LAPORAN HASIL PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

Jobsheet 11

Adi Luhung

244107020088

Kelas 1E

Program Studi Teknik Informatika

Jurusan Teknologi Informasi

Politeknik Negeri Malang

2025

1. Praktikum

1.1 Percobaan 1

1. Membuat class Mahasiswa01 yang berisi atribut, konstruktor dan method tampilInformasi().

```
package Jobsheet11;
public class Mahasiswa01 {
    String nim;
    String nama;
    String kelas;
    double ipk;

    Mahasiswa01() {
    }
    Mahasiswa01(String nm, String name, String kls, double ip) {
        nim = nm;
        nama = name;
        kelas = kls;
        ipk = ip;
    }
    void tampilInformasi() {
        System.out.println(nama + "\t" + nim + "\t" + kelas + "\t" + ipk);
    }
}
```

2. Membuat class NodeMahasiswa01 yang berisi object dan konstruktor.

```
package Jobsheet11;
public class NodeMahasiswa01 {
    Mahasiswa01 data;
    NodeMahasiswa01 next;

    public NodeMahasiswa01 (Mahasiswa01 data, NodeMahasiswa01 next) {
        this.data = data;
        this.next = next;
    }
}
```

3. Membuat class SingleLinkList01 yang berisi method-method untuk algoritma Linked List.

```
package Jobsheet11;
public class SingleLinkedList01 {
    NodeMahasiswa01 head;
    NodeMahasiswa01 tail;

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

    public void print () {
        if (!isEmpty()) {
            NodeMahasiswa01 tmp = head;
            System.out.println("Isi Linked List:\t");
            while (tmp != null) {
                tmp.data.tampilInformasi();
                tmp = tmp.next;
            }
            System.out.println("");
        } else {
            System.out.println("Linked List kosong.");
        }

    }
    public void addFirst (Mahasiswa01 input) {
        NodeMahasiswa01 ndInput = new NodeMahasiswa01(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }
    public void addLast(Mahasiswa01 input) {
        NodeMahasiswa01 ndInput = new NodeMahasiswa01(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            tail.next = ndInput;
            tail = ndInput;
        }
    }
```

```
    public void insertAfter (String key, Mahasiswa01 input) {
        NodeMahasiswa01 ndInput = new NodeMahasiswa01(input, null);
        NodeMahasiswa01 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);
    }
    public void insertAt (int index, Mahasiswa01 input) {
        if (index < 0) {
            System.out.println("Indeks salah");
        } else if (index == 0) {
            addFirst(input);
        } else {
            NodeMahasiswa01 temp = head;
            for (int i = 0; i < index-1; i++) {
                temp = temp.next;
            }
            temp.next = new NodeMahasiswa01(input, temp.next);
            if (temp.next.next == null) {
                tail = temp.next;
            }
        }
    }

}
```

4. Membuat class main untuk memanggil method-method yang sudah dibuat.

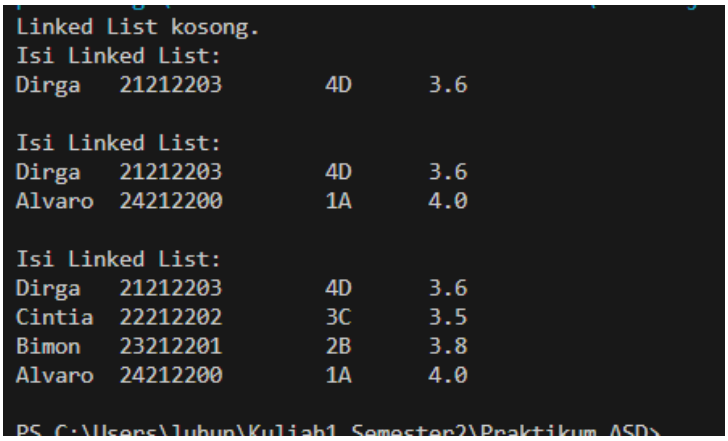
```
package Jobsheet11;

public class SLLMain01 {
    public static void main(String[] args) {
        SingleLinkedList01 sll = new SingleLinkedList01();
        Mahasiswa01 mhs1 = new Mahasiswa01("24212200", "Alvaro", "1A", 4.0);
        Mahasiswa01 mhs2 = new Mahasiswa01("23212201", "Bimon", "2B", 3.8);
        Mahasiswa01 mhs3 = new Mahasiswa01("22212202", "Cintia", "3C", 3.5);
        Mahasiswa01 mhs4 = new Mahasiswa01("21212203", "Dirga", "4D", 3.6);

        sll.print();
        sll.addFirst(mhs4);
        sll.print();
        sll.addLast(mhs1);
        sll.print();
        sll.insertAfter("Dirga", mhs3);
        sll.insertAt(2, mhs2);
        sll.print();

    }
}
```

5. Hasil running kode program



1.1.1 Pertanyaan

1. Hasil pertama compile menghasilkan “Linked List Kosong” karena method print() dipanggil terlebih dahulu sebelum data dimasukkan.

2. Variabel temp pada setiap method berfungsi sebagai wadah sementara saat ingin mengakses Linked List agar data yang ada pada Linked List tidak berubah.

3. Memodifikasi kode program agar dapat menginputkan data secara dinamis.

```
package Jobsheet11;
import java.util.Scanner;
public class SLLMain01 {
    public static void main(String[] args) {
        Scanner sc = new Scanner (System.in);
        SingleLinkedList01 sll = new SingleLinkedList01();
        int pilih;
        do{
            System.out.println("\n=== Menu Antrian layanan Akademik ===");
            System.out.println("1. Tambah Mahasiswa Paling Depan");
            System.out.println("2. Tambah Mahasiswa Paling Belakang");
            System.out.println("3. Tambah Mahasiswa Setelah");
            System.out.println("4. Tambah Mahassiwa Pada Posisi");
            System.out.println("5. Tampilkan Daftar Mahasiswa");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilih = sc.nextInt();
            sc.nextLine();
```

```
            switch (pilih) {
                case 1:
                    System.out.println("Masukkan data mahasiswa yang ingin ditambahkan.");
                    System.out.print("NIM     : ");
                    String nim = sc.nextLine();
                    System.out.print("Nama    : ");
                    String nama = sc.nextLine();
                    System.out.print("Kelas   : ");
                    String kelas = sc.nextLine();
                    System.out.print("IPK     : ");
                    double ipk = sc.nextDouble();
                    Mahasiswa01 mhs = new Mahasiswa01(nim, nama, kelas, ipk);
                    sll.addFirst(mhs);
                    break;

                case 2:
                    System.out.println("Masukkan data mahasiswa yang ingin ditambahkan.");
                    System.out.print("NIM     : ");
                    nim = sc.nextLine();
                    System.out.print("Nama    : ");
                    nama = sc.nextLine();
                    System.out.print("Kelas   : ");
                    kelas = sc.nextLine();
                    System.out.print("IPK     : ");
                    ipk = sc.nextDouble();
                    mhs = new Mahasiswa01(nim, nama, kelas, ipk);
                    sll.addLast(mhs);
                    break;

                case 3:
                    System.out.println("Masukkan data mahasiswa yang ingin ditambahkan.");
                    System.out.print("NIM     : ");
                    nim = sc.nextLine();
                    System.out.print("Nama    : ");
                    nama = sc.nextLine();
                    System.out.print("Kelas   : ");
                    kelas = sc.nextLine();
                    System.out.print("IPK     : ");
                    ipk = sc.nextDouble();
                    sc.nextLine();
                    mhs = new Mahasiswa01(nim, nama, kelas, ipk);
                    System.out.print("Setelah siapa anda ingin menambahkan data? : ");
                    String key = sc.nextLine();
                    sll.insertAfter(key, mhs);
                    break;
```

```
                case 4:
                    System.out.println("Masukkan data mahasiswa yang ingin ditambahkan.");
                    System.out.print("NIM     : ");
                    nim = sc.nextLine();
                    System.out.print("Nama    : ");
                    nama = sc.nextLine();
                    System.out.print("Kelas   : ");
                    kelas = sc.nextLine();
                    System.out.print("IPK     : ");
                    ipk = sc.nextDouble();
                    mhs = new Mahasiswa01(nim, nama, kelas, ipk);
                    System.out.print("Pada indeks ke berapa anda ingin menambahkan data? : ");
                    int index = sc.nextInt();
                    sc.nextLine();
                    sll.insertAt(index, mhs);
                    break;

                case 5:
                    sll.print();
                    break;
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Input tidah valid.");
                    break;
            }
        } while (pilih != 0);
        sc.close();
    }
}
```

1.2 Percobaan 2

1. Melanjutkan kode program dengan menambahkan method-method baru pada class SingleLinkedList01.

```
    }public void getData (int index) {
        NodeMahasiswa01 tmp = head;
        for (int i = 0; i < index; i++) {
            tmp = tmp.next;
        }
        tmp.data.tampilInformasi();
    }
    public int indexOf (String key) {
        NodeMahasiswa01 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;
        }
    }
```

```
    public void removeFirst () {
        if (isEmpty()) {
            System.out.println("Linked List masih kosong, tidak dapat dihapus!");
        } else if (head == tail) {
            head = tail = null;
        } else {
            head = head.next;
        }
    }
    public void removeLast () {
        if (isEmpty()) {
            System.out.println("Linked List masih kosong, tidak dapat dihapus!");
        } else if (head == tail) {
            head = tail = null;
        } else {
            NodeMahasiswa01 temp = head;
            while (temp.next != tail) {
                temp = temp.next;
            }
            temp.next = null;
            tail = temp;
        }
    }
    public void remove (String key) {
        if (isEmpty()) {
            System.out.println("Linked List masih kosong, tidak dapat dihapus!");
        } else {
            NodeMahasiswa01 temp = head;
            while (temp != null) {
                if ((temp.data.nama.equalsIgnoreCase(key)) && (temp == head)) {
                    this.removeFirst();
                    break;
                } else if (temp.data.nama.equalsIgnoreCase(key)) {
                    temp.next = temp.next.next;
                    if (temp.next == null) {
                        tail = temp;
                    }
                    break;
                }
                temp = temp.next;
            }
        }
    }
    public void removeAt (int index) {
        if (index == 0) {
            removeFirst();
        } else {
            NodeMahasiswa01 temp = head;
            for (int i = 0; i < index - 1; i++) {
                temp = temp.next;
            }
            temp.next = temp.next.next;
            if (temp.next == null) {
                tail = temp;
            }
        }
    }
}
```

2. Modifikasi pada class SLLMain01.

```
package Jobsheet11;
public class SLLMain01 {
    public static void main(String[] args) {
        SingleLinkedList01 sll = new SingleLinkedList01();
        Mahasiswa01 mhs1 = new Mahasiswa01("24212200", "Alvaro", "1A", 4.0);
        Mahasiswa01 mhs2 = new Mahasiswa01("23212201", "Bimon", "2B", 3.8);
        Mahasiswa01 mhs3 = new Mahasiswa01("22212202", "Cintia", "3C", 3.5);
        Mahasiswa01 mhs4 = new Mahasiswa01("21212203", "Dirga", "4D", 3.6);

        sll.print();
        sll.addFirst(mhs4);
        sll.print();
        sll.addLast(mhs1);
        sll.print();
        sll.insertAfter("Dirga", mhs3);
        sll.insertAt(2, mhs2);
        sll.print();
        System.out.println("data index 1: ");
        sll.getData(1);

        System.out.println("data mahasiswa an Bimon berada pada index: "+sll.indexOf("bimon"));
        System.out.println();

        sll.removeFirst();
        sll.removeLast();
        sll.print();
        sll.removeAt(0);
        sll.print();
    }
}
```

3. Hasil runing kode 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

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
```

1.2.1 Pertanyaan

1. Fungsi break pada method tersebut untuk menghentikan perulangan while setelah data yang dicari cocok dan sudah dihapus.

2. Pada class SingleLinkedList01 di method remove.

```
temp.next = temp.next.next;
if (temp.next == null) {
    tail = temp;
}
```

Kode tersebut berfungsi untuk memutus hubungan node yang ingin dihapus dengan node lainnya, sehingga node tersebut akan dianggap dihapus. Dan jika node yang dihapus bernilai null maka tail akan diperbarui ke node sebelumnya.

2. Tugas

1. Membuat kode program untuk layanan mahasiswa.

a. Class Mahasiswa01

```
package TugasJobsheet11;
public class Mahasiswa01 {
    String nim;
    String nama;
    String prodi;
    String kelas;

    public Mahasiswa01(String nim, String nama, String prodi, String kelas) {
        this.nim = nim;
        this.nama = nama;
        this.prodi = prodi;
        this.kelas = kelas;
    }
    public void tampilkanData() {
        System.out.println(nim + " - " + nama + " - " + prodi + " - " + kelas);
    }
}
```

Class Mahasiswa01 berfungsi sebagai tempat untuk menyimpan nilai dari atribut. Class ini juga berisi method tampilkanData() untuk menampilkan data yang ada di dalam objek.

b. NodeQueue01

```
package TugasJobsheet11;
public class NodeQueue01 {
    Mahasiswa01 data;
    NodeQueue01 next;

    public NodeQueue01(Mahasiswa01 data) {
        this.data = data;
        this.next = null;
    }
}
```

Class ini berisi struktur node yang berfungsi untuk menyimpan satu elemen antrian yang berisi data dan next.

c. AntrianLayanan01

```
package TugasJobsheet11;
public class AntrianLayanan01 {
    NodeQueue01 front, rear;
    int size;
    int max;

    public AntrianLayanan01(int max) {
        this.front = null;
        this.rear = null;
        this.size = 0;
        this.max = max;
    }
    public boolean isEmpty() {
        return front == null;
    }
    public boolean isFull() {
        return size >= max;
    }
    public void tambahAntrian(Mahasiswa01 mhs) {
        if (isFull()) {
            System.out.println("Antrian penuh. Tidak bisa menambah mahasiswa baru.");
            return;
        }
        NodeQueue01 baru = new NodeQueue01(mhs);
        if (isEmpty()) {
            front = rear = baru;
        } else {
            rear.next = baru;
            rear = baru;
        }
        size++;
        System.out.println(mhs.nama + " berhasil masuk ke antrian.");
    }
```

```
    public Mahasiswa01 layaniMahasiswa() {
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
            return null;
        }
        Mahasiswa01 mhs = front.data;
        if (front == rear) {
            front = rear = null;
        } else {
            front = front.next;
        }
        size--;
        return mhs;
    }
    public void lihatTerdepan() {
        if (isEmpty()) {
            System.out.println("Antrian kosong");
        } else {
            System.out.println("Mahasiswa terdepan:");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            front.data.tampilkanData();
        }
    }
    public void lihatAkhir() {
        if (isEmpty()) {
            System.out.println("Antrian kosong");
        } else {
            System.out.println("Mahasiswa paling akhir:");
            System.out.println("NIM - NAMA - PRODI - KELAS");
            rear.data.tampilkanData();
        }
    }
    public void tampilkanSemua() {
        if (isEmpty()) {
            System.out.println("Antrian kosong");
            return;
        }
        System.out.println("Daftar Mahasiswa dalam antrian:");
        System.out.println("NIM - NAMA - PRODI - KELAS");
        NodeQueue01 temp = front;
        int i = 1;
        while (temp != null) {
            System.out.print(i + ". ");
            temp.data.tampilkanData();
            temp = temp.next;
            i++;
        }
    }
    public int getJumlahAntrian() {
        return size;
    }
    public void clear() {
        front = rear = null;
        size = 0;
        System.out.println("Antrian berhasil dikosongkan.");
    }
}
```

Class AntrianLayanan01 berisi method-method untuk algoritma Queue berbasis Linked List. Class ini digunakan untuk mengelola layanan antrian mahasiswa.

d. LayananMahasiswa01

```
package TugasJobsheet11;
import java.util.Scanner;
public class LayananAkademikSiakad01 {
    public static void main(String[] args) {
        Scanner sc = new Scanner (System.in);
        System.out.println("Masukkan panjang antrian: ");
        int max = sc.nextInt();
        AntrianLayanan01 antrian = new AntrianLayanan01(max);
        int pilihan;
        do{
            System.out.println("\n=== Menu Antrian layanan Akademik ===");
            System.out.println("1. Tambah Mahasiswa ke Antrian");
            System.out.println("2. Layani Mahasiswa");
            System.out.println("3. Lihat Mahasiswa Terdepan");
            System.out.println("4. Lihat Mahasiswa Paling Akhir");
            System.out.println("5. Lihat Semua Antrian");
            System.out.println("6. Jumlah Mahasiswa dalam Antrian");
            System.out.println("7. Kosongkan Antrian");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilihan = sc.nextInt();
            sc.nextLine();
            switch (pilihan) {
                case 1:
                    System.out.print("NIM     : ");
                    String nim = sc.nextLine();
                    System.out.print("Nama    : ");
                    String nama = sc.nextLine();
                    System.out.print("Prodi   : ");
                    String prodi = sc.nextLine();
                    System.out.print("Kelas   : ");
                    String kelas = sc.nextLine();
                    Mahasiswa01 input = new Mahasiswa01(nim, nama, prodi, kelas);
                    antrian.tambahAntrian(input);
                    break;

                case 2:
                    Mahasiswa01 dilayani = antrian.layaniMahasiswa();
                    if (dilayani != null) {
                        System.out.print("Melayani mahasiswa: ");
                        dilayani.tampilkanData();
                    }
                    break;

                case 3:
                    antrian.lihatTerdepan();
                    break;
                case 4:
                    antrian.lihatAkhir();
                    break;
                case 5:
                    antrian.tampilkanSemua();
                    break;
```

```
                case 6:
                    System.out.println("Jumlah data antrian: " +
antrian.getJumlahAntrian());
                    break;
                case 7:
                    antrian.clear();
                    System.out.println("Antrian sudah dikosongkan!");
                case 0:
                    System.out.println("Terima kasih.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid!");
            }
        } while (pilihan != 0);
        sc.close();
    }
}
```

2. Hasil running kode program
   a. Menu 1:

```
Masukkan panjang antrian: 5

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM     : 123
Nama    : Adi
Prodi   : TI
Kelas   : 1E
Adi berhasil masuk ke antrian.

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM     : 124
Nama    : Budi
Prodi   : TI
Kelas   : 1B
Budi berhasil masuk ke antrian.
```

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 1
NIM     : 125
Nama    : Nasid
Prodi   : TI
Kelas   : 1A
Nasid berhasil masuk ke antrian.

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu:
```

b. Menu 2:

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 2
Melayani mahasiswa: 123 - Adi - TI - 1E

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu:
```

c. Menu 3:

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 3
Mahasiswa terdepan:
NIM - NAMA - PRODI - KELAS
124 - Budi - TI - 1B

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu:
```

d. Menu 4:

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 4
Mahasiswa paling akhir:
NIM - NAMA - PRODI - KELAS
125 - Nasid - TI - 1A

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu:
```

e. Menu 5:

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 5
Daftar Mahasiswa dalam antrian:
NIM - NAMA - PRODI - KELAS
1. 124 - Budi - TI - 1B
2. 125 - Nasid - TI - 1A

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu:
```

f. Menu 6:

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 6
Jumlah data antrian: 2

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu:
```

g. Menu 7:

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 7
Antrian berhasil dikosongkan.

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 5
Antrian kosong

=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu:
```

h. Menu 0:

```
=== Menu Antrian layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Mahasiswa Paling Akhir
5. Lihat Semua Antrian
6. Jumlah Mahasiswa dalam Antrian
7. Kosongkan Antrian
0. Keluar
Pilih menu: 0
Terima kasih.
```

3. Push kode program ke github

```
PS C:\Users\luhun\Kuliah1 Semester2\Praktikum_ASD> git add .
PS C:\Users\luhun\Kuliah1 Semester2\Praktikum_ASD> git commit -m "Jobsheet 11"
[main 3370396] Jobsheet 11
 8 files changed, 397 insertions(+)
 create mode 100644 Jobsheet11/Mahasiswa01.java
 create mode 100644 Jobsheet11/NodeMahasiswa01.java
 create mode 100644 Jobsheet11/SLLMain01.java
 create mode 100644 Jobsheet11/SingleLinkedList01.java
 create mode 100644 TugasJobsheet11/AntrianLayanan01.java
 create mode 100644 TugasJobsheet11/LayananAkademikSiakad01.java
 create mode 100644 TugasJobsheet11/Mahasiswa01.java
 create mode 100644 TugasJobsheet11/NodeQueue01.java
PS C:\Users\luhun\Kuliah1 Semester2\Praktikum_ASD> git push origin main
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (12/12), 3.76 KiB | 1.88 MiB/s, done.
Total 12 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote: This repository moved. Please use the new location:
remote:   https://github.com/Kumaaan/Praktikum_ASD.git
To https://github.com/Kumaaan/Praktikum-ASD.git
   1c66c2f..3370396  main -> main
PS C:\Users\luhun\Kuliah1 Semester2\Praktikum_ASD>
```