LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

QUEUE

AHMAD SOFYAN BADAWI

244107020073

KELAS TI-1B

PRODI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

Percobaan 1:

Langkah-langkah:

1. Buat folder baru bernama P1Jobsheet10 di dalam repository Praktikum ASD, kemudian buat class baru dengan nama Queue.
2. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti berikut ini :

```
public Queue(int n) {
    max = n;
    data = new int[max];
    size = 0;
    front = rear = -1;
}
```

3. Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong.

```
public boolean IsEmpty() {
    if (size == 0) {
        return true;
    } else {
        return false;
    }
}
```

4. Buat method IsFull bertipe boolean yang digunakan untuk mengecek apakah queue sudah penuh.

```
public boolean IsFull() {
    if (size == max) {
        return true;
    } else {
        return false;
    }
}
```

5. Buat method peek bertipe void untuk menampilkan elemen queue pada posisi paling depan.

```
public void peek() {
    if (!IsEmpty()) {
        System.out.println("Elemen terdepan: " + data[front]);
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}
```

6. Buat method print bertipe void untuk menampilkan seluruh elemen pada queue mulai dari posisi front sampai dengan posisi rear.

```
public void print() {
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong!");
    } else {
        int i = front;
        while (i != rear) {
            System.out.println(data[i] + " ");
            i = (i + 1) % max;
        }
        System.out.println(data[i] + " ");
        System.out.println("Jumlah elemen = " + size);
    }
}
```

7. Buat method clear bertipe void untuk menghapus semua elemen pada queue

```
public void clear() {
    if (!IsEmpty()) {
        front = rear = -1;
        size = 0;
        System.out.println(x:"Queue berhasil diksosongkan");
    } else {
        System.out.println(x:"Queue masih kosong!");
    }
}
```

8. Buat method Enqueue bertipe void untuk menambahkan isi queue dengan parameter dt yang bertipe integer

```
public void Enqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh!");
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}
```

9. Buat method Dequeue bertipe int untuk mengeluarkan data pada queue di posisi belakang

```
public int Dequeue() {
    int dt = 0;
    if (IsEmpty()) {
        System.out.println(x:"Queue masih kosong!");
    } else {
        dt = data[front];
        size--;
        if (IsEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    }
    return dt;
}
```

10. Selanjutnya, buat class baru dengan nama QueueMain tetap pada package Praktikum1. Buat method menu bertipe void untuk memilih menu program pada saat dijalankan.

```
public static void menu() {
    System.out.println(x:"Masukkan operasi yang diinginkan");
    System.out.println(x:"1. Enqueue");
    System.out.println(x:"2. Dequeue");
    System.out.println(x:"3. Print");
    System.out.println(x:"4. Peek");
    System.out.println(x:"5. Clear");
    System.out.println(x:"--------------------------------");
}
```

11. Buat fungsi main, kemudian deklarasikan Scanner dengan nama sc.

```
public class QueueMain {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

12. Buat variabel n untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.

```
        System.out.print(s:"Masukkan kapasitas queue: ");
        int n = sc.nextInt();
```

13. Lakukan instansiasi objek Queue dengan nama Q dengan mengirimkan parameter n sebagai kapasitas elemen queue

```
        Queue Q = new Queue(n);
```

14. Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna.

```
        int pilih;
```

15. Lakukan perulangan menggunakan do-while untuk menjalankan program secara terus menerus sesuai masukan yang diberikan. Di dalam perulangan tersebut, terdapat pemilihan kondisi menggunakan switch-case untuk menjalankan operasi queue sesuai dengan masukan pengguna.

```
do {
    menu();
    System.out.print(s:"Masukkan: ");
    pilih = sc.nextInt();

    switch (pilih) {
        case 1:
            System.out.print(s:"Masukkan data baru: ");
            int dataMasuk = sc.nextInt();
            Q.Enqueue(dataMasuk);
            break;

        case 2:
            int dataKeluar = Q.Dequeue();
            if (dataKeluar != 0) {
                System.out.println("Data yang dikeluarkan: " + dataKeluar);
            }
            break;

        case 3:
            Q.print();
            break;

        case 4:
            Q.peek();
            break;

        case 5:
            Q.clear();
            break;

        default:
            break;
    }
} while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
```

16. Compile dan jalankan class QueueMain, kemudian amati hasilnya.

Verifikasi:

```
Masukkan kapasitas queue: 4
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
--------------------------------
Masukkan: 1
Masukkan data baru: 15
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
--------------------------------
Masukkan: 1
Masukkan data baru: 31
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
--------------------------------
Masukkan: 4
Elemen terdepan: 15
Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
--------------------------------
Masukkan:
```

Pertanyaan:

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
   Jawab: Front dan rear pada awal diberi value -1 itu ditujukan untuk menandakan bahwasannya data masih kosong, jika menggunakan 0 itu akan merujuk indeks 0 yang berarti datanya ada maka dari itu digunakan value -1. Size ditujukkan untuk menandakan banyaknya data yang ada pada Queue, seperti tadi front dan rear bernilai -1 menandakan belum ada data maka dari itu size yang bertugas sebagai penanda jumlah data yang ada di queue di set valuenya 0
2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!

```
if (rear == max - 1) {
    rear = 0;
```

Jawab: Kode diatas digunakan untuk ngecek apakah data paling terakhir yang ada sudah sampai diujung array (indeks terakhir array) jika ternyata rear sudah ada di ujung maka akan dimasukkan data yang baru di indeks ke 0 (menggunakan kembali ruang kosong di awal array (jika ada))

3. Pada method Dequeue, jelaskan maksud dan kegunaan dari potongan kode berikut!

```
if (front == max - 1) {
    front = 0;
```

Jawab: Sama seperti Enqueue tetapi front, kode tersebut digunakan untuk ngecek apakah front yang ada sudah sampai diujung array (indeks terakhir array), jika benar maka front akan di set kembali di indeks ke 0

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
Jawab: int i = front ditujukkan untuk ngeprint data, karena data Queue selalu diawali dengan front, maka dari itu digunakan kode i = front

5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!

```
i = (i + 1) % max;
```

Jawab: Kode tersebut ditujukkan untuk mengeprint seluruh data yang ada pada Queue karena bisa saja ada data baru yang ada pada indeks ke 0 (rear tidak pada indeks terakhir array) maka jika tidak menggunakan kode tersebut data baru yang sudah dimasukkan tidak akan terbaca dan tidak akan terprint. Perulangan ini juga ditujukkan untuk menghindari error (melewati batas array)

6. Tunjukkan potongan kode program yang merupakan queue overflow!
Jawab:

```
public boolean Enqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh!");
        System.out.println(x:"Queue Overflow!");
        return false;
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
        return true;
    }
}
```

```
case 1:
    System.out.print("Masukkan data baru: ");
    int dataMasuk = sc.nextInt();
    lanjut = Q.Enqueue(dataMasuk);
    break;
```

7. Pada saat terjadi queue overflow dan queue underflow, program tersebut tetap dapat berjalan dan hanya menampilkan teks informasi. Lakukan modifikasi program sehingga pada saat terjadi queue overflow dan queue underflow, program dihentikan!

Jawab:

```
public boolean Enqueue(int dt) {
    if (IsFull()) {
        System.out.println(x:"Queue sudah penuh!");
        System.out.println(x:"Queue Overflow!");
        return false;
    } else {
        if (IsEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
        return true;
    }
}
```

Pada main:

```
boolean lanjut = true;
```

```
case 1:
    System.out.print("Masukkan data baru: ");
    int dataMasuk = sc.nextInt();
    lanjut = Q.Enqueue(dataMasuk);
    break;
```

```
case 2:
    int dataKeluar = Q.Dequeue();
    if (dataKeluar != 0) {
        System.out.println("Data yang dikeluarkan: " + dataKeluar);
    } else {
        System.out.println("Queue Underflow!");
        lanjut = false;
    }
    break;
```

```
} while (pilih >= 1 && pilih <= 5 && lanjut);
```

Percobaan 2:

Langkah-langkah:

1. Buat folder baru bernama P2Jobsheet10 di dalam repository Praktikum ASD, kemudian buat class baru dengan nama Mahasiswa.
2. Tambahkan atribut-atribut Nasabah seperti pada Class Diagram, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```
public Mahasiswa(String nim, String nama, String prodi, String kelas)
    this.nama = nama;
    this.nim = nim;
    this.prodi = prodi;
    this.kelas = kelas;
}
```

Dan tambahkan method tampilkanData berikut :

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

3. Salin kode program class Queue pada Praktikum 1 untuk digunakan kembali pada Praktikum 2 ini, ganti nama class-nya dengan AntrianLayanan. Karena pada Praktikum 1, data yang disimpan pada queue hanya berupa array bertipe integer, sedangkan pada Praktikum 2 data yang digunakan adalah object, maka perlu dilakukan modifikasi pada class AntrianLayanan tersebut.

```
Mahasiswa[] data;
int front, rear, size, max;

public AntrianLayanan(int max) {
    this.max = max;
    this.data = new Mahasiswa[max];
    this.front = 0;
    this.rear = -1;
    this.size = 0;
}
```

4. Lakukan modifikasi pada class AntrianLayanan dengan mengubah tipe int[] data menjadi Mahasiswa[] data karena pada kasus ini data yang akan disimpan berupa object Mahasiswa. Modifikasi perlu dilakukan pada atribut, method Enqueue, dan method Dequeue.

```
public void tambahAntrian(Mahasiswa mhs) {
    if (IsFull()) {
        System.out.println(x:"Antrian penuh, tidak dapat menambah Mahasiswa.");
        return;
    }
    rear = (rear + 1) % max;
    data[rear] = mhs;
    size++;
    System.out.println(mhs.nama + " behasil masuk ke antrian.");
```

```
public Mahasiswa layaniMahasiswa() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
        return null;
    }
    Mahasiswa mhs = data[front];
    front = (front + 1) % max;
    size--;
    return mhs;
}
```

5. Berikutnya method peek dan print yaitu untuk menampilkan data antrian layanan paling depan dan menampilkan semua data antrian layanan.

```
public void lihatTerdepan() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
    } else {
        System.out.println(x:"Mahasiswa terdepan: ");
        System.out.println(x:"NIM - NAMA - PRODI - KELAS");
        data[front].tampilkanData();
    }
}

public void tampilkanSemua() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
        return;
    }
    System.out.println(x:"Daftar Mahasiswa dalam Antrian:");
    System.out.println(x:"NIM - NAMA - PRODI - KELAS");
    for (int i = 0; i < size; i++) {
        int index = (front + i) % max;
        System.out.print((i + 1) + ". ");
        data[index].tampilkanData();
    }

}
```

Ditambahkan dengan method getJumlahAntrian yaitu menampilkan nilai size:

```
public int getJumlahAntrian() {
    return size;
}
```

6. Selanjutnya, buat class baru dengan nama LayananAkademikSIAKAD tetap pada package yang sama. Buat fungsi main, deklarasikan Scanner dengan nama sc.
7. Kemudian lakukan instansiasi objek AntrianLayanan dengan nama antrian dan nilai parameternya adalah nilai maksimal antrian yang ditentukan (misal sama dengan 5).
8. Deklarasikan variabel dengan nama pilihan bertipe integer untuk menampung pilih menu dari pengguna.

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    AntrianLayanan antrian = new AntrianLayanan(max:5);
    int pilihan;
```

9. Tambahkan kode berikut untuk melakukan perulangan menu sesuai dengan masukan yang diberikan oleh pengguna.

```
    do {
        System.out.println(x:"\n--- Menu Antrian Layanan Akademik ---");
        System.out.println(x:"1. Tambah Mahasiswa ke Antrian");
        System.out.println(x:"2. Layani Mahasiswa");
        System.out.println(x:"3. Lihat Mahasiswa Terdepan");
        System.out.println(x:"4. Lihat Semua Antrian");
        System.out.println(x:"5. Jumlah Mahasiswa dalam Antrian");
        System.out.println(x:"6. Cek Antrian Belakang");
        System.out.println(x:"0. Keluar");
        System.out.print(s:"Pilih menu: ");
        pilihan = sc.nextInt();
        sc.nextLine();

        switch (pilihan) {
            case 1:
                System.out.print(s:"NIM: ");
                String nim = sc.nextLine();
                System.out.print(s:"Nama: ");
                String nama = sc.nextLine();
                System.out.print(s:"Prodi: ");
                String prodi = sc.nextLine();
                System.out.print(s:"Kelas: ");
                String kelas = sc.nextLine();
                Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
                antrian.tambahAntrian(mhs);
                break;

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

            case 3:
                antrian.lihatTerdepan();
                break;

            case 4:
                antrian.tampilkanSemua();
                break;

            case 5:
                System.out.println("Jumlah dalam antrian: " + antrian.getJumlahAntrian());
                break;

            case 6:
                antrian.lihatTerbelakang();
                break;

            case 0:
                System.out.println(x:"Terima kasih.");
                break;

            default:
                System.out.println(x:"Pilihan tidak valid.");
        }
    } while (pilihan != 0 && pilihan <= 6);

    sc.close();
}
```

Verifikasi:

```
=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 1
NIM: 123
Nama: Aldi
Prodi: TI
Kelas: 1A
Aldi behasil masuk ke antrian.

=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 1
NIM: 124
Nama: Bobi
Prodi: TI
Kelas: 1G
Bobi behasil masuk ke antrian.
```

```
=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 4
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 123 - Aldi - TI - 1A
2. 124 - Bobi - TI - 1G

=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 2
Melayani mahasiswa: 123 - Aldi - TI - 1A

=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 4
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 124 - Bobi - TI - 1G
```

```
=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 5
Jumlah dalam antrian: 1

=== Menu Antrian Layanan Akademik ===
1. Tambah Mahasiswa ke Antrian
2. Layani Mahasiswa
3. Lihat Mahasiswa Terdepan
4. Lihat Semua Antrian
5. Jumlah Mahasiswa dalam Antrian
0. Keluar
Pilih menu: 0
Terima kasih.
```

Pertanyaan:

1. Lakukan modifikasi program dengan menambahkan method baru bernama LihatAkhir pada class AntrianLayanan yang digunakan untuk mengecek antrian yang berada di posisi belakang. Tambahkan pula daftar menu 6. Cek Antrian paling belakang pada class LayananAkademikSIAKAD sehingga method LihatAkhir dapat dipanggil!
   Jawab:

```
public void lihatTerbelakang() {
    if (IsEmpty()) {
        System.out.println(x:"Antrian kosong.");
    } else {
        System.out.println(x:"Mahasiswa terdepan: ");
        System.out.println(x:"NIM - NAMA - PRODI - KELAS");
        data[rear].tampilkanData();
    }
}
```

```
case 6:
    antrian.lihatTerbelakang();
    break;
```

Tugas:

Buatlah program antrian untuk mengilustasikan antrian persetujuan Kartu Rencana Studi (KRS) Mahasiswa oleh Dosen Pembina Akademik (DPA). Ketika seorang mahasiswa akan mengantri, maka dia harus mendaftarkan datanya (data mahasiswa seperti pada praktikum 2). Gunakan class untuk antrian seperti pada Praktikum 1 dan 2, dengan method-method yang berfungsi :

- Cek antrian kosong, Cek antrian penuh, Mengosongkan antrian.
- Menambahkan antrian, Memanggil antrian untuk proses KRS – setiap 1x panggilan terdiri dari 2 mahasiswa (pada antrian no 1 dan 2)
- Menampilkan semua antrian, Menampilkan 2 antrian terdepan, Menampilkan antrian paling akhir.
- Cetak jumlah antrian, Cetak jumlah yang sudah melakukan proses KRS
- Jumlah antrian maximal 10, jumlah yang ditangani masing-masing DPA 30 mahasiswa, cetak jumlah mahasiswa yang belum melakukan proses KRS.

Gambarkan Diagram Class untuk antriannya. Implementasikan semua method menggunakan menu pilihan pada fungsi main.

Output:

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 3
Mahasiswa terdepan:
NIM - NAMA - PRODI - KELAS
1 - Abim - TI - 1b
2 - Anval - TI - 1b
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 4
Mahasiswa terdepan:
NIM - NAMA - PRODI - KELAS
10 - Aziz - TI - 1b
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 5
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 1 - Abim - TI - 1b
2. 2 - Anval - TI - 1b
3. 3 - Adel - TI - 1b
4. 4 - Dawi - TI - 1b
5. 5 - Angga - TI - 1b
6. 6 - Alfat - TI - 1b
7. 7 - Aras - TI - 1b
8. 8 - Aqil - TI - 1b
9. 9 - Axel - TI - 1b
10. 10 - Aziz - TI - 1b
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 6
Jumlah dalam antrian yang tersisa: 10
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 7
Jumlah yang sudah proses KRS: 0
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 8
Jumlah yang belum proses KRS: 30
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 2
Memproses mahasiswa: 1 - Abim - TI - 1b
2 - Anval - TI - 1b
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 3
Mahasiswa terdepan:
NIM - NAMA - PRODI - KELAS
3 - Adel - TI - 1b
4 - Dawi - TI - 1b
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 5
Daftar Mahasiswa dalam Antrian:
NIM - NAMA - PRODI - KELAS
1. 3 - Adel - TI - 1b
2. 4 - Dawi - TI - 1b
3. 5 - Angga - TI - 1b
4. 6 - Alfat - TI - 1b
5. 7 - Aras - TI - 1b
6. 8 - Aqil - TI - 1b
7. 9 - Axel - TI - 1b
8. 10 - Aziz - TI - 1b
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 6
Jumlah dalam antrian yang tersisa: 8
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 7
Jumlah yang sudah proses KRS: 2
```

```
=== Menu Antrian Persetujuan KRS ===
1. Tambah Antrian
2. Memanggil Antrian
3. Tampil 2 Antrian Terdepan
4. Tampil Antrian Belakang
5. Tampil Semua Antrian
6. Jumlah Antrian tersisa
7. Jumlah yang sudah melakukan KRS
8. Jumlah yang belum melakukan KRS
9. Mengosongkan Antrian
0. Keluar
Pilih menu: 8
Jumlah yang belum proses KRS: 28
```

dst.