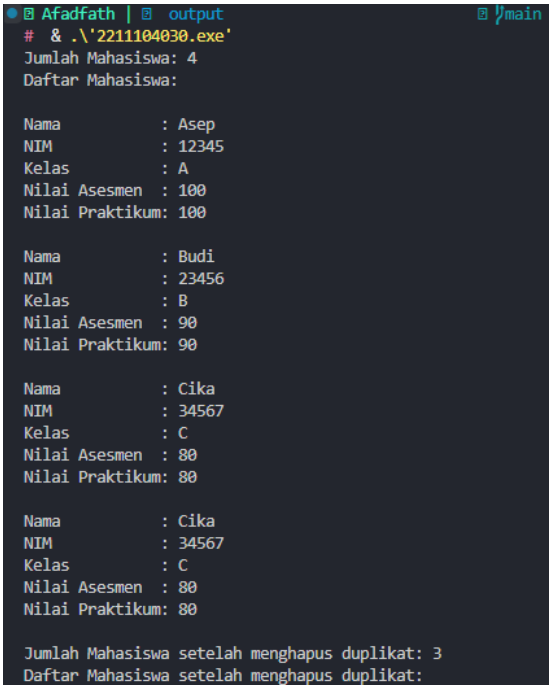
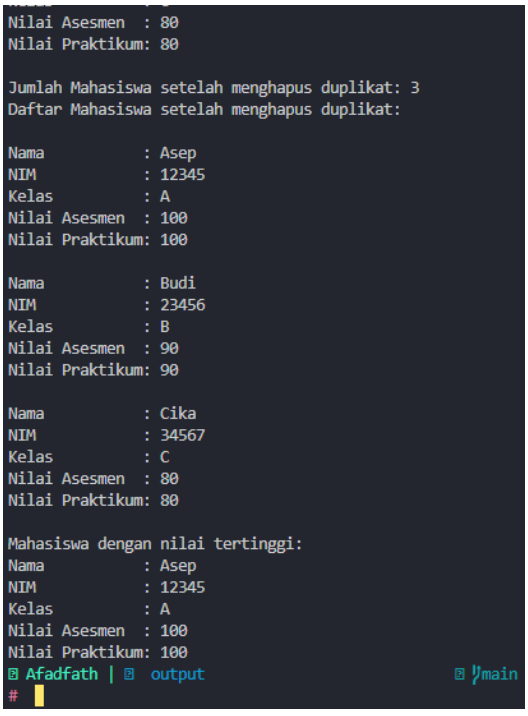
Nama : Afad Fath Musyarof Halim
NIM : 2211104030
Kelas : SE-06-01

Running:





Penjelasan:

- Struktur DLL

```
struct mhsList {
    string nama;
    int nim;
    string kelas;
    int nilai_ass;
    int nilai_prak;
    address next;
    address prev;
};


struct List {
    address first;
    address last;
};
```

- Buat list kosong

```
void newList(List &L) {
    L.first = nullptr;
    L.last = nullptr;
}
```

- Cek apakah list kosong

```
bool isEmpty(const List &L) {
    return (L.first == nullptr && L.last == nullptr);
}
```

- Membuat elemen baru

```
address newElement(string nama, int nim, string kelas, int nilai_ass, int nilai_prak) {
    address mhsBaru = new mhsList;
    mhsBaru->nama = nama;
    mhsBaru->nim = nim;
    mhsBaru->kelas = kelas;
    mhsBaru->nilai_ass = nilai_ass;
    mhsBaru->nilai_prak = nilai_prak;
    mhsBaru->next = nullptr;
    mhsBaru->prev = nullptr;
    return mhsBaru;
}
```

- Memasukan elemen baru ke urutan list paling belakang

```
void insertLast(List &L, address newElement) {
    if (L.first == nullptr) {
        L.first = newElement;
        L.last = newElement;
    } else {
        L.last->next = newElement;
        newElement->prev = L.last;
        L.last = newElement;
    }
}
```

- Menampilkan daftar list

```
void printList(const List &L) {
    if(isEmpty(L)){
        cout << "Tidak ada mahasiswa yang terdaftar" << endl;
        return;
    }

    address current = L.first;
    while (current != nullptr) {
        cout << endl;
        cout << "Nama           : " << current->nama << endl;
        cout << "NIM            : " << current->nim << endl;
        cout << "Kelas          : " << current->kelas << endl;
        cout << "Nilai Asesmen  : " << current->nilai_ass << endl;
        cout << "Nilai Praktikum: " << current->nilai_prak << endl;
        current = current->next;
    }
}
```

- Menghitung ukuran list

```
int length(const List &L) {
    int i = 0;
    address current = L.first;
    while (current != nullptr) {
        i++;
        current = current->next;
    }
    return i;
}
```

- Menampilkan data mahasiswa dengan nilai asesmen paling tinggi dengan membandingkan nilai terlebih dahulu

```
void highest(const List &L){
    if(isEmpty(L)){
        cout << "Tidak ada mahasiswa yang terdaftar" << endl;
        return;
    }

    address CekNilai = L.first;
    address Tertinggi = L.first;

    while (CekNilai != nullptr) {
        if (CekNilai->nilai_ass > Tertinggi->nilai_ass) {
            Tertinggi = CekNilai;
        }
        CekNilai = CekNilai->next;
    }

    cout << "Nama           : " << Tertinggi->nama << endl;
    cout << "NIM            : " << Tertinggi->nim << endl;
    cout << "Kelas          : " << Tertinggi->kelas << endl;
    cout << "Nilai Asesmen  : " << Tertinggi->nilai_ass << endl;
    cout << "Nilai Praktikum: " << Tertinggi->nilai_prak << endl;
}
```

- Menghapus data duplikat berdasarkan nim dengan membandingkan setiap elemen masing masing

```
void removeDuplicates(List &L) {
    if (isEmpty(L)) {
        cout << "Tidak ada mahasiswa yang terdaftar" << endl;
        return;
    }

    address current = L.first;
    while (current != nullptr) {
        address indexing = current->next;
        while (indexing != nullptr) {
            if (indexing->nim == current->nim) {

                address duplicate = indexing;

                if (indexing->prev != nullptr) {
                    indexing->prev->next = indexing->next;
                }

                if (indexing->next != nullptr) {
                    indexing->next->prev = indexing->prev;
                }

                if (duplicate == L.last) {
                    L.last = duplicate->prev;
                }

                indexing = indexing->next;
                delete duplicate;
            } else {
                indexing = indexing->next;
            }
        }
        current = current->next;
    }
}
```

- Menggunakan fungsi pada main

```
int main(){
    List L;
    newList(L);

    address mhs1 = newElement("Asep", 12345, "A", 100, 100);
    insertLast(L, mhs1);

    address mhs2 = newElement("Budi", 23456, "B", 90, 90);
    insertLast(L, mhs2);

    address mhs3 = newElement("Cika", 34567, "C", 80, 80);
    insertLast(L, mhs3);

    address mhs5 = newElement("Cika", 34567, "C", 80, 80);
    insertLast(L, mhs5);

    cout << "Jumlah Mahasiswa: " << length(L) << endl;

    cout << "Daftar Mahasiswa: \n";
    printList(L);

    removeDuplicates(L);
    cout << "\nJumlah Mahasiswa setelah menghapus duplikat: " << length(L) << endl;
    cout << "Daftar Mahasiswa setelah menghapus duplikat: \n";
    printList(L);

    cout << "\nMahasiswa dengan nilai tertinggi: \n";
    highest(L);
}
```