LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA
JOBSHEET 11

VANESA MARDIANA PUTRI

244107020129

KELAS TI-1B

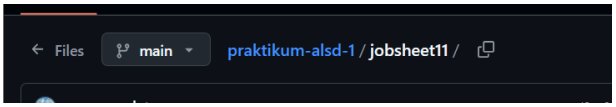
PRODI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

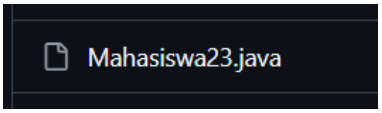
POLITEKNIK NEGERI MALANG

A. PERCOBAAN 1

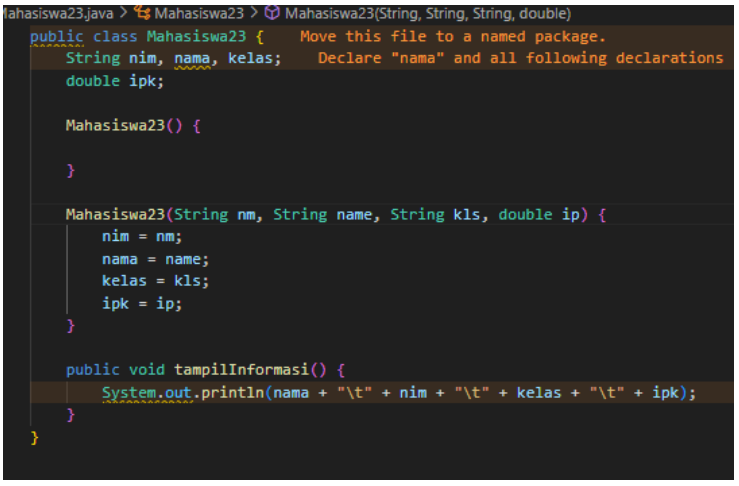
- Langkah - langkah :
    1. Buat folder atau package baru bernama Jobsheet11 di dalam repository Praktikum ASD.



    2. Tambahkan class-class berikut:

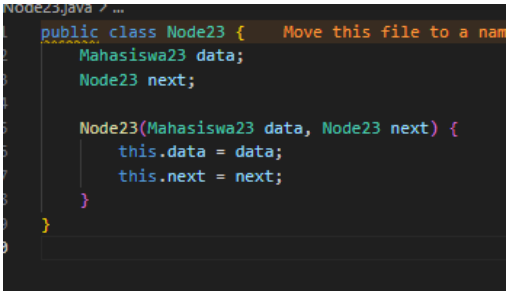


    3. Implementasikan Class Mahasiswa00 sesuai dengan diagram class berikut ini :



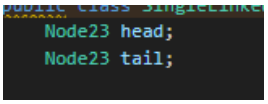
```
Mahasiswa23.java > Mahasiswa23 > Mahasiswa23(String, String, String, double)
public class Mahasiswa23 {    Move this file to a named package.
    String nim, nama, kelas;    Declare "nama" and all following declarations
    double ipk;

    Mahasiswa23() {

    }

    Mahasiswa23(String nm, String name, String kls, double ip) {
        nim = nm;
        nama = name;
        kelas = kls;
        ipk = ip;
    }

    public void tampilInformasi() {
        System.out.println(nama + "\t" + nim + "\t" + kelas + "\t" + ipk);
    }
}
```

    4. Implementasi class Node seperti gambar berikut ini



```
Node23.java > ...
public class Node23 {    Move this file to a nam
    Mahasiswa23 data;
    Node23 next;

    Node23(Mahasiswa23 data, Node23 next) {
        this.data = data;
        this.next = next;
    }
}
```

    5. Tambahkan attribute head dan tail pada class SingleLinkedList.



```
    Node23 head;
    Node23 tail;
```

6. Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada SingleLinkedList.
7. Tambahkan method isEmpty().

```
public boolean isEmpty() {
    return (head == null);
}
```

8. Implementasi method untuk mencetak dengan menggunakan proses traverse.

```
public void print() {
    if (!isEmpty()) {
        Node23 tmp = head;
        System.out.println(x:"Isi Linked List:\t");    Re
        while (tmp != null) {
            tmp.data.tampilInformasi();
            tmp = tmp.next;
        }
        System.out.println(x:"");    Replace this use of
    } else {
        System.out.println(x:"Linked List Kosong!");
    }
}
```

9. Implementasikan method addFirst().

```
public void addFirst(Mahasiswa23 input) {
    Node23 ndInput = new Node23(input, next:null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else {
        ndInput.next = head;
        head = ndInput;
    }
}
```

10. Implementasikan method addLast().

```
public void addLast(Mahasiswa23 input) {
    Node23 ndInput = new Node23(input, next:null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else {
        tail.next = ndInput;
        tail = ndInput;
    }
}
```

11. Implementasikan method insertAfter, untuk memasukkan node yang memiliki data input setelah node yang memiliki data key.

```java
public void insertAfter(String key, Mahasiswa23 input) {
    Node23 ndInput = new Node23(input, next:null);
    Node23 temp = head;
    do {
        if (temp.data.nama.equalsIgnoreCase(key)) {
            ndInput.next = temp.next;
            temp.next = ndInput;
            if (ndInput.next == null) {
                tail = ndInput;
            }
            break;
        }
        temp = temp.next;
    } while (temp != null);
}
```

12. Tambahkan method penambahan node pada indeks tertentu.

```java
public void insertAt(int index, Mahasiswa23 input) {
    if (index < 0) {
        System.out.println(x:"Index salah");    Replace this use of S
    } else if (index == 0) {
        addFirst(input);
    } else {
        Node23 temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new Node23(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}
```

13. Pada class SLLMain00, buatlah fungsi main, kemudian buat object dari class SingleLinkedList.

```java
import java.util.Scanner;    Move this file to a named package.

public class SLLMain23 {
    Run | Debug
    public static void main(String[] args) {
        SingleLinkedList23 sll = new SingleLinkedList23();
```

14. Buat empat object mahasiswa dengan nama mhs1, mhs2, mhs3, mhs4 kemudian isi data setiap object melalui konstruktor.

```java
SingleLinkedList23 sll = new SingleLinkedList23();
Mahasiswa23 mhs1 = new Mahasiswa23(nm:"24212200",name:"Alvaro", kls:"1A", ip:4.0);
Mahasiswa23 mhs2 = new Mahasiswa23(nm:"23212201",name:"Bimon", kls:"2B", ip:3.8);
Mahasiswa23 mhs3 = new Mahasiswa23(nm:"22212202",name:"Cintia", kls:"3C", ip:3.5);
Mahasiswa23 mhs4 = new Mahasiswa23(nm:"21212203",name:"Dirga", kls:"4D", ip:3.6);
```

15. Tambahkan Method penambahan data dan pencetakan data di setiap penambahannya agar terlihat perubahannya.

```java
sll.print();
sll.addFirst(mhs4);
sll.print();
sll.addLast(mhs1);
sll.print();
sll.insertAfter(key:"Dirga", mhs3);
sll.insertAt(index:2, mhs2);
sll.print();
System.out.println();    Replace this
```

16. Run program.

```
Linked List Kosong!
Isi Linked List:
Dirga   21212203    4D  3.6

Isi Linked List:
Dirga   21212203    4D  3.6
Alvaro  24212200    1A  4.0

Isi Linked List:
Dirga   21212203    4D  3.6
Cintia  22212202    3C  3.5
Bimon   23212201    2B  3.8
Alvaro  24212200    1A  4.0
```

- Jawaban Pertanyaan :
    1. Program menampilkan "Linked List Kosong" karena belum ada data (node) yang dimasukkan ke dalam linked list, sehingga pointer head masih null.
    2. Temp(temporary) adalah variabel bantu sementara yang sering digunakan untuk menelusuri (traverse) atau membantu proses seperti menambah, mencari, menghapus, atau menampilkan data pada linked list.
    3.

```java
Scanner sc = new Scanner(System.in);    Resource leak: 'sc' is never closed
String jawab = "";
do {
    System.out.println(x:"Seluruh Data: ");    Replace this use of System.out by a logger.
    sll.print();
    System.out.print(s:"Apakah ingin menambah data baru?: ");    Replace this use of System.out by a logger.
    jawab = sc.nextLine();

    System.out.print(s:"Masukkan Nama Mahasiswa : ");    Replace this use of System.out by a logger.
    String nama = sc.nextLine();
    System.out.print(s:"Masukkan NIM Mahasiswa : ");    Replace this use of System.out by a logger.
    String nim = sc.nextLine();
    System.out.print(s:"Masukkan Kelas Mahasiswa : ");    Replace this use of System.out by a logger.
    String kelas = sc.nextLine();
    System.out.print(s:"Masukkan IPK Mahasiswa : ");    Replace this use of System.out by a logger.
    double ipk = sc.nextDouble();

    Mahasiswa23 mhs = new Mahasiswa23(nim, nama, kelas, ipk);

    System.out.println();    Replace this use of System.out by a logger.
    System.out.println(x:"Silahkan pilih menu: ");    Replace this use of System.out by a logger.
    System.out.println(x:"1. Menambahkan data Diawal");    Replace this use of System.out by a logger.
    System.out.println(x:"2. Menambahkan data Setelah Nama Tertentu");    Replace this use of System.out by a logger.
    System.out.println(x:"3. Menambahkan data di Index tertentu");    Replace this use of System.out by a logger.
    System.out.println(x:"4. Menambahkan data di Akhir");    Replace this use of System.out by a logger.
    System.out.print(s:"Masukkan Pilihanmu : ");    Replace this use of System.out by a logger.
    int jwbnMenu = sc.nextInt();
    sc.nextLine();
    switch (jwbnMenu) {
        case 1:
            sll.addFirst(mhs);
            sll.print();
            break;
        case 2:
            System.out.print(s:"Masukkan Nama yang ingin di cari : ");    Replace this use of System.out by a logger.
            String dicari = sc.nextLine();
            sll.insertAfter(dicari, mhs);
            sll.print();
            break;
        case 3:
            System.out.print(s:"Masukkan Index tempat : ");    Replace this use of System.out by a logger.
            int index = sc.nextInt();
            sc.nextLine();
            sll.insertAt(index, mhs);
            sll.print();
            break;
        case 4:
            sll.addLast(mhs);
            sll.print();
            break;
        default:
            break;
    }
} while (jawab.equalsIgnoreCase(anotherString:"ya") || jawab.equalsIgnoreCase(anotherString:"y"));
}
```

B. PERCOBAAN 2

- Langkah - langkah :

1. Implementasikan method untuk mengakses data dan indeks pada linked list.
2. Tambahkan method untuk mendapatkan data pada indeks tertentu pada class Single Linked List.

```
public void getData(int index) {
    Node23 tmp = head;
    for (int i = 0; i < index; i++) {
        tmp = tmp.next;
    }
    tmp.data.tampilInformasi();
}
```

3. Implementasikan method indexOf.

```
public int indexOf(String key) {
    Node23 tmp = head;
    int index = 0;
    while (tmp != null && !tmp.data.nama.equalsIgnoreCase(key)) {
        tmp = tmp.next;
        index++;
    }
    if (tmp == null) {
        return -1;
    } else {
        return index;
    }
}
```

4. Tambahkan method removeFirst pada class SingleLinkedList.

```
public void removeFirst() {
    if (isEmpty()) {
        System.out.println(x:"Linked List masih kosong, tidak dapat dihapus!");    Define
    } else if (head == tail) {
        head = tail = null;
    } else {
        head = head.next;
    }
}
```

5. Tambahkan method untuk menghapus data pada bagian belakang pada class SingleLinkedList.

```
public void removeLast() {
    if (isEmpty()) {
        System.out.println(x:"Linked List masih kosong, tidak dapat dihapus!");    Repl
    } else if (head == tail) {
        head = tail = null;
    } else {
        Node23 tmp = head;
        while (tmp.next != tail) {
            tmp = tmp.next;
        }
        tmp.next = null;
        tail = tmp;
    }
}
```

6. Sebagai langkah berikutnya, akan diimplementasikan method remove.

```
public void removeBy(String key) {
    if (isEmpty()) {
        System.out.println(x:"Linked List masih kosong, tidak dapat dihapus!");    Replac
    } else {
        Node23 tmp = head;
        while (tmp != null) {    Reduce the total number of break and continue statements
            if (tmp.data.nama.equalsIgnoreCase(key) && tmp == head) {
                this.removeFirst();
                break;
            } else if (tmp.data.nama.equalsIgnoreCase(key)) {
                tmp.next = tmp.next.next;
                if (tmp.next == null) {
                    tail = tmp;
                }
                break;
            }
            tmp = tmp.next;
        }
    }
}
```

7. Implementasi method untuk menghapus node dengan menggunakan index.

```
    public void removeAt(int index) {
        if (index == 0) {
            removeFirst();
        } else {
            Node23 tmp = head;
            for (int i = 0; i < index - 1; i++) {
                tmp = tmp.next;
            }
            tmp.next = tmp.next.next;
            if (tmp.next == null) {
                tail = tmp;
            }
        }
    }

}
```

8. Kemudian, coba lakukan pengaksesan dan penghapusan data di method main pada class SLLMain dengan menambahkan kode berikut:

```
System.out.println(x:"data index 1 : ");    Replace this use of System.out by a logger.
sll.getData(index:1);

System.out.println("data mahasiswa an Bimon berada pada index : " + sll.indexOf(key:"Bimon"));
System.out.println();    Replace this use of System.out by a logger.

sll.removeFirst();
sll.removeLast();
sll.print();
sll.removeAt(index:0);
sll.print();
```

9. Jalankan class SLLMain.

```
Isi Linked List:
Dirga   21212203        4D      3.6
Alvaro  24212200        1A      4.0

Isi Linked List:
Dirga   21212203        4D      3.6
Cintia  22212202        3C      3.5
Bimon   23212201        2B      3.8
Alvaro  24212200        1A      4.0


data index 1 :
Cintia  22212202        3C      3.5
data mahasiswa an Bimon berada pada index : 2

Isi Linked List:
Cintia  22212202        3C      3.5
Bimon   23212201        2B      3.8

Isi Linked List:
Bimon   23212201        2B      3.8

Seluruh Data:
Isi Linked List:
Bimon   23212201        2B      3.8
```

- Jawaban Pertanyaan :
    1. Keyword break digunakan untuk menghentikan perulangan (looping) saat kondisi tertentu telah terpenuhi.
    2. Kode ini biasanya digunakan ketika kita sedang menghapus node di tengah atau akhir dari single linked list kemudian kode akan memastikan pointer tail tetap menunjuk ke node terakhir setelah penghapusan.

C. TUGAS

1. Main

```java
import java.util.Scanner;    Move this file to a named package.

public class AntrianMain {
    Run | Debug
    public static void main(String[] args) {    A "Brain Method" was detected. Refactor it to reduce at lea
    Scanner scanner = new Scanner(System.in);
    System.out.print(s:"Masukkan kapasitas maksimal antrian: ");    Replace this use of System.out by a log
    int kapasitas = scanner.nextInt();
    scanner.nextLine();
    Antrian antrian = new Antrian(kapasitas);

    int pilihan;
    do {
      System.out.println(x:"\n--- MENU ANTRIAN MAHASISWA ---");    Replace this use of System.out by a logg
      System.out.println(x:"1. Tambah Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"2. Proses Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"3. Cetak Seluruh Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"4. Tampilkan Mahasiswa Terdepan");    Replace this use of System.out by a logge
      System.out.println(x:"5. Tampilkan Mahasiswa Terakhir");    Replace this use of System.out by a logge
      System.out.println(x:"6. Cek Antrian Kosong");    Replace this use of System.out by a logger.
      System.out.println(x:"7. Cek Antrian Penuh");    Replace this use of System.out by a logger.
      System.out.println(x:"8. Lihat Jumlah Mahasiswa dalam Antrian");    Replace this use of System.out by
      System.out.println(x:"9. Kosongkan Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"10. Keluar");    Replace this use of System.out by a logger.
      System.out.print(s:"Pilih: ");    Replace this use of System.out by a logger.
      pilihan = scanner.nextInt();
      scanner.nextLine();

      switch (pilihan) {
        case 1:
          if (antrian.isFull()) {
            System.out.println(x:"Antrian penuh, tidak bisa menambah.");    Replace this use of System.out
            break;
          }
          System.out.print(s:"Nama: ");    Replace this use of System.out by a logger.
          String nama = scanner.nextLine();
          System.out.print(s:"NIM: ");    Replace this use of System.out by a logger.
          String nim = scanner.nextLine();
          System.out.print(s:"Kelas: ");    Replace this use of System.out by a logger.
          String kelas = scanner.nextLine();
          MahasiswaAntri mhs = new MahasiswaAntri(nama, nim, kelas);
          antrian.insert(mhs);
          break;
```


```java
import java.util.Scanner;    Move this file to a named package.

public class AntrianMain {
    Run | Debug
    public static void main(String[] args) {    A "Brain Method" was detected. Refactor it to reduce at lea
    Scanner scanner = new Scanner(System.in);
    System.out.print(s:"Masukkan kapasitas maksimal antrian: ");    Replace this use of System.out by a log
    int kapasitas = scanner.nextInt();
    scanner.nextLine();
    Antrian antrian = new Antrian(kapasitas);

    int pilihan;
    do {
      System.out.println(x:"\n--- MENU ANTRIAN MAHASISWA ---");    Replace this use of System.out by a logg
      System.out.println(x:"1. Tambah Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"2. Proses Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"3. Cetak Seluruh Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"4. Tampilkan Mahasiswa Terdepan");    Replace this use of System.out by a logge
      System.out.println(x:"5. Tampilkan Mahasiswa Terakhir");    Replace this use of System.out by a logge
      System.out.println(x:"6. Cek Antrian Kosong");    Replace this use of System.out by a logger.
      System.out.println(x:"7. Cek Antrian Penuh");    Replace this use of System.out by a logger.
      System.out.println(x:"8. Lihat Jumlah Mahasiswa dalam Antrian");    Replace this use of System.out by
      System.out.println(x:"9. Kosongkan Antrian");    Replace this use of System.out by a logger.
      System.out.println(x:"10. Keluar");    Replace this use of System.out by a logger.
      System.out.print(s:"Pilih: ");    Replace this use of System.out by a logger.
      pilihan = scanner.nextInt();
      scanner.nextLine();

      switch (pilihan) {
        case 1:
          if (antrian.isFull()) {
            System.out.println(x:"Antrian penuh, tidak bisa menambah.");    Replace this use of System.out
            break;
          }
          System.out.print(s:"Nama: ");    Replace this use of System.out by a logger.
          String nama = scanner.nextLine();
          System.out.print(s:"NIM: ");    Replace this use of System.out by a logger.
          String nim = scanner.nextLine();
          System.out.print(s:"Kelas: ");    Replace this use of System.out by a logger.
          String kelas = scanner.nextLine();
          MahasiswaAntri mhs = new MahasiswaAntri(nama, nim, kelas);
          antrian.insert(mhs);
          break;
```


2. Antrian.java

```java
public class Antrian {    Move this file to a named package.
    NodeMahasiswa head;
    NodeMahasiswa tail;
    int size;
    int currentSize;

  public Antrian(int size) {
    head = null;
    tail = null;
    this.size = size;
    this.currentSize = 0;
  }

  public boolean isEmpty() {
    return head == null;
  }

  public boolean isFull() {
    return currentSize == size;
  }

  public int getCurrentSize() {
    return currentSize;
  }

  public void clear() {
    head = null;
    tail = null;
    currentSize = 0;
    System.out.println(x:"Antrian Telah Dikosongkan");    Replace this use of System.out by a lo
  }

  public void print() {
    if (isEmpty()) {
      System.out.println(x:"Antrian Kosong");    Replace this use of System.out by a logger.
      return;
    }
    NodeMahasiswa temp = head;
    System.out.println(x:"Mahasiswa Mengantri : ");    Replace this use of System.out by a logge
    while (temp != null) {
      temp.data.tampilInformasi();
      temp = temp.next;
    }
  }
```

```java
  public void insert(MahasiswaAntri data) {
    if (isFull()) {
      System.out.println(x:"Antrian Sudah Penuh");    Replace this use of System.out by a log
      return;
    }
    NodeMahasiswa temp = new NodeMahasiswa(data, next:null);
    if (head == null) {
      head = temp;
      tail = temp;
    } else {
      tail.next = temp;
      tail = temp;
    }
    currentSize++;
  }

  public void pop() {
    if (isEmpty()) {
      System.out.println(x:"Antrian Masih Kosong");    Define a constant instead of duplicati
    } else {
      System.out.println(x:"Memproses 1 Antrian Mahasiswa");    Replace this use of System.ou
      head.data.tampilInformasi();
      head = head.next;
      currentSize--;
    }
  }

  public void tampilTerdepan() {
    if (isEmpty()) {
      System.out.println(x:"Antrian Masih Kosong");    Replace this use of System.out by a lo
    } else {
      System.out.println(x:"Mahasiswa Terdepan : ");    Replace this use of System.out by a l
      head.data.tampilInformasi();
    }
  }

  public void tampilTerakhir() {
    if (isEmpty()) {
      System.out.println(x:"Antrian Masih Kosong");    Replace this use of System.out by a lo
    } else {
      System.out.println(x:"Mahasiswa Terakhir : ");    Replace this use of System.out by a l
      tail.data.tampilInformasi();
    }
  }
}
```

3. **NodeMahasiswa.java**

```java
public class NodeMahasiswa {    Move this file to a named package.
    MahasiswaAntri data;
    NodeMahasiswa next;

  public NodeMahasiswa() {}

  public NodeMahasiswa(MahasiswaAntri data, NodeMahasiswa next) {
    this.data = data;
    this.next = next;
  }
}
```

4. **MahasiswaAntri.java**

```java
public class MahasiswaAntri {    Move this file to a named package.
    String nama;
    String nim;
    String kelas;

  public MahasiswaAntri(String nama, String nim, String kelas) {
    this.nama = nama;
    this.nim = nim;
    this.kelas = kelas;
  }

  public void tampilInformasi() {
    System.out.println(nama + "\t" + nim + "\t" + kelas);    Replace this us
  }
}
```

5. **Test.java**

```java
public class Test {    Move this file to a named package.
    Run | Debug
    public static void main(String[] args) {
    System.out.println(x:"=== TEST ANTRIAN ===");    Replace this use of System.out by a logger.

    Antrian antrian = new Antrian(size:3);

    System.out.println(x:"\n[TEST 1] Tambah Mahasiswa ke Antrian");    Replace this use of System.out by a lo
    antrian.insert(new MahasiswaAntri(nama:"Ani", nim:"123", kelas:"1B"));
    antrian.insert(new MahasiswaAntri(nama:"Budi", nim:"124", kelas:"2A"));
    antrian.print();

    System.out.println(x:"\n[TEST 2] Tampilkan Mahasiswa Terdepan dan Terakhir");    Replace this use of Syst
    antrian.tampilTerdepan();
    antrian.tampilTerakhir();

    System.out.println(x:"\n[TEST 3] Cek apakah antrian penuh?");    Replace this use of System.out by a logg
    System.out.println(antrian.isFull() ? "Antrian penuh" : "Antrian belum penuh");    Replace this use of Sy

    System.out.println(x:"\n[TEST 4] Menambahkan mahasiswa untuk memenuhi antrian");    Replace this use of S
    antrian.insert(new MahasiswaAntri(nama:"Citra", nim:"125", kelas:"2A"));
    antrian.print();
    System.out.println("Apakah antrian penuh? " + (antrian.isFull() ? "Ya" : "Tidak"));    Replace this use o

    System.out.println(x:"\n[TEST 5] Tambah Mahasiswa saat penuh");    Replace this use of System.out by a lo
    antrian.insert(new MahasiswaAntri(nama:"Dina", nim:"126", kelas:"1D"));

    System.out.println(x:"\n[TEST 6] Proses (pop) Mahasiswa dari Antrian");    Replace this use of System.out
    antrian.pop();
    antrian.print();

    System.out.println(x:"\n[TEST 7] Tampilkan Mahasiswa Terdepan setelah pop");    Replace this use of Syste
    antrian.tampilTerdepan();

    System.out.println(x:"\n[TEST 8] Cek ukuran antrian saat ini");    Replace this use of System.out by a lo
    System.out.println("Jumlah: " + antrian.getCurrentSize());    Replace this use of System.out by a logger.

    System.out.println(x:"\n[TEST 9] Kosongkan antrian");    Replace this use of System.out by a logger.
    antrian.clear();

    System.out.println(x:"\n[TEST 10] Cek apakah antrian kosong?");    Replace this use of System.out by a lo
    System.out.println(antrian.isEmpty() ? "Antrian kosong" : "Antrian tidak kosong");    Replace this use of

    System.out.println(x:"\n[TEST 11] Cetak antrian yang sudah dikosongkan");    Replace this use of System.o
    antrian.print();
  }
}
```

6. **Run Program**


Masukkan kapasitas maksimal antrian: 5
--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 1
Nama: Vanesa
NIM: 2345
Kelas: 1B
--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 1
Nama: Nina
NIM: 2344
Kelas: 1C
--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 1
Nama: Adit
NIM: 2346
Kelas: 1C

```
--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 1
Nama: Dea
NIM: 2347
Kelas: 1A

--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 1
Nama: Vanya
NIM: 2341
Kelas: 1B

--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 2
Memproses 1 Antrian Mahasiswa
Vanesa  2345    1B
```

```
--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 3
Mahasiswa Mengantri :
Nina    2344    1C
Adit    2346    1C
Dea     2347    1A
Vanya   2341    1B

--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 4
Mahasiswa Terdepan :
Nina    2344    1C

--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 5
Mahasiswa Terakhir :
Vanya   2341    1B
```

```
--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 6
Antrian tidak kosong.

--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 7
Masih ada ruang di antrian.

--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 8
Jumlah mahasiswa dalam antrian: 4
```

```
--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 9
Antrian Telah Dikosongkan

--- MENU ANTRIAN MAHASISWA ---
1. Tambah Antrian
2. Proses Antrian
3. Cetak Seluruh Antrian
4. Tampilkan Mahasiswa Terdepan
5. Tampilkan Mahasiswa Terakhir
6. Cek Antrian Kosong
7. Cek Antrian Penuh
8. Lihat Jumlah Mahasiswa dalam Antrian
9. Kosongkan Antrian
10. Keluar
Pilih: 10
Terima kasih!
```