JOBSHEET 16

Collection

Nama: Rizqi Bagus Andrean

Absen: 25

Kelas: TI-1D

# Tujuan Praktikum

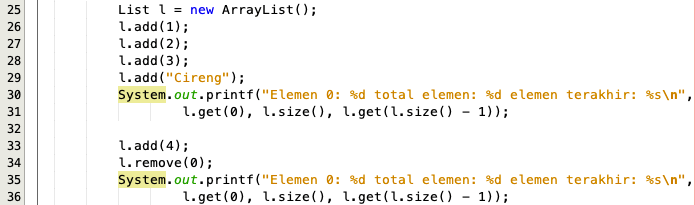
Setelah melakukan praktikum ini, mahasiswa mampu:

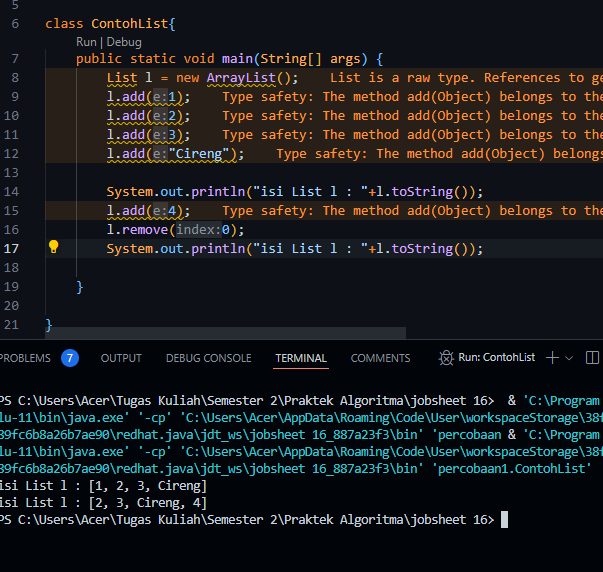
* + 1. memahami bentuk-bentuk collection dan hierarkinya;
    2. menerapkan collection sesuai dengan fungsi dan jenisnya;
    3. menyelesaikan kasus menggunakan collection yang sesuai.

# Kegiatan Praktikum 1

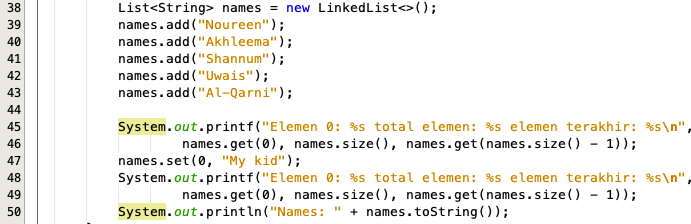
* + 1. **Percobaan 1**

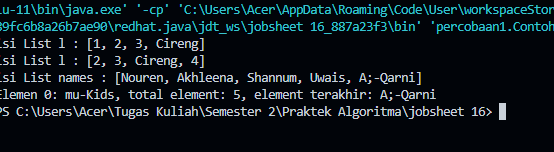
Pada percobaan 1 ini akan dicontohkan untuk menambahkan sebuah elemen, mengakses elemen, dan menghapus sebuah elemen.

* + - 1. Buatlah sebuah class ContohList yang main methode berisi kode program seperti di bawah ini



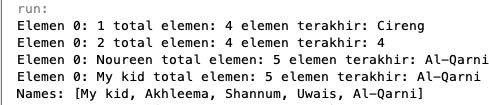
* + - 1. Tambahkan kode program untuk menggunakan collection dengan aturan penulisan kode program seperti berikut





## Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.

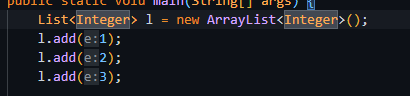


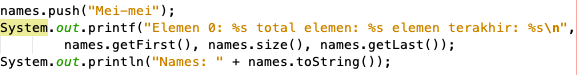
## Pertanyaan Percobaan

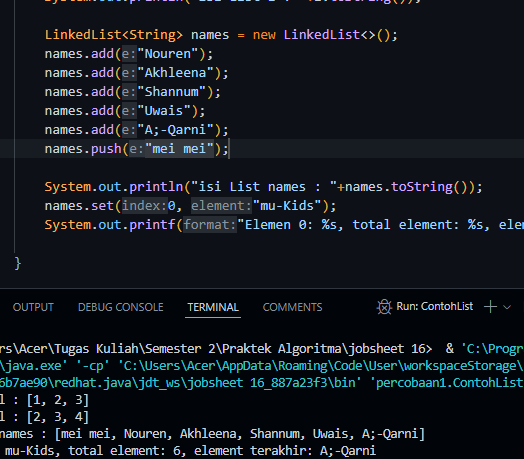
1. Perhatikan baris kode 25-36, mengapa semua jenis data bisa ditampung ke dalam sebuah Arraylist?

Karena di array list tidak diberi tipe data, jadi tipe datanya dynamic

1. Modifikasi baris kode 25-36 seingga data yang ditampung hanya satu jenis atau spesifik tipe tertentu!



1. Ubah kode pada baris kode 38 menjadi seperti ini
2. Tambahkan juga baris berikut ini, untuk memberikan perbedaan dari tampilan yang sebelumnya



1. Dari penambahan kode tersebut, silakan dijalankan dan apakah yang dapat Anda jelaskan!

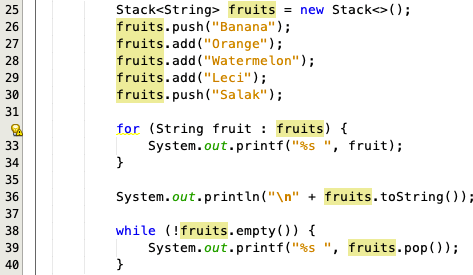
Mengubah tipe data names menjadi linked list, dan berakhir ada method push

# Kegiatan Praktikum 2

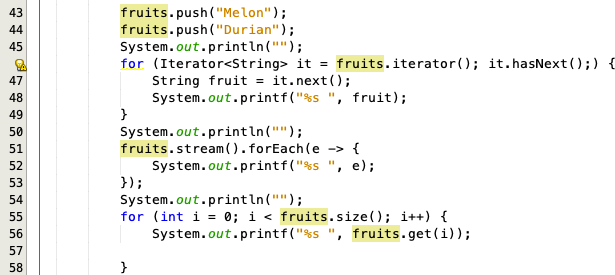
## Tahapan Percobaan

Pada praktikum 2 ini akan dibuat beberapa method untuk menampilkan beberapa cara yang dapat dilakukan untuk mengambil/menampilkan elemen pada sebuah collection. Silakan ikutilah Langkah-langkah di bawah ini

* + - 1. Buatlah class dengan nama LoopCollection serta tambahkan method main yang isinya adalah sebagai berikut**.**

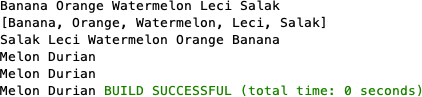


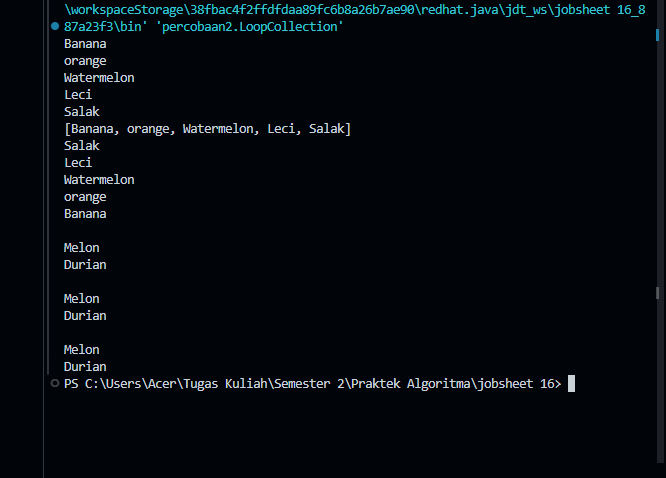
* + - 1. Tambahkan potongan kode berikut ini dari yang sebelumnya agar proses menampilkan elemen pada sebuah stack bervariasi.



## Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.





## Pertanyaan Percobaan

* + - 1. Apakah perbedaan fungsi push() dan add() pada objek *fruits*?

**push()**: Metode khusus dari kelas Stack untuk menambahkan elemen ke atas stack. Ini digunakan secara khusus untuk operasi pada stack.

**add()**: Metode umum dari antarmuka Collection yang digunakan untuk menambahkan elemen ke koleksi. Ini tidak spesifik untuk stack dan dapat digunakan untuk berbagai jenis koleksi.

* + - 1. Silakan hilangkan baris 43 dan 44, apakah yang akan terjadi? Mengapa bisa demikian?

Element melon dan durian hilang dan untuk looping dibawah method tersebut menampilkan list kosong, karena semua elemen telah dihapus senelumnya

* + - 1. Jelaskan fungsi dari baris 46-49?

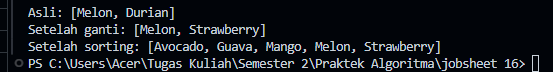
Sama aja kaya looping, tapi dia memeriksa apakah element setelah elemen saat ini ad, jika iya maka akan ditampilkan, jika tidak makan looping akn berhenti.

* + - 1. Silakan ganti baris kode 25, *Stack<String>* menjadi *List<String>* dan apakah yang terjadi? Mengapa bisa demikian?

Eror karena type stak tidak sama dengan List

* + - 1. Ganti elemen terakhir dari dari objek fruits menjadi “Strawberry”!
      2. Tambahkan 3 buah seperti “Mango”,”guava”, dan “avocado” kemudian dilakukan sorting!



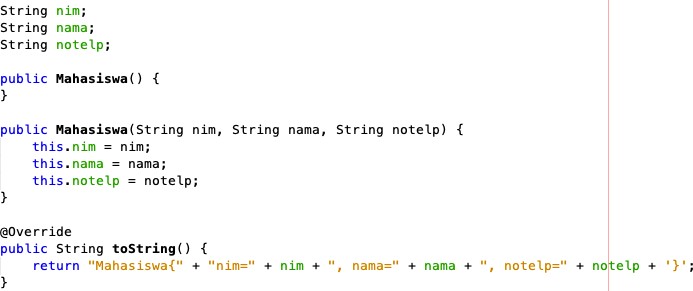


# Kegiatan Praktikum 3

## Tahapan Percobaan

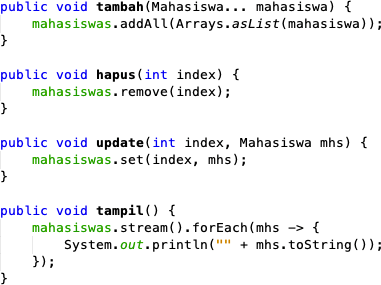
Pada praktikum 3 ini dilakukan uji coba untuk mengimplementasikan sebuah collection untuk menampung objek yang dibuat sesuai kebutuhan. Objek tersebut adalah sebuah objek mahasiswa dengan fungsi-fungsi umum seperti menambahkan, menghapus, mengubah, dan mencari.

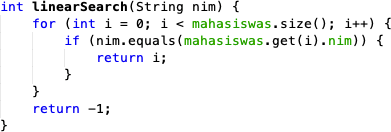
* + - 1. Buatlah sebuah class Mahasiswa dengan attribute, kontruktor, dan fungsi sebagai berikut.



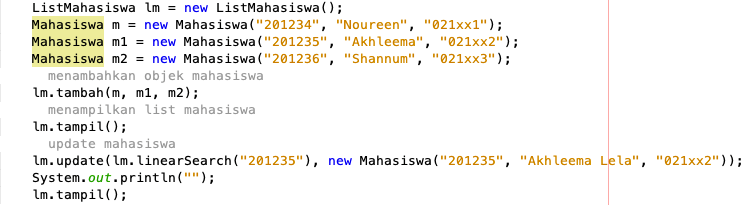
* + - 1. Selanjutnya, buatlah sebuah class ListMahasiswa yang memiliki attribute seperti di bawah ini

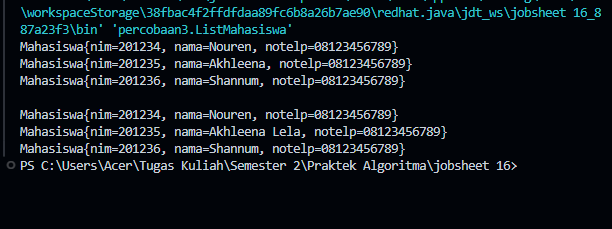


* + - 1. Method **tambah(), hapus(), update(), dan tampil()** secara berurut dibuat agar bisa melakukan operasi-operasi seperti yang telah disebutkan.
      2. Untuk proses hapus, update membutuhkan fungsi pencarian terlebih dahulu yang potongan kode programnya adalah sebagai berikut



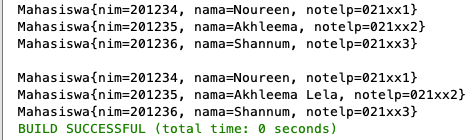
* + - 1. Pada class yang sama, tambahkan main method seperti potongan program berikut dan amati hasilnya!





## Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.



## Pertanyaan Percobaan

* + - 1. Pada fungsi tambah() yang menggunakan unlimited argument itu menggunakan konsep apa? Dan kelebihannya apa?

menggunakan konsep varargs dalam Java, memungkinkan penggunaan jumlah argumen yang tidak terbatas. Kelebihannya adalah memudahkan penambahan beberapa elemen sekaligus ke dalam koleksi.

* + - 1. Pada fungsi linearSearch() di atas, silakan diganti dengan fungsi binarySearch() dari collection!



* + - 1. Tambahkan fungsi sorting baik secara ascending ataupun descending pada class tersebut!

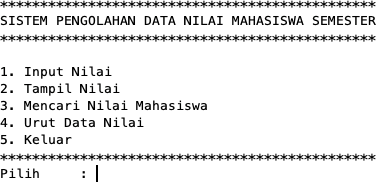
1. void sortAscending() {
2. Collections.sort(mahasiswas, Comparator.comparing(mhs -> mhs.nim));
3. }
4. void sortDescending() {
5. Collections.sort(mahasiswas, (mhs1, mhs2) -> mhs2.nim.compareTo(mhs1.nim));
6. }

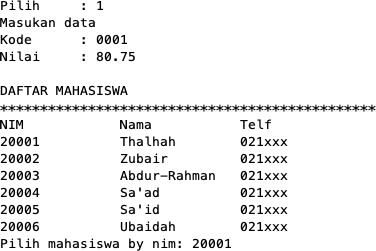
# Tugas Praktikum

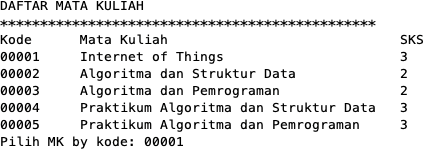
* + 1. Buatlah implementasi program daftar nilai mahasiswa semester, minimal memiliki 3 class yaitu Mahasiswa, Nilai, dan Mata Kuliah. Data Mahasiswa dan Mata Kuliah perlu melalui penginputan data terlebih dahulu.

## Ilustrasi Program

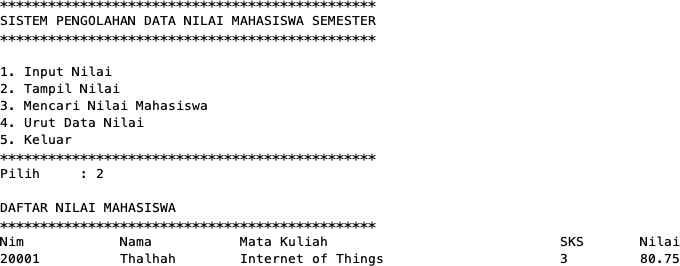
*Menu Awal dan Penambahan Data*

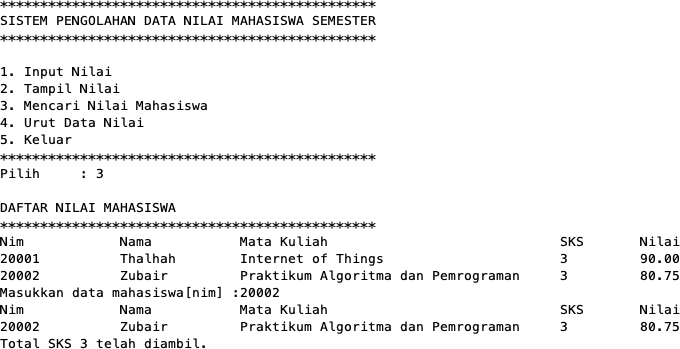


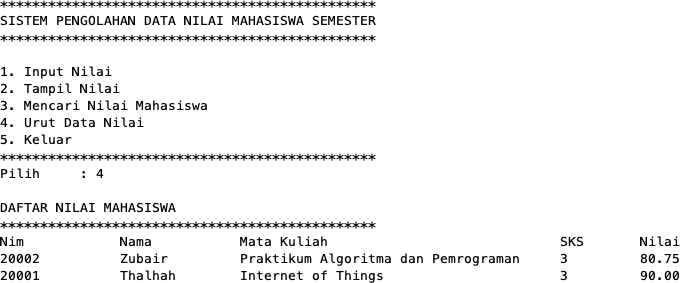




*Tampil Nilai*



*Pencarian Data Mahasiswa*

*Pengurutan Data Nilai*

* + 1. Tambahkan prosedur hapus data mahasiswa melalui implementasi Queue pada collections Tugas nomor 1!

--- \*\*\* ---

Matakuliah.java

package tugas;

public class MataKuliah {

    String nama, kode;

    double nilai;

    int sks;

    public MataKuliah(String nama, String kode, double nilai, int sks){

        this.nama = nama;

        this.kode = kode;

        this.nilai = nilai;

        this.sks = sks;

    }

    public MataKuliah(String nama, String kode, int sks){

        this.nama = nama;

        this.kode = kode;

        this.sks = sks;

    }

}

Mahasiswa.java

package tugas;

import java.util.ArrayList;

public class Mahasiswa {

    String nim, nama, notelp;

    ArrayList<MataKuliah> mataKuliahs = new ArrayList<>();

    public Mahasiswa(String nim, String nama, String notelp){

        this.nim = nim;

        this.nama = nama;

        this.notelp = notelp;

    }

    @Override

    public String toString() {

        return "Mahasiswa{" + "nim=" + nim + ", nama=" + nama + ", notelp=" + notelp + '}';

    }

}

ListMahasiwa.java

package tugas;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collections;

import java.util.Comparator;

import java.util.HashMap;

import java.util.LinkedList;

import java.util.Queue;

import java.util.Scanner;

public class ListMahasiswa {

    ArrayList<Mahasiswa> mahasiswas = new ArrayList<>();

    ArrayList<MataKuliah> mataKuliahs = new ArrayList<>();

    private static Scanner scanner = new Scanner(System.in);

    Queue<Integer> queue = new LinkedList<>();

    void tambah(Mahasiswa... mahasiswa) {

        mahasiswas.addAll(Arrays.asList(mahasiswa));

    }

    void hapus(int index) {

        mahasiswas.remove(index);

    }

    // Prosedur hapus menggunakan Queue

    void hapusMahasiswaDenganQueue() {

        System.out.print("Masukkan NIM mahasiswa yang ingin dihapus: ");

        String nim = scanner.nextLine();

        int index = linearSearch(nim);

        if (index == -1) {

            System.out.println("Mahasiswa tidak ditemukan");

            return;

        }

        queue.add(index);

        hapus(queue.poll());

        System.out.println("Mahasiswa dengan NIM " + nim + " telah dihapus.");

    }

    void update(int index, Mahasiswa mahasiswa) {

        mahasiswas.set(index, mahasiswa);

    }

    void tampil() {

        mahasiswas.stream().forEach((mhs) -> {

            System.out.println(mhs);

        });

    }

    int linearSearch(String nim) {

        for (int i = 0; i < mahasiswas.size(); i++) {

            if (nim.equals(mahasiswas.get(i).nim)) {

                return i;

            }

        }

        return -1;

    }

    int binarySearch(String nim) {

        // Urutkan dulu array list

        Collections.sort(mahasiswas, (mhs1, mhs2) -> mhs1.nim.compareTo(mhs2.nim));

        // Lakukan binary search

        int index = Collections.binarySearch(mahasiswas, new Mahasiswa(nim, "", ""),

                                             (mhs1, mhs2) -> mhs1.nim.compareTo(mhs2.nim));

        return index >= 0 ? index : -1;

    }

    MataKuliah searchByKode(String kode) {

        for (MataKuliah mk : mataKuliahs) {

            if (mk.kode.equals(kode)) {

                return mk;

            }

        }

        return null;

    }

    void sortAscending() {

        Collections.sort(mahasiswas, Comparator.comparing(mhs -> mhs.nim));

    }

    void sortDescending() {

        Collections.sort(mahasiswas, (mhs1, mhs2) -> mhs2.nim.compareTo(mhs1.nim));

    }

    void menu() {

        System.out.println("Menu: ");

        System.out.println("1. Input Nilai");

        System.out.println("2. Tampil Nilai");

        System.out.println("3. Mencari Nilai Mahasiswa");

        System.out.println("4. Urut Data Nilai");

        System.out.println("5. Hapus Mahasiswa");

        System.out.println("6. Exit");

    }

    void inputNilai() {

        System.out.print("Masukkan NIM: ");

        String nim = scanner.nextLine();

        int index = linearSearch(nim);

        if (index == -1) {

            System.out.println("Mahasiswa tidak ditemukan");

            return;

        }

        Mahasiswa m = mahasiswas.get(index);

        System.out.print("Masukkan Kode Mk: ");

        String kode = scanner.nextLine();

        System.out.print("Masukkan Nilai: ");

        double nilai = scanner.nextDouble();

        scanner.nextLine();

        // Cari mata kuliah

        MataKuliah mk = searchByKode(kode);

        if (mk == null) {

            System.out.println("Mata kuliah tidak ditemukan");

            return;

        }

        mk.nilai = nilai;

        m.mataKuliahs.add(mk);

        System.out.println("Nilai berhasil dimasukkan ke matakuliah: " + mk.nama);

    }

    void tampilSemuaNilaiMahasiswa() {

        mahasiswas.stream().forEach((mhs) -> {

            System.out.println("NIM: " + mhs.nim + ", Nama: " + mhs.nama);

            mhs.mataKuliahs.stream().forEach((mk) -> {

                System.out.println("- Mata Kuliah: " + mk.nama + ", Nilai: " + mk.nilai);

            });

            System.out.println();

        });

    }

    void cariNilaiByNim() {

        System.out.print("Masukkan nim: ");

        String nim = scanner.nextLine();

        int index = linearSearch(nim);

        if (index == -1) {

            System.out.println("Mahasiswa tidak ditemukan");

            return;

        }

        Mahasiswa m = mahasiswas.get(index);

        System.out.println("NIM: " + m.nim + ", Nama: " + m.nama);

        m.mataKuliahs.stream().forEach((mk) -> {

            System.out.println("- Mata Kuliah: " + mk.nama + ", Nilai: " + mk.nilai);

        });

        System.out.println();

    }

    void urutNilaiByNiilaiTerbesar() {

        mahasiswas.stream().forEach((mhs) -> {

            mhs.mataKuliahs.sort((mk1, mk2) -> Double.compare(mk2.nilai, mk1.nilai));

        });

    }

    public static void main(String[] args) {

        ListMahasiswa lm = new ListMahasiswa();

        // isi mahasiswa

        Mahasiswa m = new Mahasiswa("201234", "Nouren", "08123456789");

        Mahasiswa m1 = new Mahasiswa("201235", "Akhleena", "08123456789");

        Mahasiswa m2 = new Mahasiswa("201236", "Shannum", "08123456789");

        lm.tambah(m, m1, m2);

        lm.tambah(

            new Mahasiswa("201237", "Uwais", "08123456789"),

            new Mahasiswa("201238", "Al-Qarni", "08123456789"),

            new Mahasiswa("201239", "mei mei", "08123456789"));

        // isi list mk

        lm.mataKuliahs.add(new MataKuliah("Internet Of Things", "00001", 3));

        lm.mataKuliahs.add(new MataKuliah("Algoritma Struktur data", "00002", 2));

        lm.mataKuliahs.add(new MataKuliah("Algoritma dan Pemrograman", "00003", 2));

        lm.mataKuliahs.add(new MataKuliah("Praktikum Algoritma dan Struktur Data", "00004", 3));

        lm.mataKuliahs.add(new MataKuliah("Praktikum Algoritma dan Pemrograman", "00005", 3));

        boolean isExit = false;

        while (!isExit) {

            lm.menu();

            System.out.print("Pilih menu: ");

            int menu = scanner.nextInt();

            scanner.nextLine();

            switch (menu) {

                case 1:

                    lm.inputNilai();

                    break;

                case 2:

                    lm.tampilSemuaNilaiMahasiswa();

                    break;

                case 3:

                    lm.cariNilaiByNim();

                    break;

                case 4:

                    lm.urutNilaiByNiilaiTerbesar();

                    break;

                case 5:

                    lm.hapusMahasiswaDenganQueue();

                    break;

                case 6:

                    isExit = true;

                    break;

                default:

                    System.out.println("Pilihan tidak valid");

                    break;

            }

        }

        scanner.close();

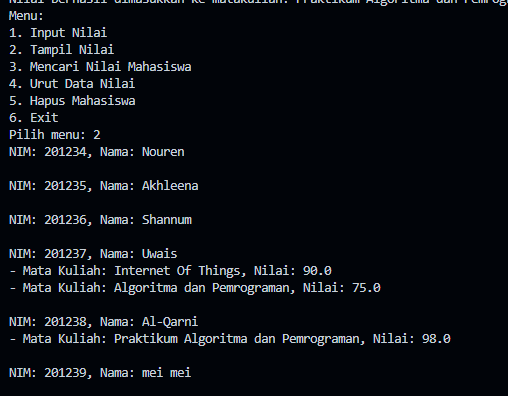
    }

}

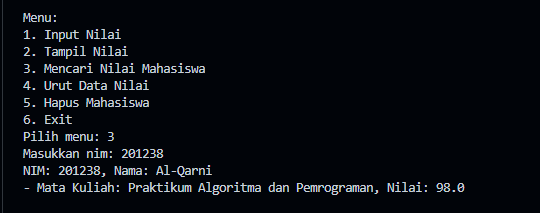
Tambah Nilai



Tampil Nilai



Tampil nilai by nim



Hapus Mahasiswa

