Nama : Zaskia Maulidina Mutiara Hati

NIM : 244107060056

Kelas : SIB 1A

# JOBSHEET 10 PRAKTIKUM ALGORITMA STRUKTUR DATA

## Percobaan 1

- **Class Queue**

jobsheet_10 > queue.java > queue

```java
package jobsheet_10;

public class queue {
    int [] data;
    int front, rear, size, max;

    public queue(int n) {
        max = n;
        data = new int[max];
        front = rear = -1;
    }

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

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

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

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

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

```
    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++;
        }
    }
```

```
    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;
    }
}
```

- **Class QueueMain**

```
package jobsheet_10;

import java.util.Scanner;

public class queueMain {
    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:"--------------------------------");
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print(s:"Masukkan kapasitas queue: ");
        int n = sc.nextInt();

        queue q = new queue(n);
        int pilih;
        do {
            menu();
            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:
            }
        } while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
    }
}
```

- **Output**

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

**Pertanyaan 1**

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
   - Nilai awal atribut front dan rear bernilai -1 untuk menandakan bahwa queue masih kosong. Jika front dan rear diberi nilai 0 sejak awal, maka itu menunjukkan bahwa queue sudah memiliki satu elemen, karena indeks 0 dalam array sudah dianggap terisi.
   - Atribut size bernilai 0, berarti queue masih kosong. Hal ini karena size menunjukkan jumlah elemen yang saat ini ada di dalam queue.

2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!

```
If (rear == max -1) {
    Rear = 0;
```

   - Berfungsi untuk memutar rear kembali ke depan (index 0), saat rear sudah mentok di belakang (index paling akhir), agar tempat kosong di depan bisa di isi lagi.

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

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

   - Berfungsi untuk memutar front Kembali ke depan (index 0) jika front sudah bmentok di belakang (index paling akhir).

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
   - Pada metode print, perulangan dimulai dari front dan bukan 0 karena antrian yang sudah melalui operasi dequeue tidak selalu dimulai dari indeks 0. Setelah elemen dikeluarkan, posisi front bergeser ke indeks berikutnya, dan elemen yang masih ada berada di belakang front.

5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!
   i = (i + 1) % max;
   - Untuk memastikan bahwa setelah mencapai indeks terakhir (indeks max - 1), perulangan akan kembali ke awal (indeks 0), menjaga agar antrian tetap berputar dan memanfaatkan ruang kosong secara efisien.

6. Tunjukkan potongan kode program yang merupakan queue overflow!
   -
```
public boolean isFull() {
    if (size == max) {
        return true;
    } else {
        return false;
    }
}
```

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!
   -
```
public void enqueue(int dt) {
    if (isFull()) {
        System.out.println(x:"Queue sudah penuh");
        System.exit(status:0);
    } else {
        if (isEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
                rear++;
            }
        }
        data[rear] = dt;
        size++;
    }
}
```
```
public int dequeue() {
    int dt = 0;
    if (isEmpty()) {
        System.out.println(x:"Queue masih kosong");
        System.exit(status:0);
    } else {
        dt = data[front];
        size--;
        if (isEmpty()) {
            front = rear = -1;
        } else {
            if (front == max - 1) {
                front = 0;
            } else {
                front++;
            }
        }
    }
    return dt;
}
```

**Percobaan 2**

- **Class Mahasiswa**

```java
package jobsheet_10.P2jobsheet10;

public class Mahasiswa {
    String nim, nama, prodi, kelas;

    public Mahasiswa(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 AntrianLayanan**

```java
package jobsheet_10.P2jobsheet10;

public class AntrianLayanan {
    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;
    }

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

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

```
31      public void tambahAntrian(Mahasiswa mhs) {
32          if (isFull()) {
33              System.out.println(x:"Antrian penuh, tidak dapat menambah mahasiswa.");
34              return;
35          }
36          rear = (rear + 1) % max;
37          data[rear] = mhs;
38          size++;
39          System.out.println(mhs.nama + " berhasil masuk ke antrian.");
40      }
41
42      public Mahasiswa layaniMahasiswa() {
43          if (isEmpty()) {
44              System.out.println(x:"Antrian kosong.");
45              return null;
46          }
47          Mahasiswa mhs = data[front];
48          front = (front + 1) % max;
49          size--;
50          return mhs;
51      }
52
53      public void lihatTerdepan() {
54          if (isEmpty()) {
55              System.out.println(x:"Antrian kosong.");
56          } else {
57              System.out.print(s:"Mahasiswa terdepan: ");
58              System.out.println(x:"NIM - NAMA - PRODI - KELAS");
59              data[front].tampilkanData();
60          }
61      }
```

```
63      public void tampilkanSemua() {
64          if (isEmpty()) {
65              System.out.println(x:"Antrian kosong.");
66              return;
67          }
68          System.out.println(x:"Daftar Mahasiswa dalam Antrian:");
69          System.out.println(x:"NIM - NAMA - PRODI - KELAS");
70          for (int i = 0; i < size; i++) {
71              int index = (front + i) % max;
72              System.out.print((i + 1) + ". ");
73              data[index].tampilkanData();
74          }
75      }
76
77      public int getJumlahAntrian() {
78          return size;
79      }
80
81  }
```

- **Class LayananAkademikSIAKAD**

```java
package jobsheet_10.P2jobsheet10;
import java.util.Scanner;
public class LayananAkademikSIAKAD {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianLayanan antrian = new AntrianLayanan(max:5);
        int pilihan;

        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:"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 0:
                    System.out.println(x:"Terima Kasih.");
                    break;
                default:
                    System.out.println(x:"Pilihan tidak valid.");
            }
        } while (pilihan != 0);
    }
}
```

- **Output**

```
=== 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 berhasil 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 berhasil 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 2**

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!

```
public void lihatTerakhir() {
    if (isEmpty()) {
        System.out.println(x:"Antrian kosong.");
    } else {
        System.out.print(s:"Mahasiswa terbelakang: ");
        System.out.println(x:"NIM - NAMA - PRODI - KELAS");
        data[rear].tampilkanData();
    }
}
```

-

```
=== 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
6. Lihat Mahasiswa Terakhir
0. Keluar
Pilih menu: 6
Mahasiswa terbelakang: NIM - NAMA - PRODI - KELAS
126 - Rara - SIB - 1F
```

**Tugas**

- **Class MahasiswaKRS**

jobsheet_10 > MahasiswaKRS.java > MahasiswaKRS

```java
package jobsheet_10;

public class MahasiswaKRS {
    String nim;
    String nama;

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

    public void tampil() {
        System.out.println("NIM : " + nim + ", Nama : " + nama);
    }
}
```

- **Class AntrianKRS**

jobsheet_10 > AntrianKRS.java > AntrianKRS > menambahAntrian(MahasiswaKRS)

```java
package jobsheet_10;
public class AntrianKRS {
    MahasiswaKRS[] data;
    int front, rear, size, max;
    int totalDiproses = 0;

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

    public boolean isEmpty() {
        return size == 0;
    }

    public boolean isFull() {
        return size == max;
    }
```

```
23      public void menambahAntrian(MahasiswaKRS mhs) {
24          if (isFull()) {
25              System.out.println(x:"Antrian penuh, tidak dapat menambah mahasiswa.");
26              return;
27          }
28          rear = (rear + 1) % max;
29          data[rear] = mhs;
30          size++;
31          System.out.println(mhs.nama + " berhasil masuk ke antrian.");
32      }
33
34      public void panggilAntrian() {
35          if (size < 2) {
36              System.out.println(x:"Antrian kurang dari 2, belum bisa dipanggil.");
37              return;
38          }
39          for (int i = 0; i < 2 && size > 0; i++) {
40              MahasiswaKRS mhs = data[front];
41              System.out.print(s:"Memproses: ");
42              mhs.tampil();
43              front = (front + 1) % max;
44              size--;
45              totalDiproses++;
46          }
47      }
```

```
49      public void tampilkanSemuaAntrian() {
50          if (isEmpty()) {
51              System.out.println(x:"Antrian kosong.");
52              return;
53          }
54          System.out.println(x:"Daftar mahasiswa dalam antrian:");
55          for (int i = 0; i < size; i++) {
56              int idx = (front + i) % max;
57              data[idx].tampil();
58          }
59      }
60
61      public void tampilkanDuaTerdepan() {
62          if (size < 1) {
63              System.out.println(x:"Antrian kosong.");
64              return;
65          }
66          System.out.println(x:"2 Mahasiswa terdepan:");
67          for (int i = 0; i < Math.min(a:2, size); i++) {
68              int idx = (front + i) % max;
69              data[idx].tampil();
70          }
71      }
```

```
    public void tampilkanTerakhir() {
        if (isEmpty()) {
            System.out.println("Antrian kosong.");
        } else {
            System.out.println("Mahasiswa terakhir:");
            data[rear].tampil();
        }
    }

    public void clear() {
        if (!isEmpty()) {
            front = 0;
            rear = -1;
            size = 0;
            System.out.println("Antrian berhasil dikosongkan.");
        } else {
            System.out.println("Antrian sudah kosong.");
        }
    }

    public void cetakJumlahAntrian() {
        System.out.println("Jumlah antrian saat ini: " + size);
    }

    public void cetakJumlahProses() {
        System.out.println("Jumlah mahasiswa yang telah diproses: " + totalDiproses);
    }

    public void cetakBelumProses() {
        int sisa = 30 - totalDiproses;
        System.out.println("Sisa mahasiswa yang belum diproses (maks 30): " + sisa);
    }
}
```

- **Class LayananAntrianKRS**

```
package jobsheet_10;
import java.util.Scanner;
public class LayananAntrianKRS {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        AntrianKRS antrian = new AntrianKRS(max:10);

        int pilih;
        do {
            System.out.println("\n--- MENU ANTRIAN KRS ---");
            System.out.println("1. Tambah Antrian");
            System.out.println("2. Panggil 2 Mahasiswa");
            System.out.println("3. Tampilkan Semua Antrian");
            System.out.println("4. Tampilkan 2 Terdepan");
            System.out.println("5. Tampilkan Antrian Terakhir");
            System.out.println("6. Kosongkan Antrian");
            System.out.println("7. Cetak Jumlah Antrian");
            System.out.println("8. Cetak Jumlah yang Sudah Diproses");
            System.out.println("9. Cetak Mahasiswa Belum Diproses");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilih = sc.nextInt(); sc.nextLine();
```

```
24              switch (pilih) {
25                  case 1:
26                      if (!antrian.isFull()) {
27                          System.out.print(s:"Masukkan NIM: ");
28                          String nim = sc.nextLine();
29                          System.out.print(s:"Masukkan Nama: ");
30                          String nama = sc.nextLine();
31                          MahasiswaKRS m = new MahasiswaKRS(nim, nama);
32                          antrian.menambahAntrian(m);
33                      } else {
34                          System.out.println(x:"Antrian sudah penuh.");
35                      }
36                      break;
37                  case 2:
38                      antrian.panggilAntrian();
39                      break;
40                  case 3:
41                      antrian.tampilkanSemuaAntrian();
42                      break;
43                  case 4:
44                      antrian.tampilkanDuaTerdepan();
45                      break;
46                  case 5:
47                      antrian.tampilkanTerakhir();
48                      break;
49                  case 6:
50                      antrian.clear();
51                      break;
52                  case 7:
53                      antrian.cetakJumlahAntrian();
54                      break;
55                  case 8:
56                      antrian.cetakJumlahProses();
57                      break;
58                  case 9:
59                      antrian.cetakBelumProses();
60                      break;
61                  case 0:
62                      System.out.println(x:"Terima kasih.");
63                      break;
64                  default:
65                      System.out.println(x:"Pilihan tidak valid.");
66              }
67          } while (pilih != 0);
68      }
69  }
```

- **Output**

```
--- MENU ANTRIAN KRS ---
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 1
Masukkan NIM: 123
Masukkan Nama: Iak
Iak berhasil masuk ke antrian.

--- MENU ANTRIAN KRS ---
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 1
Masukkan NIM: 124
Masukkan Nama: Rara
Rara berhasil masuk ke antrian.
```

```
--- MENU ANTRIAN KRS ---
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 1
Masukkan NIM: 125
Masukkan Nama: Zaskia
Zaskia berhasil masuk ke antrian.

--- MENU ANTRIAN KRS ---
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 3
Daftar mahasiswa dalam antrian:
NIM : 123, Nama : Iak
NIM : 124, Nama : Rara
NIM : 125, Nama : Zaskia
```

```
--- MENU ANTRIAN KRS ---
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 4
2 Mahasiswa terdepan:
NIM : 123, Nama : Iak
NIM : 124, Nama : Rara

--- MENU ANTRIAN KRS ---
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 5
Mahasiswa terakhir:
NIM : 125, Nama : Zaskia
```

```
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 8
Jumlah mahasiswa yang telah diproses: 2

--- MENU ANTRIAN KRS ---
1. Tambah Antrian
2. Panggil 2 Mahasiswa
3. Tampilkan Semua Antrian
4. Tampilkan 2 Terdepan
5. Tampilkan Antrian Terakhir
6. Kosongkan Antrian
7. Cetak Jumlah Antrian
8. Cetak Jumlah yang Sudah Diproses
9. Cetak Mahasiswa Belum Diproses
0. Keluar
Pilih menu: 9
Sisa mahasiswa yang belum diproses (maks 30): 28
```