

Nama : Muhammad Zakaria Haniya

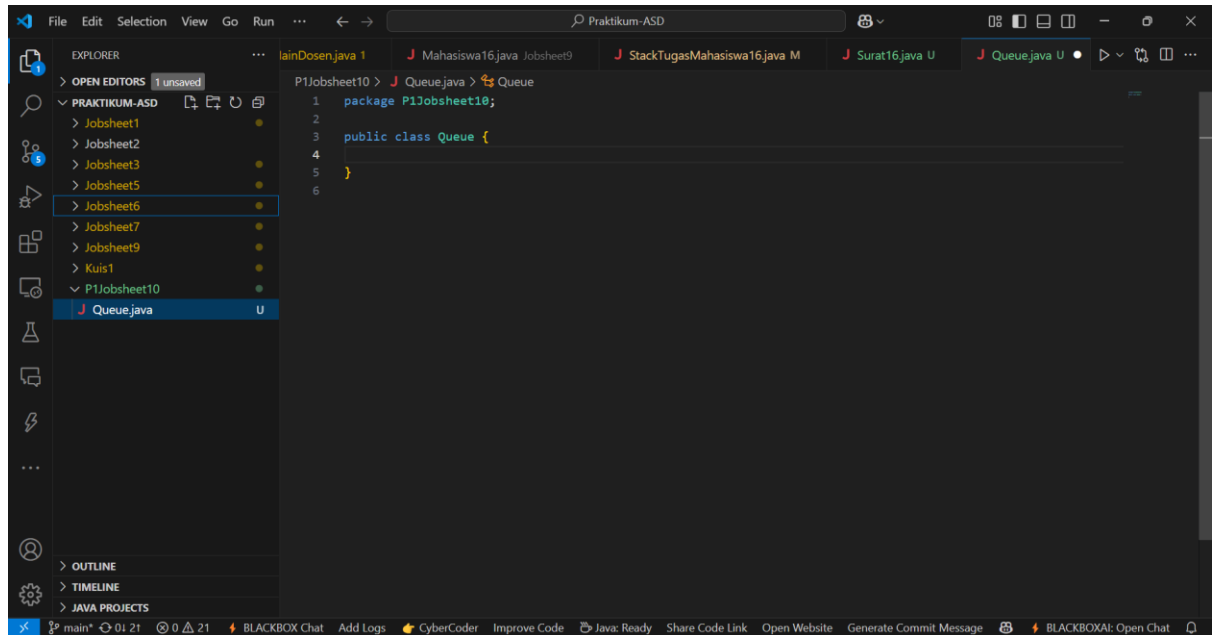
NIM : 244107020135

Kelas : TI-1B

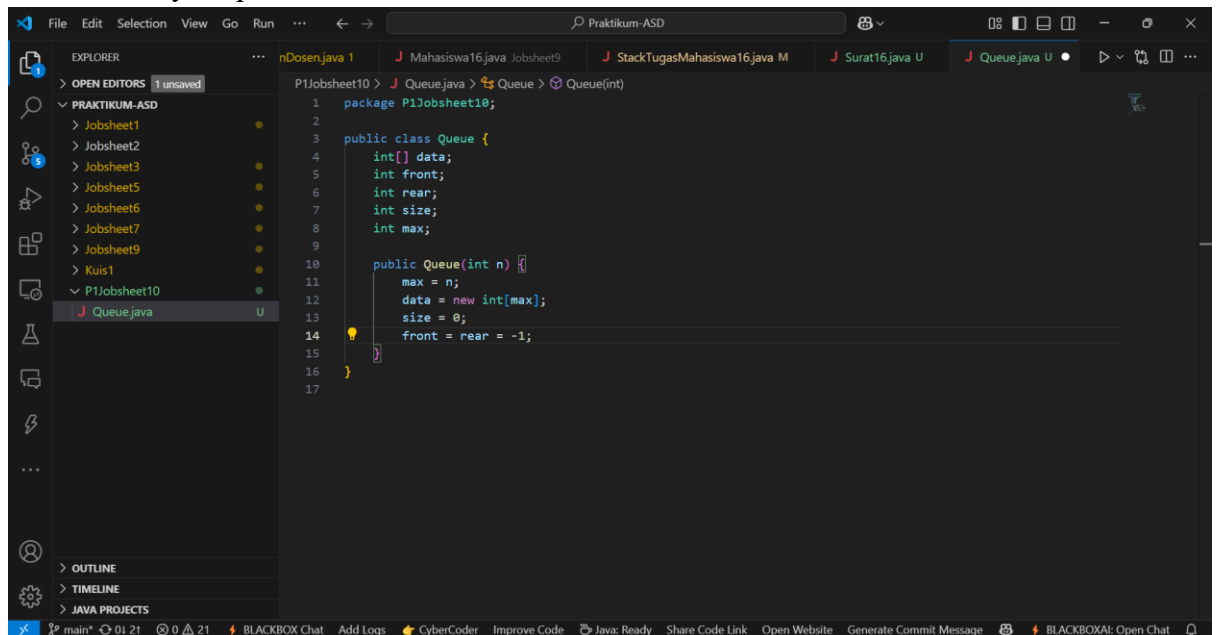
Absen : 16

Percobaan 1 : Operasi Dasar Queue

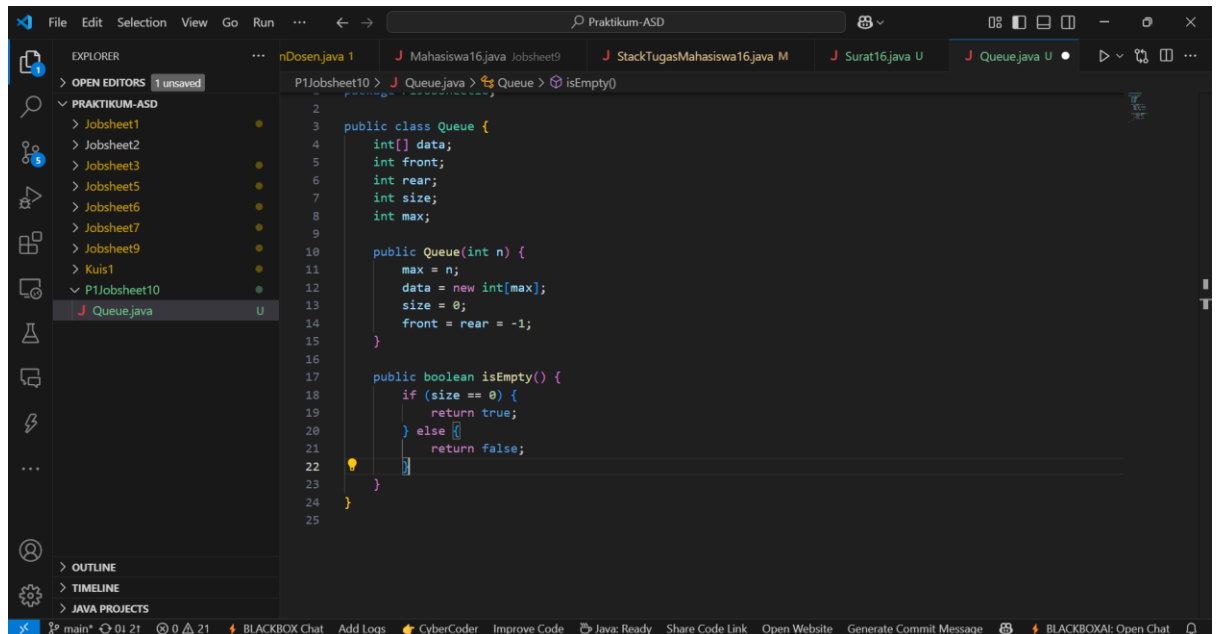
1. Buat folder baru bernama P1Jobsheet10 di dalam repository Praktikum ASD, kemudian buat class baru dengan nama Queue



2. Tambahkan atribut-atribut Queue sesuai diagram class, kemudian tambahkan pula konstruktornya seperti berikut ini



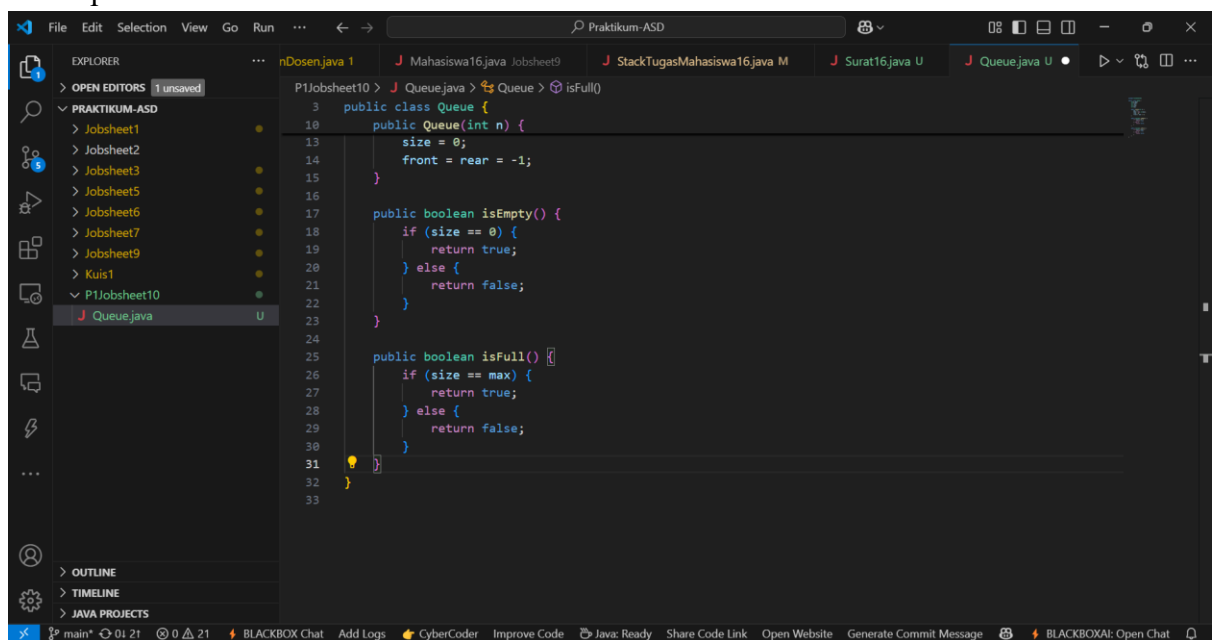
3. Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah queue kosong



The screenshot shows an IDE with the following code in `Queue.java`:

```
1 public class Queue {
2     int[] data;
3     int front;
4     int rear;
5     int size;
6     int max;
7
8     public Queue(int n) {
9         max = n;
10        data = new int[max];
11        size = 0;
12        front = rear = -1;
13    }
14
15    public boolean isEmpty() {
16        if (size == 0) {
17            return true;
18        } else {
19            return false;
20        }
21    }
22 }
23
24
25
```

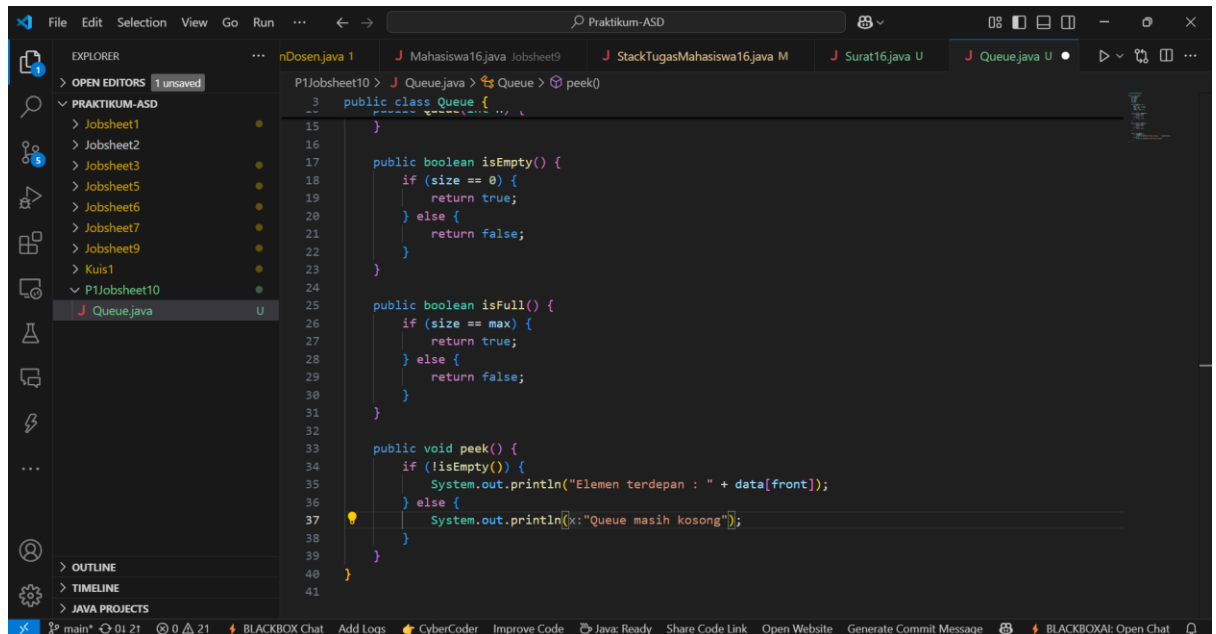
4. Buat method IsFull bertipe boolean yang digunakan untuk mengecek apakah queue sudah penuh



The screenshot shows an IDE with the following code in `Queue.java`:

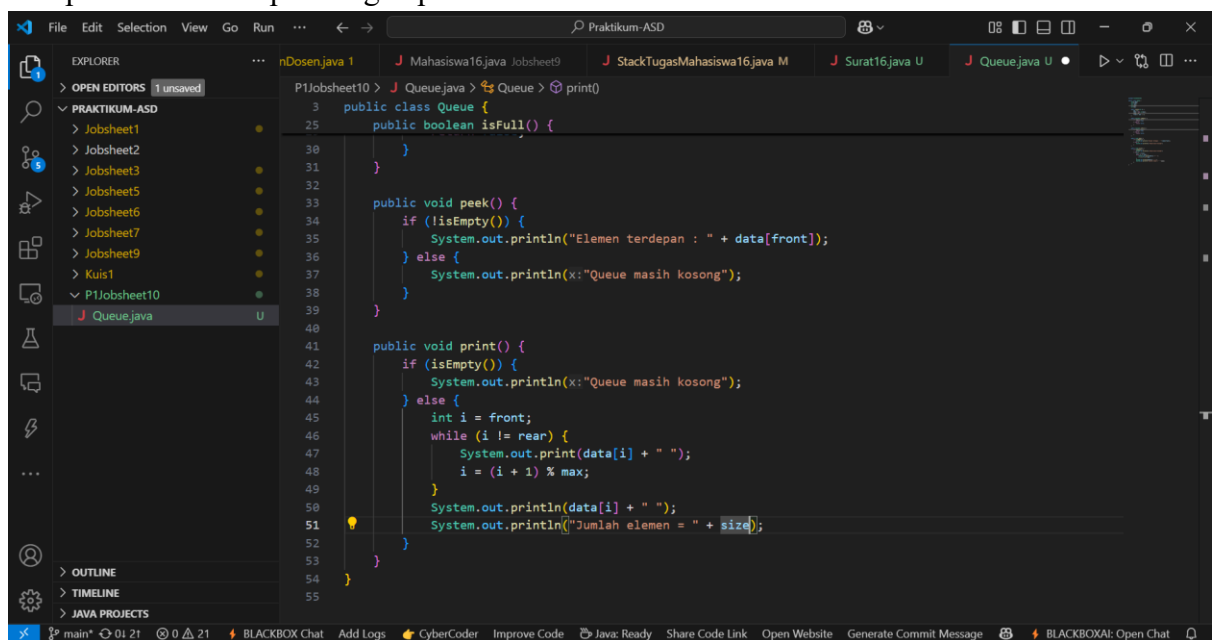
```
1 public class Queue {
2     public Queue(int n) {
3         size = 0;
4         front = rear = -1;
5     }
6
7     public boolean isEmpty() {
8         if (size == 0) {
9             return true;
10        } else {
11            return false;
12        }
13    }
14
15    public boolean isFull() {
16        if (size == max) {
17            return true;
18        } else {
19            return false;
20        }
21    }
22 }
23
24
25
```

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



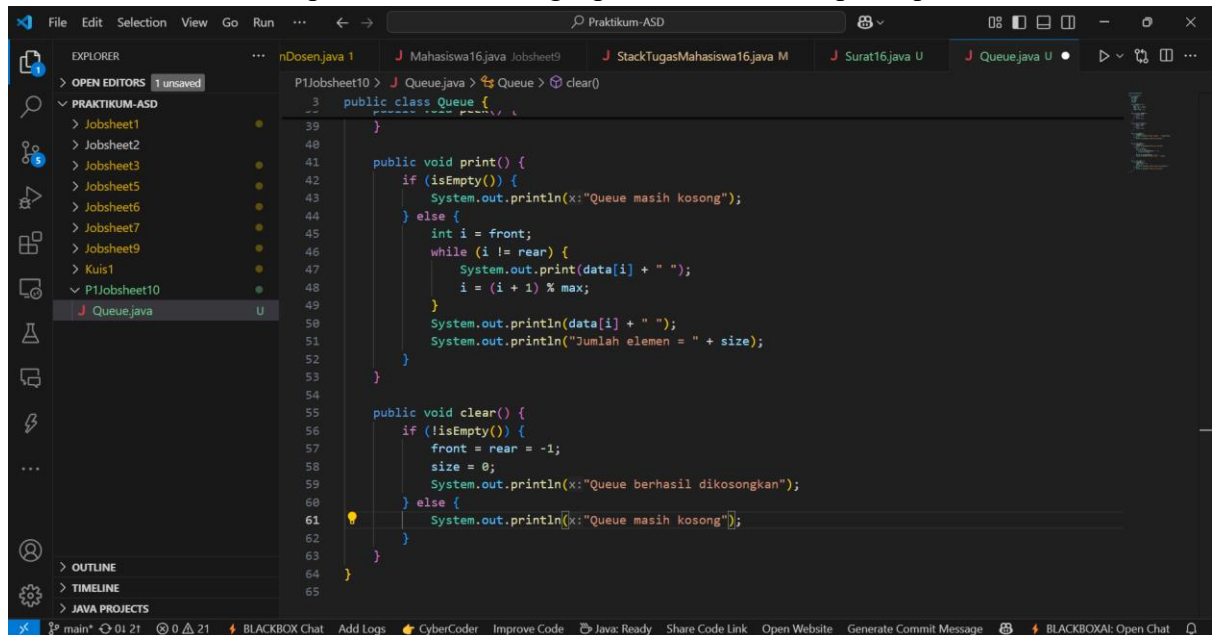
```
public class Queue {  
    3  
    15  
    16  
    17  
    18    public boolean isEmpty() {  
    19        if (size == 0) {  
    20            return true;  
    21        } else {  
    22            return false;  
    23        }  
    24    }  
    25  
    26    public boolean isFull() {  
    27        if (size == max) {  
    28            return true;  
    29        } else {  
    30            return false;  
    31        }  
    32    }  
    33  
    34    public void peek() {  
    35        if (!isEmpty()) {  
    36            System.out.println("Elemen terdepan : " + data[front]);  
    37        } else {  
    38            System.out.println(x:"Queue masih kosong");  
    39        }  
    40    }  
    41  
    42
```

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



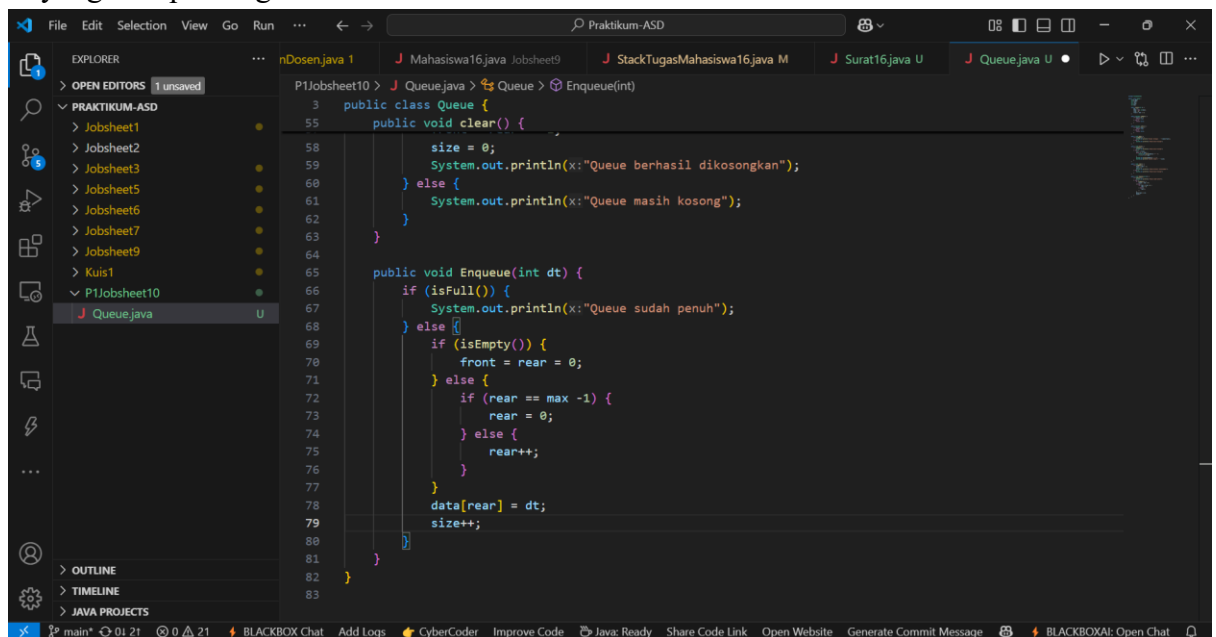
```
public class Queue {  
    3    public boolean isFull() {  
    25  
    30    }  
    31  
    32  
    33  
    34    public void peek() {  
    35        if (!isEmpty()) {  
    36            System.out.println("Elemen terdepan : " + data[front]);  
    37        } else {  
    38            System.out.println(x:"Queue masih kosong");  
    39        }  
    40    }  
    41  
    42    public void print() {  
    43        if (isEmpty()) {  
    44            System.out.println(x:"Queue masih kosong");  
    45        } else {  
    46            int i = front;  
    47            while (i != rear) {  
    48                System.out.print(data[i] + " ");  
    49                i = (i + 1) % max;  
    50            }  
    51            System.out.println(data[i] + " ");  
    52            System.out.println("Jumlah elemen = " + size);  
    53        }  
    54    }  
    55
```

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



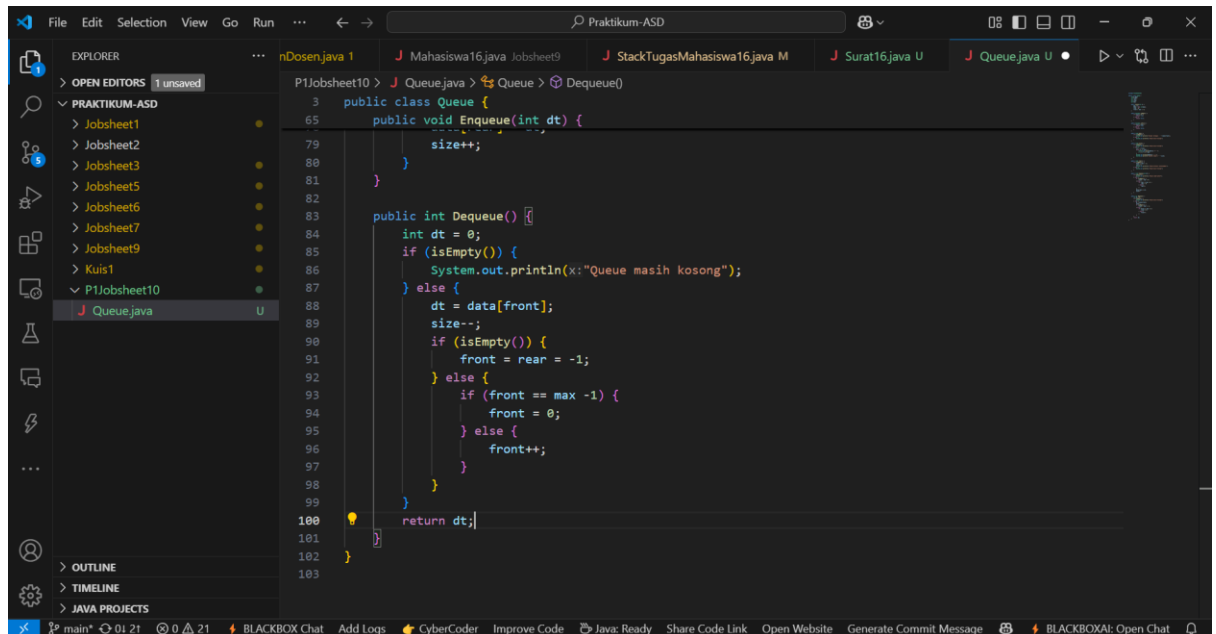
```
public class Queue {  
    ...  
    public void print() {  
        if (isEmpty()) {  
            System.out.println(x:"Queue masih kosong");  
        } else {  
            int i = front;  
            while (i != rear) {  
                System.out.print(data[i] + " ");  
                i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

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



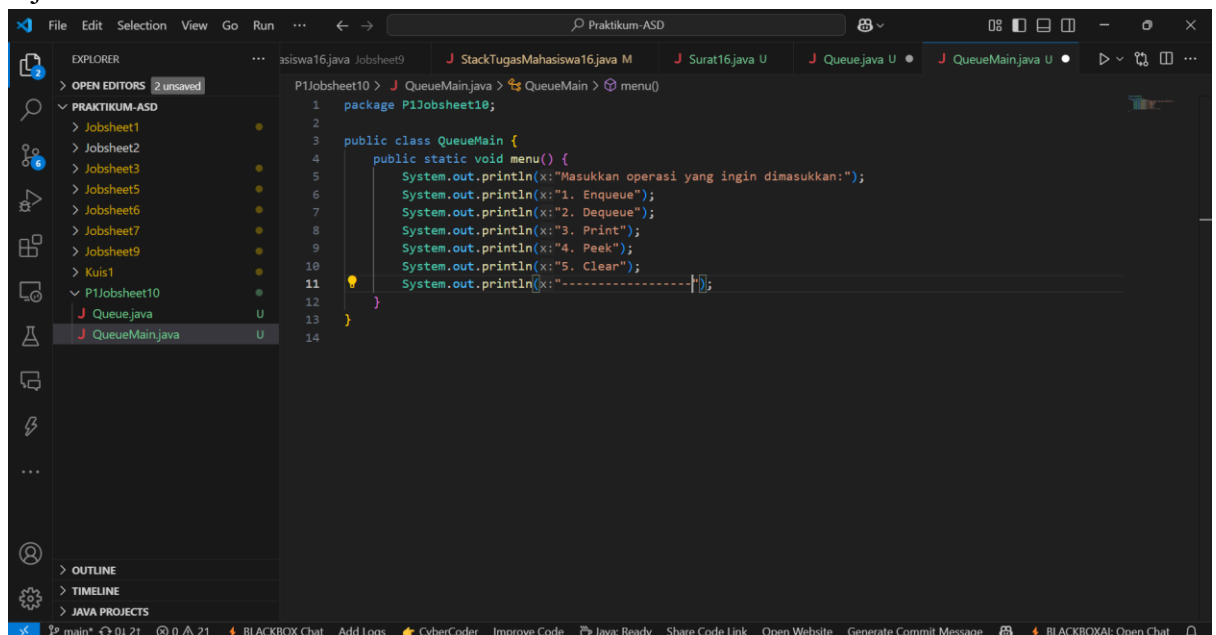
```
public class Queue {  
    ...  
    public void clear() {  
        size = 0;  
        System.out.println(x:"Queue berhasil dikosongkan");  
    } else {  
        System.out.println(x:"Queue masih kosong");  
    }  
    public void Enqueue(int dt) {  
        if (isFull()) {  
            System.out.println(x:"Queue sudah penuh");  
        } else {  
            if (isEmpty()) {  
                front = rear = 0;  
            } else {  
                if (rear == max - 1) {  
                    rear = 0;  
                } else {  
                    rear++;  
                }  
            }  
            data[rear] = dt;  
            size++;  
        }  
    }  
}
```

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



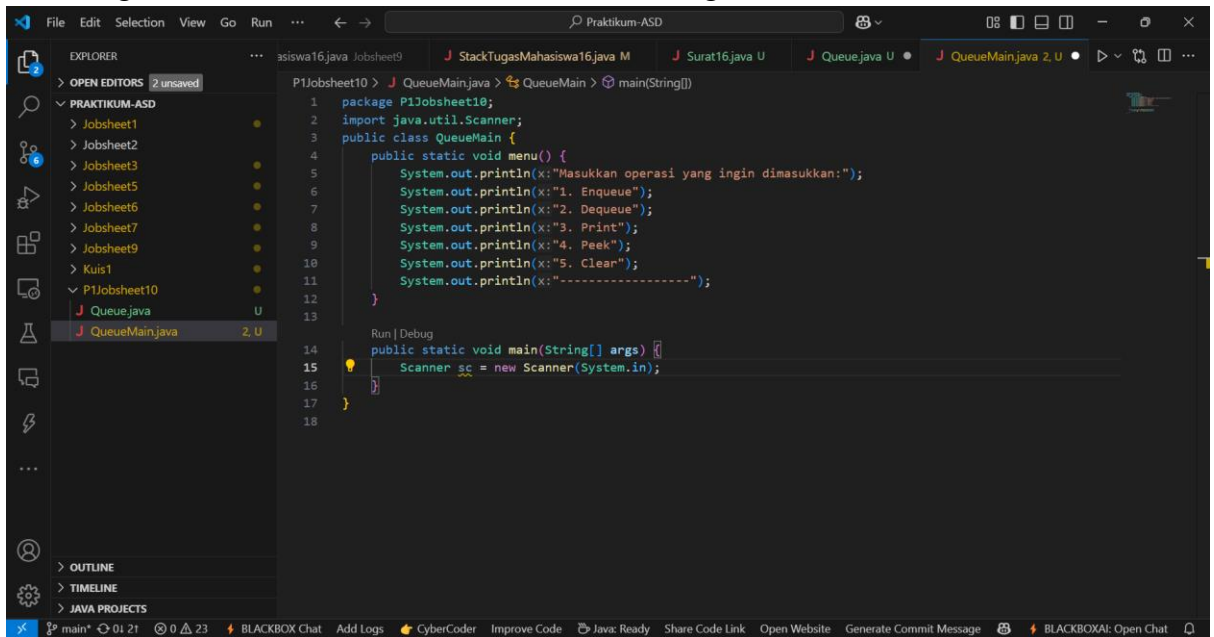
```
1 public class Queue {
2     ...
3     public void Enqueue(int dt) {
4         ...
5     }
6
7     public int Dequeue() {
8         int dt = 0;
9         if (isEmpty()) {
10             System.out.println("Queue masih kosong");
11         } else {
12             dt = data[front];
13             size--;
14             if (isEmpty()) {
15                 front = rear = -1;
16             } else {
17                 if (front == max - 1) {
18                     front = 0;
19                 } else {
20                     front++;
21                 }
22             }
23         }
24         return dt;
25     }
26 }
```

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



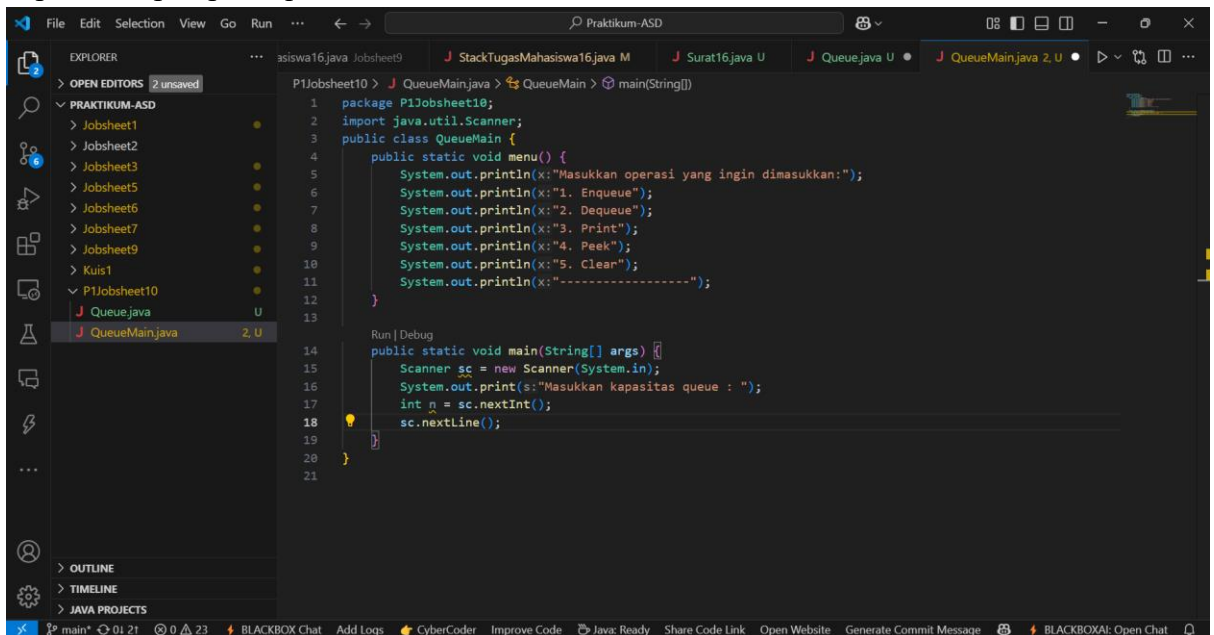
```
1 package P1Jobsheet10;
2
3 public class QueueMain {
4     public static void menu() {
5         System.out.println("Masukkan operasi yang ingin dimasukkan:");
6         System.out.println("1. Enqueue");
7         System.out.println("2. Dequeue");
8         System.out.println("3. Print");
9         System.out.println("4. Peek");
10        System.out.println("5. Clear");
11        System.out.println("-----");
12    }
13 }
14
```

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



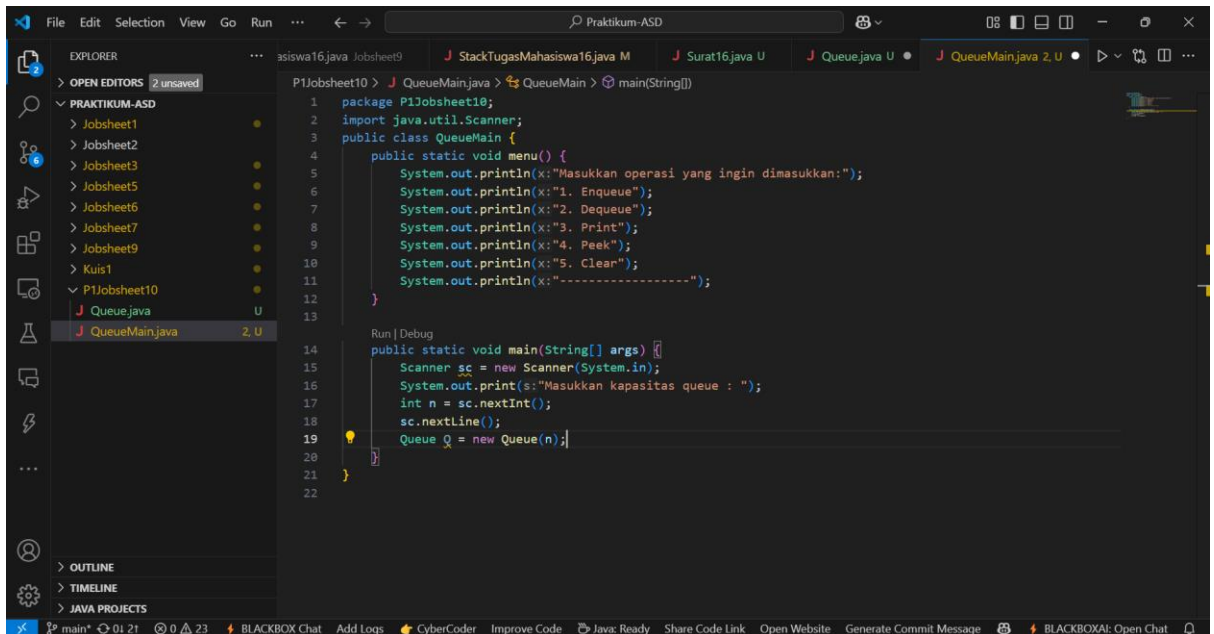
```
1 package P1Jobsheet10;
2 import java.util.Scanner;
3 public class QueueMain {
4     public static void menu() {
5         System.out.println(x:"Masukkan operasi yang ingin dimasukkan:");
6         System.out.println(x:"1. Enqueue");
7         System.out.println(x:"2. Dequeue");
8         System.out.println(x:"3. Print");
9         System.out.println(x:"4. Peek");
10        System.out.println(x:"5. Clear");
11        System.out.println(x:"-----");
12    }
13
14    Run | Debug
15    public static void main(String[] args) {
16        Scanner sc = new Scanner(System.in);
17    }
18 }
```

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



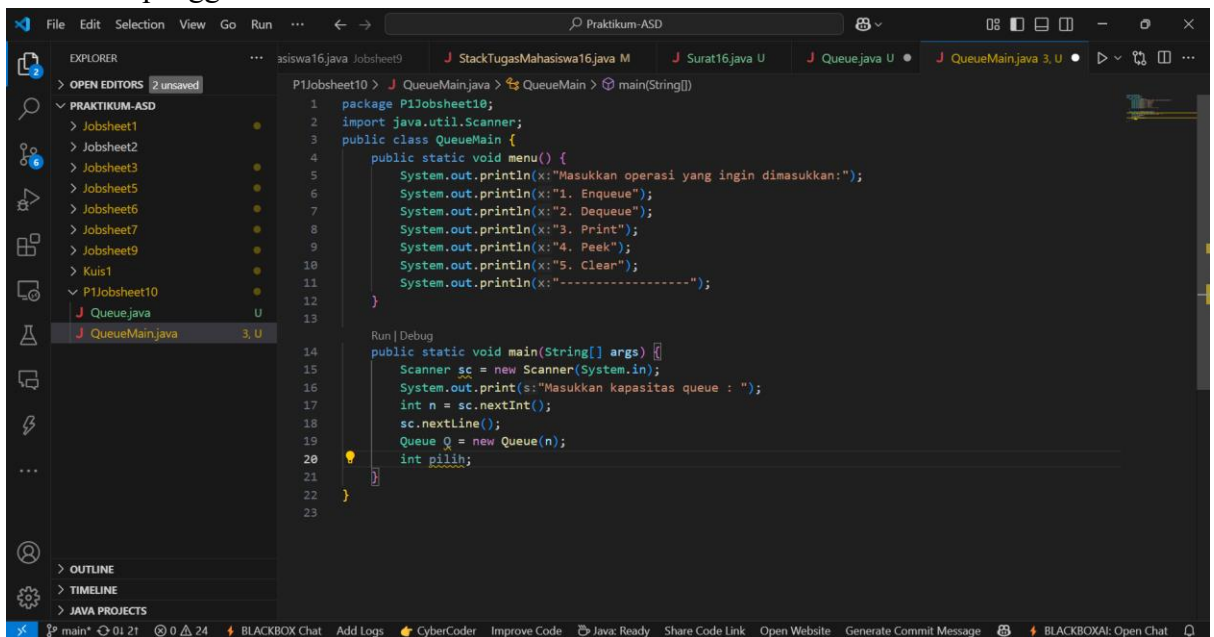
```
1 package P1Jobsheet10;
2 import java.util.Scanner;
3 public class QueueMain {
4     public static void menu() {
5         System.out.println(x:"Masukkan operasi yang ingin dimasukkan:");
6         System.out.println(x:"1. Enqueue");
7         System.out.println(x:"2. Dequeue");
8         System.out.println(x:"3. Print");
9         System.out.println(x:"4. Peek");
10        System.out.println(x:"5. Clear");
11        System.out.println(x:"-----");
12    }
13
14    Run | Debug
15    public static void main(String[] args) {
16        Scanner sc = new Scanner(System.in);
17        System.out.print(s:"Masukkan kapasitas queue : ");
18        int n = sc.nextInt();
19        sc.nextLine();
20    }
21 }
```

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



```
1 package P1Jobsheet10;
2 import java.util.Scanner;
3 public class QueueMain {
4     public static void menu() {
5         System.out.println("Masukkan operasi yang ingin dimasukkan:");
6         System.out.println("1. Enqueue");
7         System.out.println("2. Dequeue");
8         System.out.println("3. Print");
9         System.out.println("4. Peek");
10        System.out.println("5. Clear");
11        System.out.println("-----");
12    }
13
14    public static void main(String[] args) {
15        Scanner sc = new Scanner(System.in);
16        System.out.print("Masukkan kapasitas queue : ");
17        int n = sc.nextInt();
18        sc.nextLine();
19        Queue Q = new Queue(n);
20    }
21 }
22
```

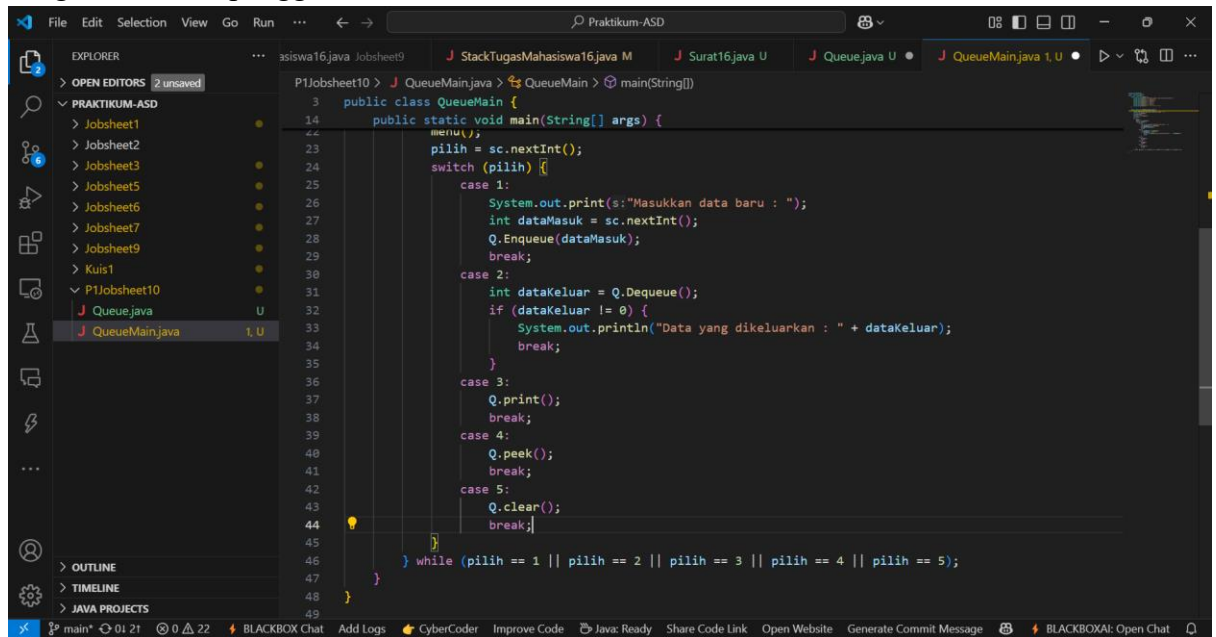
14. Deklarasikan variabel dengan nama pilih bertipe integer untuk menampung pilih menu dari pengguna



```
1 package P1Jobsheet10;
2 import java.util.Scanner;
3 public class QueueMain {
4     public static void menu() {
5         System.out.println("Masukkan operasi yang ingin dimasukkan:");
6         System.out.println("1. Enqueue");
7         System.out.println("2. Dequeue");
8         System.out.println("3. Print");
9         System.out.println("4. Peek");
10        System.out.println("5. Clear");
11        System.out.println("-----");
12    }
13
14    public static void main(String[] args) {
15        Scanner sc = new Scanner(System.in);
16        System.out.print("Masukkan kapasitas queue : ");
17        int n = sc.nextInt();
18        sc.nextLine();
19        Queue Q = new Queue(n);
20        int pilih;
21    }
22 }
23
```

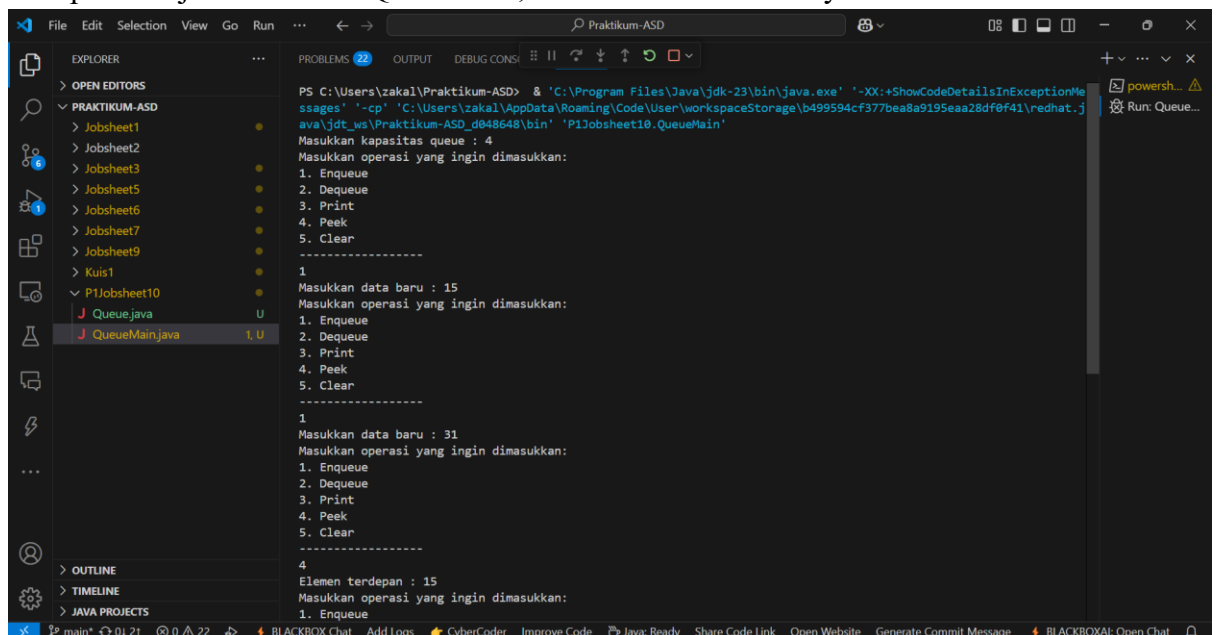
15. Lakukan perulangan menggunakan do-while untuk menjalankan program secara terus menerus sesuai masukan yang diberikan. Di dalam perulangan tersebut, terdapat pemilihan kondisi menggunakan switch-case untuk menjalankan operasi queue sesuai

dengan masukan pengguna



```
1 public class QueueMain {
2     public static void main(String[] args) {
3         menu();
4         pilih = sc.nextInt();
5         switch (pilih) {
6             case 1:
7                 System.out.print(s:"Masukkan data baru : ");
8                 int dataMasuk = sc.nextInt();
9                 Q.Enqueue(dataMasuk);
10                break;
11             case 2:
12                 int dataKeluar = Q.Dequeue();
13                 if (dataKeluar != 0) {
14                     System.out.println("Data yang dikeluarkan : " + dataKeluar);
15                 }
16                 break;
17             case 3:
18                 Q.print();
19                 break;
20             case 4:
21                 Q.peek();
22                 break;
23             case 5:
24                 Q.clear();
25                 break;
26         }
27     }
28 }
29 while (pilih == 1 || pilih == 2 || pilih == 3 || pilih == 4 || pilih == 5);
```

16. Compile dan jalankan class QueueMain, kemudian amati hasilnya



```
PS C:\Users\zakal\Praktikum-ASD> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\zakal\AppData\Roaming\Code\User\workspaceStorage\b499594cf377bea8a9195eaa28df0f41\redhat.j...
ava\jdt_ws\Praktikum-ASD_d048648\bin\' 'P1Jobsheet10.QueueMain'
Masukkan kapasitas queue : 4
Masukkan operasi yang ingin dimasukkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
1
Masukkan data baru : 15
Masukkan operasi yang ingin dimasukkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
1
Masukkan data baru : 31
Masukkan operasi yang ingin dimasukkan:
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
-----
4
Elemen terdepan : 15
Masukkan operasi yang ingin dimasukkan:
1. Enqueue
```

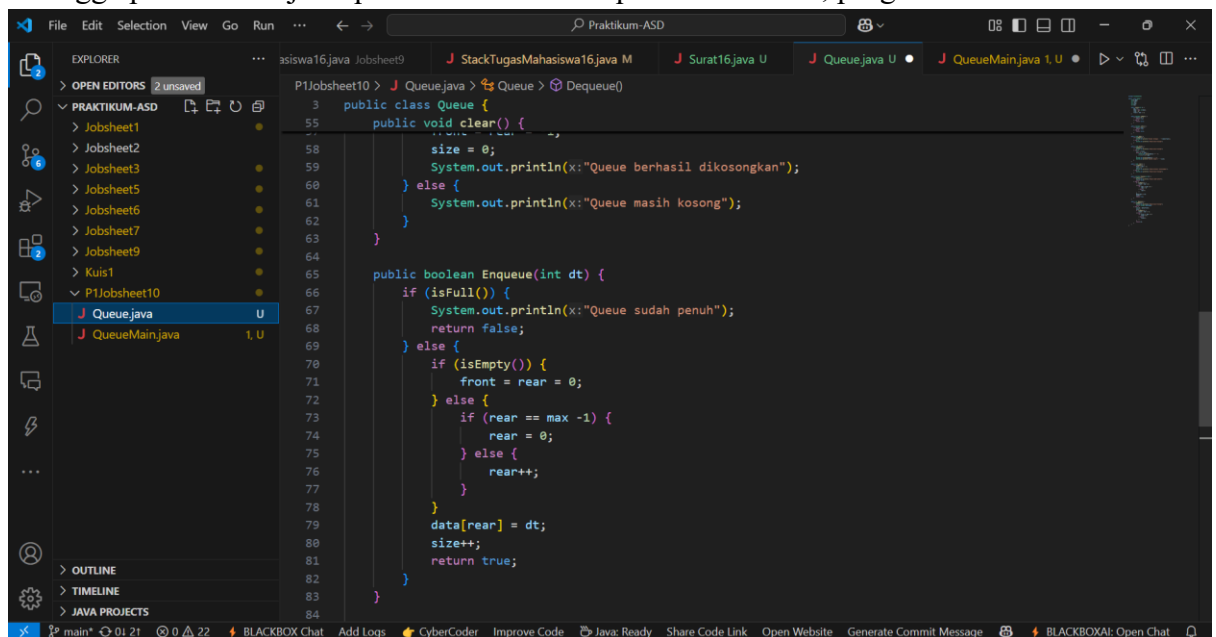
Pertanyaan

1. Pada konstruktor, mengapa nilai awal atribut front dan rear bernilai -1, sementara atribut size bernilai 0?
 - size bernilai 0 karena array masih kosong
 - front dan rear bernilai -1 karena tidak menunjuk ke data manapun
2. Pada method Enqueue, jelaskan maksud dan kegunaan dari potongan kode berikut!
 - Potongan kode tersebut mengatur agar rear kembali ke indeks 0 saat mencapai ujung array (max - 1), agar queue bisa menggunakan ruang kosong di awal array
3. Pada method Dequeue, jelaskan maksud dan kegunaan dari potongan kode berikut!

- Potongan kode ini memungkinkan pergerakan front secara melingkar, agar queue bisa mengakses elemen yang berada di awal array setelah elemen-elemen sebelumnya dikeluarkan
4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (int i=0), melainkan int i=front?
 - Karena elemen pertama dalam queue tidak selalu berada di indeks 0
 5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut!
 - $i = (i + 1) \% \text{max}$; maksud dari program tersebut adalah jika $(i + 1) \% \text{max} = 0$ maka i akan mengulang dari indeks awal dan menghindari indeks keluar batas Array
 6. Tunjukkan potongan kode program yang merupakan queue overflow!

```
public void Enqueue(int dt) {
    if (isFull()) {
        System.out.println(x:"Queue sudah penuh");
    }
}
```

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!

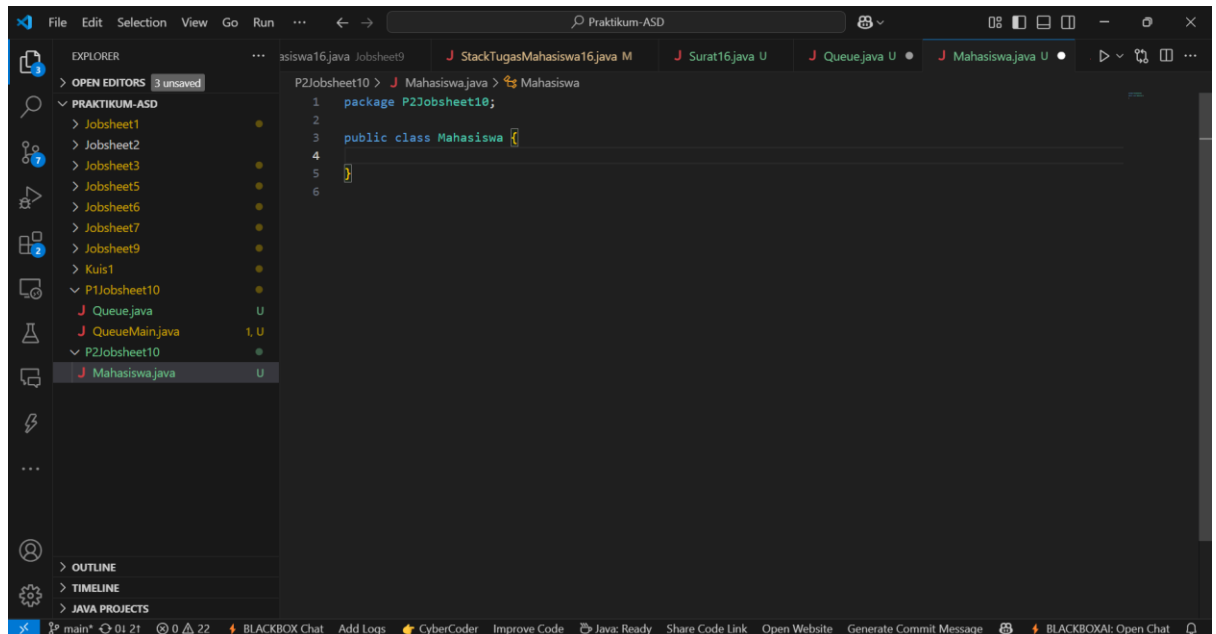


```
public class Queue {
    public void clear() {
        size = 0;
        System.out.println(x:"Queue berhasil dikosongkan");
    } else {
        System.out.println(x:"Queue masih kosong");
    }
}

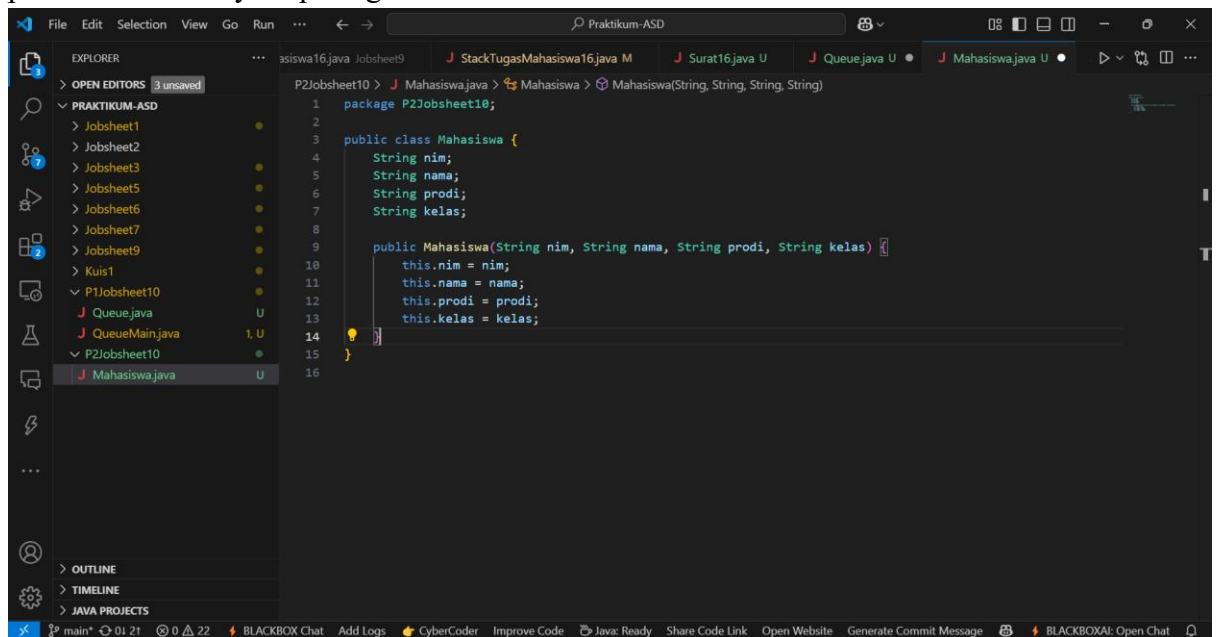
public boolean Enqueue(int dt) {
    if (isFull()) {
        System.out.println(x:"Queue sudah penuh");
        return false;
    } else {
        if (isEmpty()) {
            front = rear = 0;
        } else {
            if (rear == max - 1) {
                rear = 0;
            } else {
       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

Percobaan 2 : Antrian Layanan Akademik

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.



Dan tambahkan method tampilkanData berikut :

```

1 package P2Jobsheet10;
2
3 public class Mahasiswa {
4     String nim;
5     String nama;
6     String prodi;
7     String kelas;
8
9     public Mahasiswa(String nim, String nama, String prodi, String kelas) {
10         this.nim = nim;
11         this.nama = nama;
12         this.prodi = prodi;
13         this.kelas = kelas;
14     }
15
16     public void tampilkanData() {
17         System.out.println(nim + " - " + nama + " - " + prodi + " - " + kelas);
18     }
19 }
20

```

- 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.

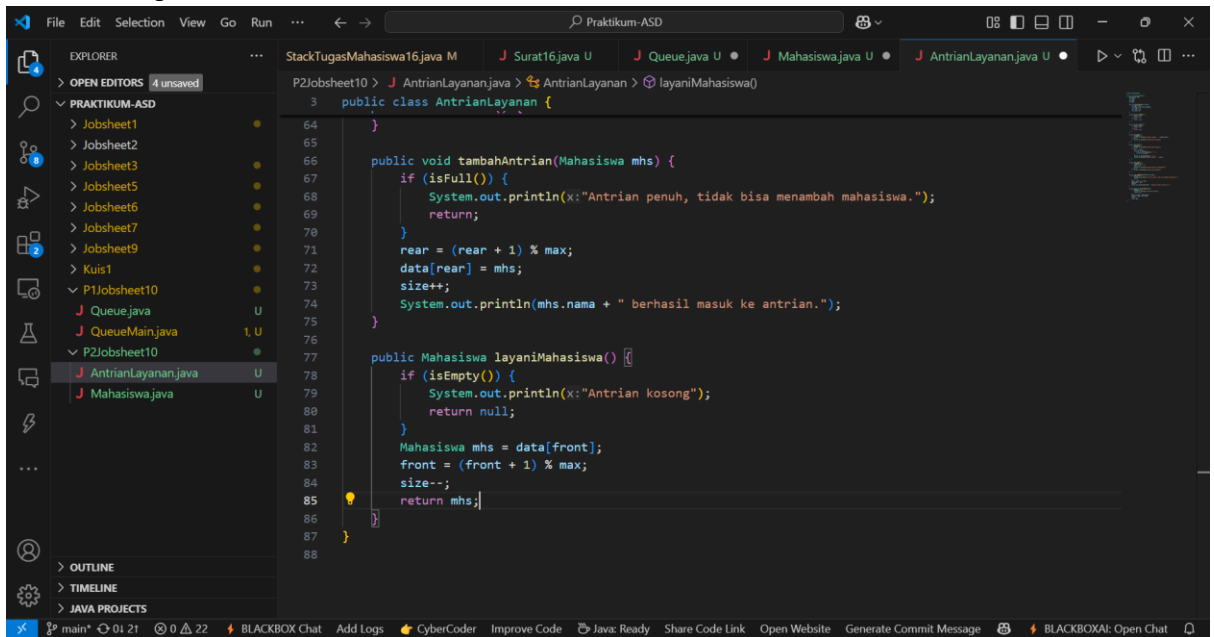
```

1 package P2Jobsheet10;
2
3 public class AntrianLayanan {
4     Mahasiswa[] data;
5     int front;
6     int rear;
7     int size;
8     int max;
9
10    public AntrianLayanan(int max) {
11        this.max = max;
12        this.data = new Mahasiswa[max];
13        this.front = 0;
14        this.rear = -1;
15        this.size = 0;
16    }
17
18    public boolean isEmpty() {
19        if (size == 0) {
20            return true;
21        } else {
22            return false;
23        }
24    }
25
26    public boolean isFull() {
27        if (size == max) {
28            return true;
29        } else {
30            return false;
31        }
32    }
33 }
34

```

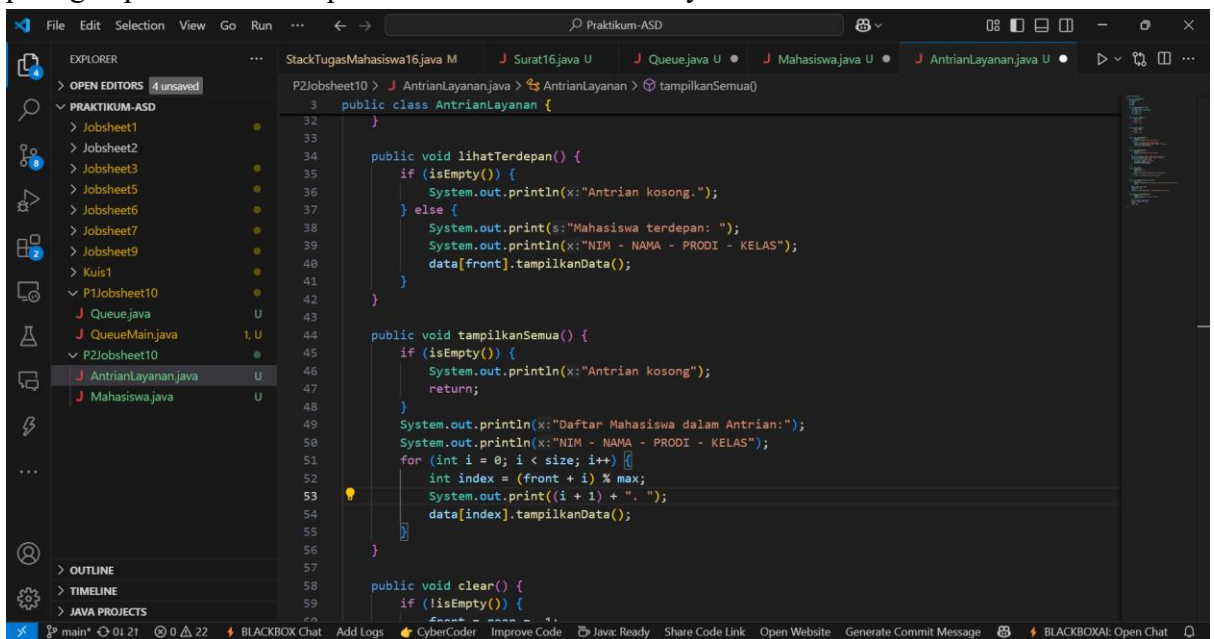
- Lakukan modifikasi pada class AntrianLayanan dengan mengubah tipe int[] data menjadi Mahasiswa[] data karena pada kasus ini data yang akan disimpan berupa object Mahasiswa. Modifikasi perlu dilakukan pada atribut, method Enqueue, dan

method Dequeue.



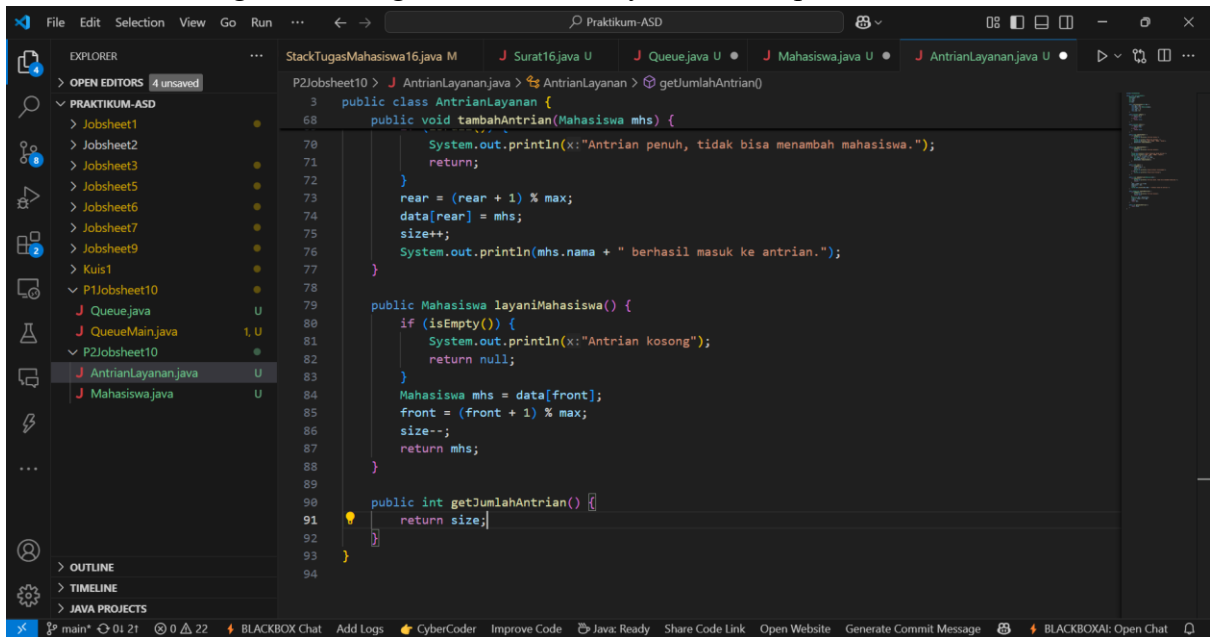
```
StackTugasMahasiswa16.java M J Surat16.java U J Queue.java U J Mahasiswa.java U J AntrianLayanan.java U
P2Jobsheet10 > J AntrianLayanan.java > AntrianLayanan > layaniMahasiswa()
3 public class AntrianLayanan {
64 }
65
66 public void tambahAntrian(Mahasiswa mhs) {
67     if (isEmpty()) {
68         System.out.println(x:"Antrian penuh, tidak bisa menambah mahasiswa.");
69         return;
70     }
71     rear = (rear + 1) % max;
72     data[rear] = mhs;
73     size++;
74     System.out.println(mhs.nama + " berhasil masuk ke antrian.");
75 }
76
77 public Mahasiswa layaniMahasiswa() {
78     if (isEmpty()) {
79         System.out.println(x:"Antrian kosong");
80         return null;
81     }
82     Mahasiswa mhs = data[front];
83     front = (front + 1) % max;
84     size--;
85     return mhs;
86 }
87 }
88 }
```

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



```
StackTugasMahasiswa16.java M J Surat16.java U J Queue.java U J Mahasiswa.java U J AntrianLayanan.java U
P2Jobsheet10 > J AntrianLayanan.java > AntrianLayanan > tampilkanSemua()
3 public class AntrianLayanan {
32 }
33
34 public void lihatTerdepan() {
35     if (isEmpty()) {
36         System.out.println(x:"Antrian kosong.");
37     } else {
38         System.out.print(s:"Mahasiswa terdepan: ");
39         System.out.println(x:"NIM - NAMA - PRODI - KELAS");
40         data[front].tampilkanData();
41     }
42 }
43
44 public void tampilkanSemua() {
45     if (isEmpty()) {
46         System.out.println(x:"Antrian kosong");
47         return;
48     }
49     System.out.println(x:"Daftar Mahasiswa dalam Antrian:");
50     System.out.println(x:"NIM - NAMA - PRODI - KELAS");
51     for (int i = 0; i < size; i++) {
52         int index = (front + i) % max;
53         System.out.print((i + 1) + ". ");
54         data[index].tampilkanData();
55     }
56 }
57
58 public void clear() {
59     if (isEmpty()) {
60         front = rear = 1;
61     }
62 }
```

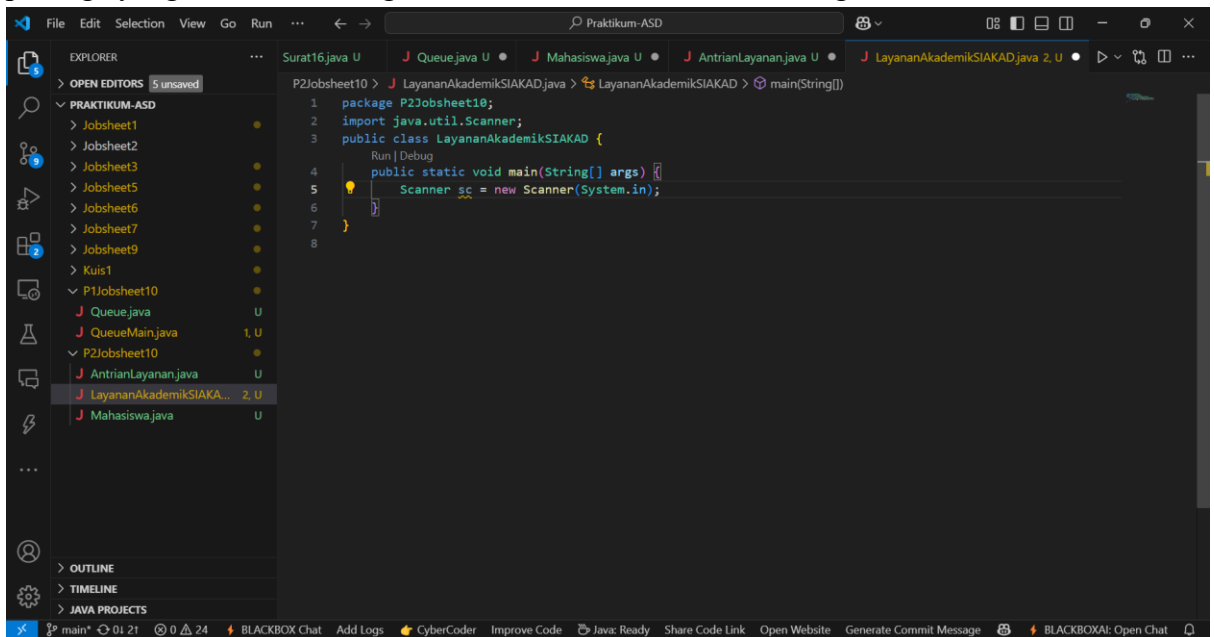
Ditambahkan dengan method `getJumlahAntrian` yaitu menampilkan nilai `size`



The screenshot shows the IDE with the `AntrianLayanan` class open. The `getJumlahAntrian()` method is implemented as follows:

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

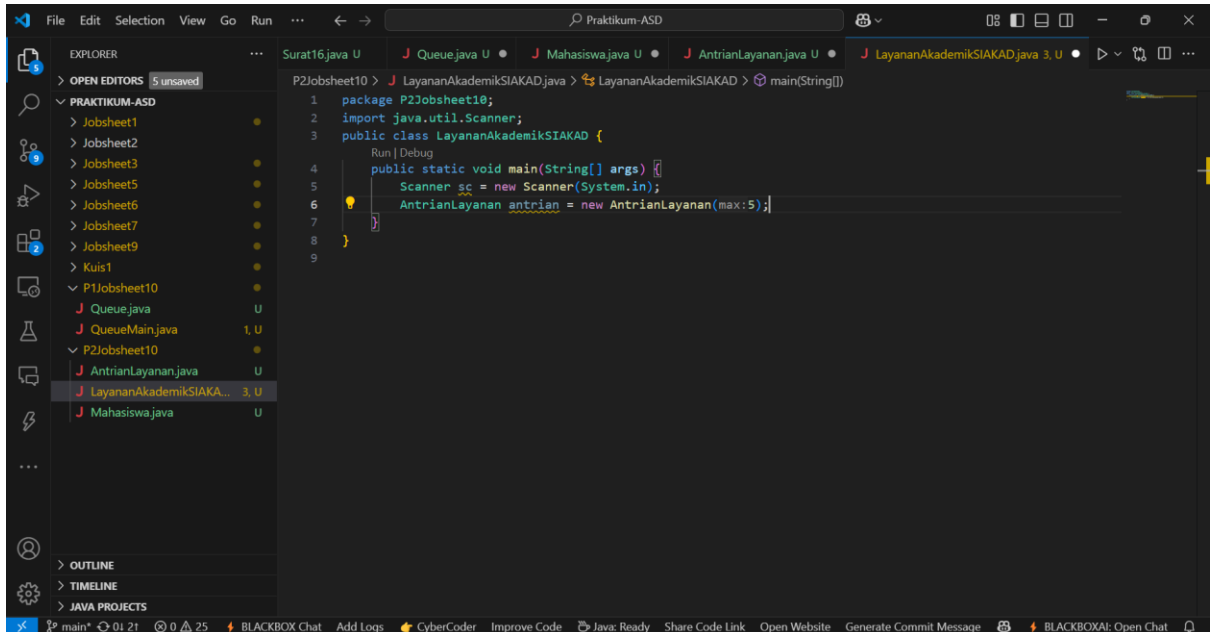
- Selanjutnya, buat class baru dengan nama `LayananAkademikSIKAD` tetap pada package yang sama. Buat fungsi main, deklarasikan `Scanner` dengan nama `sc`.



The screenshot shows the IDE with the `LayananAkademikSIKAD` class open. The `main` method is implemented as follows:

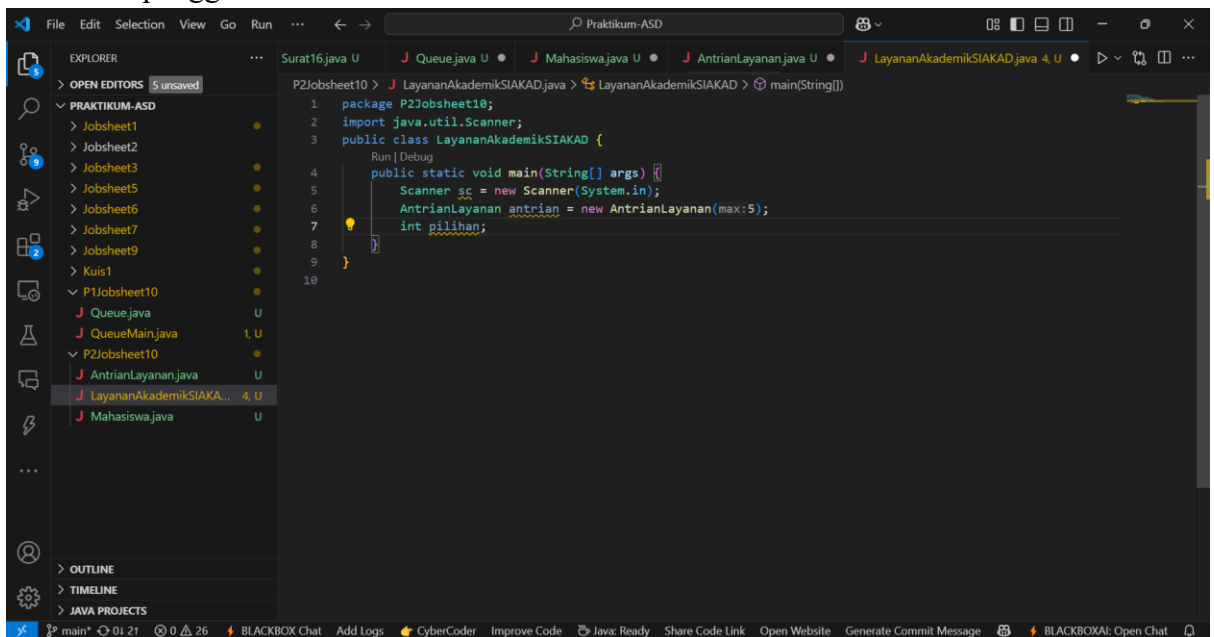
```
package P2Jobsheet10;  
import java.util.Scanner;  
public class LayananAkademikSIKAD {  
    Run | Debug  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
    }  
}
```

7. Kemudian lakukan instansiasi objek AntrianLayanan dengan nama antrian dan nilai parameternya adalah nilai maksimal antrian yang ditentukan (misal sama dengan 5).



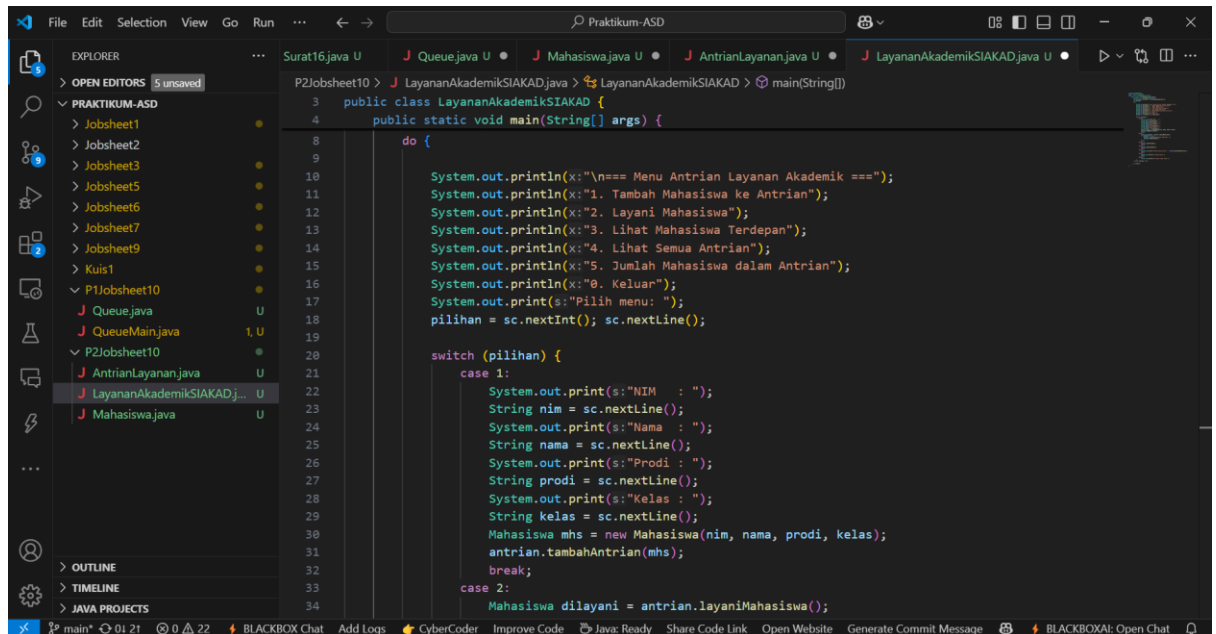
```
1 package P2Jobsheet10;
2 import java.util.Scanner;
3 public class LayananAkademikSIKAD {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         AntrianLayanan antrian = new AntrianLayanan(max:5);
7     }
8 }
9
```

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



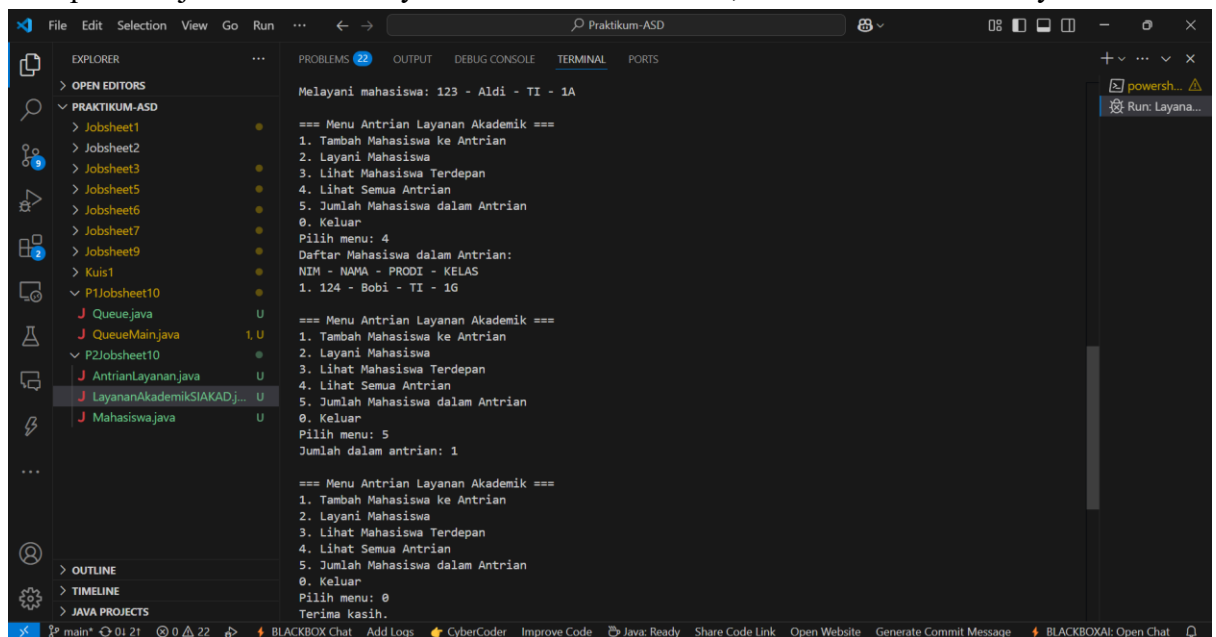
```
1 package P2Jobsheet10;
2 import java.util.Scanner;
3 public class LayananAkademikSIKAD {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6         AntrianLayanan antrian = new AntrianLayanan(max:5);
7         int pilihan;
8     }
9 }
10
```

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



```
3 public class LayananAkademikSIKAD {
4     public static void main(String[] args) {
5         do {
6             System.out.println(x:"\n== Menu Antrian Layanan Akademik ==");
7             System.out.println(x:"1. Tambah Mahasiswa ke Antrian");
8             System.out.println(x:"2. Layani Mahasiswa");
9             System.out.println(x:"3. Lihat Mahasiswa Terdepan");
10            System.out.println(x:"4. Lihat Semua Antrian");
11            System.out.println(x:"5. Jumlah Mahasiswa dalam Antrian");
12            System.out.println(x:"0. Keluar");
13            System.out.print(s:"Pilih menu: ");
14            pilihan = sc.nextInt(); sc.nextLine();
15
16            switch (pilihan) {
17                case 1:
18                    System.out.print(s:"NIM : ");
19                    String nim = sc.nextLine();
20                    System.out.print(s:"Nama : ");
21                    String nama = sc.nextLine();
22                    System.out.print(s:"Prodi : ");
23                    String prodi = sc.nextLine();
24                    System.out.print(s:"Kelas : ");
25                    String kelas = sc.nextLine();
26                    Mahasiswa mhs = new Mahasiswa(nim, nama, prodi, kelas);
27                    antrian.tambahAntrian(mhs);
28                    break;
29                case 2:
30                    Mahasiswa dilayani = antrian.layaniMahasiswa();
31            }
32        }
33    }
34 }
```

10. Compile dan jalankan class LayananAkademikSIKAD, kemudian amati hasilnya.



```
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 LayananAkademikSIKAD sehingga method LihatAkhir dapat dipanggil!

```

3 public class AntrianLayanan {
79     public Mahasiswa layaniMahasiswa() {
81         System.out.println(x:"Antrian kosong");
82         return null;
83     }
84     Mahasiswa mhs = data[front];
85     front = (front + 1) % max;
86     size--;
87     return mhs;
88 }
89
90 public int getJumlahAntrian() {
91     return size;
92 }
93
94 public void lihatAkhir() {
95     if (isEmpty()) {
96         System.out.println(x:"Antrian kosong.");
97     } else {
98         System.out.print(s:"Mahasiswa antrian belakang: ");
99         System.out.println(x:"NIM - NAMA - PRODI - KELAS");
100        data[rear].tampilkanData();
101    }
102 }
103 }
104

```

Tugas

1. Buatlah program antrian untuk mengilustrasikan 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.

AntrianKRS16
data:Mahasiswa[] front:int rear:int size:int max:int telahKRS:int jmlMhs = 30:int
AntrianKRS16(int max) boolean isEmpty() boolean isFull()

```

void tambahAntrian(Mahasiswa mhs)
Mahasiswa layaniMahasiswa()
void lihat2Terdepan()
void tampilkanSemua()
int getJumlahAntrian()
void lihatTerbelakang()
int jmlSudahKRS()
int jmlBelumKRS()
void kosongkanAntrian()

```

```

=== 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
1001 - Bisma - TI - 1B
1002 - Badawi - 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
1010 - Hikmal - 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. 1001 - Bisma - TI - 1B
2. 1002 - Badawi - TI - 1B
3. 1003 - Fiqa - TI - 1B
4. 1004 - Dicky - TI - 1B
5. 1005 - Geraldi - TI - 1B
6. 1006 - Angga - TI - 1B
7. 1007 - Abi - TI - 1B
8. 1008 - Hafiz - TI - 1B
9. 1009 - Farrel - TI - 1B
10. 1010 - Hikmal - 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 Antrian 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 melakukan 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 melakukan 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: 1001 - Bisma - TI - 1B
1002 - Badawi - 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
1003 - Fiqa - TI - 1B
1004 - Dicky - 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. 1003 - Fiqa - TI - 1B
2. 1004 - Dicky - TI - 1B
3. 1005 - Geraldi - TI - 1B
4. 1006 - Angga - TI - 1B
5. 1007 - Abi - TI - 1B
6. 1008 - Hafiz - TI - 1B
7. 1009 - Farrel - TI - 1B
8. 1010 - Hikmal - 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 Antrian 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 melakukan 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 melakukan KRS: 28