

LAPORAN PRAKTIKUM STRUKTUR DATA

PERTEMUAN KE-6



Disusun Oleh :

NAMA : Raden Isnawan Argi Aryasatya

NIM : 195410257

JURUSAN : Teknik Informatika

JENJANG : S1

Laboratorium Terpadu

Sekolah Tinggi Management Informatika Komputer

AKAKOM

YOGYAKARTA

2020

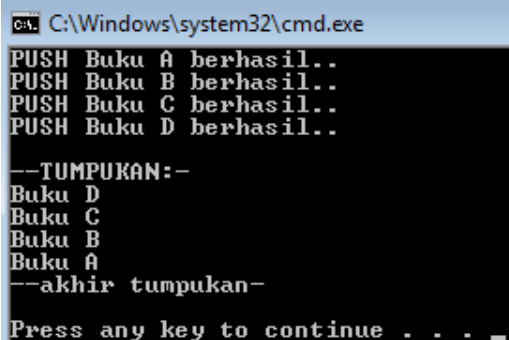
PERTEMUAN KE-6 **(PENGELOLAAN DATA PADA ARRAY/ LARIK: PENCARIAN** **DATA (SEARCHING))**

TUJUAN

Mahasiswa dapat mengimplementasikan tumpukan dan antrian untuk berbagai keperluan dengan menggunakan bahasa pemrograman Java

PRAKTIK 1

```
1 public class program_tumpukan{
2     public static int N = 5;
3     public static int atas = -1;
4     public static void PUSH (String tumpukan[], String data){
5         if (atas == N-1) //jika tumpukan penuh
6         {
7             System.out.println("maap, tumpukan penuh, PUSH " + data+ " tidak dapat dilakukan");
8         }
9         else //jika tumpukan tidak penuh
10        {
11            atas = atas + 1;
12            tumpukan[atas] = data;
13            System.out.println("PUSH " + data + " berhasil..");
14        }
15    }
16    public static String POP (String tumpukan[]){
17        String hasil;
18        if (atas < 0 ) //jika tumpukan kosong
19        {
20            hasil = "TUMPUKAN KOSONG, POP GAGAL DILAKUKAN";
21        }
22        else //jika tumpukan tidak kosong
23        {
24            hasil = tumpukan[atas];
25            atas = atas - 1;
26        }
27        return (hasil);
28    }
29    public static void lihatTumpukan(String tumpukan[]){
30        System.out.println("");
31        System.out.println("--TUMPUKAN:-");
32        for (int i=atas; i>=0; i--){
33            System.out.println(tumpukan[i]);
34        }
35        System.out.println("--akhir tumpukan-");
36        System.out.println("");
37    }
38    public static void main (String[] args){
39        String tumpukan[] = new String[4];
40        PUSH (tumpukan, "Buku A");
41        PUSH (tumpukan, "Buku B");
42        PUSH (tumpukan, "Buku C");
43        PUSH (tumpukan, "Buku D");
44        lihatTumpukan(tumpukan);
45    }
46 }
```



```
C:\Windows\system32\cmd.exe
PUSH Buku A berhasil..
PUSH Buku B berhasil..
PUSH Buku C berhasil..
PUSH Buku D berhasil..

--TUMPUKAN:-
Buku D
Buku C
Buku B
Buku A
--akhir tumpukan-

Press any key to continue . . . _
```

penjelasan: Buku apa sajakah yang dapat di push ke dalam tumpukan? Apakah buku A, B, C, D, keempatnya dapat dipush ke dalam tumpukan? Buku yang dapat dipush ke dalam tumpukan adalah buku A,B,C,D. Keempatnya dapat dipush ke dalam tumpukan karena masih dibawah nilai dari N yaitu 5. Lalu berada di posisi manakah “atas”? “atas” berada di posisi setelah buku D yaitu posisi terluar dari sebuah tumpukan. Posisi “atas” awalnya berada di bagian paling bawah. Namun seiring banyaknya data yang dipush maka “atas” terus terdorong ke luar tumpukan atau ke atas.

PRAKTIK 2

```
1 public class program_tumpukan{
2     public static int N = 5;
3     public static int atas = -1;
4     public static void PUSH (String tumpukan[], String data){
5         if (atas == N-1) //jika tumpukan penuh
6         {
7             System.out.println("maap, tumpukan penuh, PUSH " + data+ " tidak dapat dilakukan");
8         }
9         else //jika tumpukan tidak penuh
10        {
11            atas = atas + 1;
12            tumpukan[atas] = data;
13            System.out.println("PUSH " + data + " berhasil..");
14        }
15    }
16    public static String POP (String tumpukan[]){
17        String hasil;
18        if (atas < 0 ) //jika tumpukan kosong
19        {
20            hasil = "TUMPUKAN KOSONG, POP GAGAL DILAKUKAN";
21        }
22        else //jika tumpukan tidak kosong
23        {
24            hasil = tumpukan[atas];
25            atas = atas - 1;
26        }
27        return (hasil);
28    }
29    public static void lihatTumpukan(String tumpukan[]){
30        System.out.println("");
31        System.out.println("--TUMPUKAN:--");
32        for (int i=atas; i>=0; i--){
33            System.out.println(tumpukan[i]);
34        }
35        System.out.println("--akhir tumpukan--");
36        System.out.println("");
37    }
38    public static void main (String[] args){
39        String tumpukan[] = new String[7];
40        PUSH (tumpukan, "Buku A");
41        PUSH (tumpukan, "Buku B");
42        PUSH (tumpukan, "Buku C");
43        PUSH (tumpukan, "Buku D");
44        PUSH (tumpukan, "Buku E");
45        PUSH (tumpukan, "Buku F");
46        PUSH (tumpukan, "Buku G");
47        lihatTumpukan(tumpukan);
48    }
49 }
```

ca. C:\Windows\system32\cmd.exe

```
PUSH Buku A berhasil..
PUSH Buku B berhasil..
PUSH Buku C berhasil..
PUSH Buku D berhasil..
PUSH Buku E berhasil..
maap, tumpukan penuh, PUSH Buku F tidak dapat dilakukan
maap, tumpukan penuh, PUSH Buku G tidak dapat dilakukan

--TUMPUKAN:--
Buku E
Buku D
Buku C
Buku B
Buku A
--akhir tumpukan--
Press any key to continue . . .
```

penjelasan: buku yang bisa dipush adalah Buku A sampai Buku E saja. Sementara Buku F dan Buku G gagal di push ke dalam tumpukan. Mengapa demikian? Hal tersebut dikarenakan nilai dari N sendiri adalah 5. yang artinya tumpukan hanya bisa menampung 5 data. Dalam kasus ini, Buku F dan Buku G adalah data ke-6 dan data ke-7. Maka dari itu kedua Buku tersebut gagal dimasukkan ke dalam tumpukan. `if (atas == N-1)` merupakan baris kode yang berarti jika atas terpenuhi maka nilai dikurangi 1.

PRAKTIK 3

```
1 public class program_tumpukan{
2     public static int N = 5;
3     public static int atas = -1;
4     public static void PUSH (String tumpukan[], String data){
5         if (atas == N-1) //jika tumpukan penuh
6         {
7             System.out.println("maap, tumpukan penuh, PUSH " + data+ " tidak dapat dilakukan");
8         }
9         else //jika tumpukan tidak penuh
10        {
11            atas = atas + 1;
12            tumpukan[atas] = data;
13            System.out.println("PUSH " + data + " berhasil..");
14        }
15    }
16    public static String POP (String tumpukan[]){
17        String hasil;
18        if (atas < 0 ) //jika tumpukan kosong
19        {
20            hasil = "TUMPUKAN KOSONG, POP GAGAL DILAKUKAN";
21        }
22        else //jika tumpukan tidak kosong
23        {
24            hasil = tumpukan[atas];
25            atas = atas - 1;
26        }
27        return (hasil);
28    }
29    public static void lihatTumpukan(String tumpukan[]){
30        System.out.println("");
31        System.out.println("--TUMPUKAN:-");
32        for (int i=atas; i>=0; i--){
33            System.out.println(tumpukan[i]);
34        }
35        System.out.println("--akhir tumpukan-");
36        System.out.println("");
37    }
38    public static void main (String[] args){
39        String tumpukan[] = new String[7];
40        PUSH (tumpukan, "Buku A");
41        PUSH (tumpukan, "Buku B");
42        PUSH (tumpukan, "Buku C");
43        PUSH (tumpukan, "Buku D");
44        PUSH (tumpukan, "Buku E");
45        PUSH (tumpukan, "Buku F");
46        PUSH (tumpukan, "Buku G");
47        lihatTumpukan(tumpukan);
48        System.out.println("POP: " + POP(tumpukan));
49        lihatTumpukan(tumpukan);
50    }
51 }
```

```
C:\Windows\system32\cmd.exe
PUSH Buku A berhasil..
PUSH Buku B berhasil..
PUSH Buku C berhasil..
PUSH Buku D berhasil..
PUSH Buku E berhasil..
maap, tumpukan penuh, PUSH Buku F tidak dapat dilakukan
maap, tumpukan penuh, PUSH Buku G tidak dapat dilakukan
--TUMPUKAN:-
Buku E
Buku D
Buku C
Buku B
Buku A
--akhir tumpukan-
POP: Buku E
--TUMPUKAN:-
Buku D
Buku C
Buku B
Buku A
--akhir tumpukan-
Press any key to continue . . .
```

penjelasan: Buku yang ter-pop disini adalah Buku E. Mengapa demikian? Karena bisa dilihat bahwa Buku E adalah buku teratas pada tumpukan tersebut. Pop merupakan perintah untuk menghapus buku yang paling dekat dengan batas atas. Sementara kondisi “atas” langsung otomatis ditempati oleh buku di bawah E yaitu Buku D.

PRAKTIK 4

```
1 public class program_tumpukan{
2     public static int N = 5;
3     public static int atas = -1;
4     public static void PUSH (String tumpukan[], String data){
5         if (atas == N-1) //jika tumpukan penuh
6         {
7             System.out.println("maap, tumpukan penuh, PUSH " + data+ " tidak dapat dilakukan");
8         }
9         else //jika tumpukan tidak penuh
10        {
11            atas = atas + 1;
12            tumpukan[atas] = data;
13            System.out.println("PUSH " + data + " berhasil..");
14        }
15    }
16    public static String POP (String tumpukan[]){
17        String hasil;
18        if (atas < 0 ) //jika tumpukan kosong
19        {
20            hasil = "TUMPUKAN KOSONG, POP GAGAL DILAKUKAN";
21        }
22        else //jika tumpukan tidak kosong
23        {
24            hasil = tumpukan[atas];
25            atas = atas - 1;
26        }
27        return (hasil);
28    }
29    public static void lihatTumpukan(String tumpukan[]){
30        System.out.println("");
31        System.out.println("--TUMPUKAN:--");
32        for (int i=atas; i>=0; i--){
33            System.out.println(tumpukan[i]);
34        }
35        System.out.println("--akhir tumpukan--");
36        System.out.println("");
37    }
38    public static void main (String[] args){
39        String tumpukan[] = new String[10];
40        PUSH (tumpukan, "Buku A");
41        PUSH (tumpukan, "Buku B");
42        PUSH (tumpukan, "Buku C");
43        PUSH (tumpukan, "Buku D");
44        PUSH (tumpukan, "Buku E");
45        PUSH (tumpukan, "Buku F");
46        PUSH (tumpukan, "Buku G");
47        lihatTumpukan(tumpukan);
48        System.out.println("POP: " + POP(tumpukan));
49        System.out.println("POP: " + POP(tumpukan));
50        System.out.println("POP: " + POP(tumpukan));
51        lihatTumpukan(tumpukan);
52    }
53 }
```

```
C:\Windows\system32\cmd.exe
PUSH Buku A berhasil..
PUSH Buku B berhasil..
PUSH Buku C berhasil..
PUSH Buku D berhasil..
PUSH Buku E berhasil..
maap, tumpukan penuh, PUSH Buku F tidak dapat dilakukan
maap, tumpukan penuh, PUSH Buku G tidak dapat dilakukan

--TUMPUKAN:--
Buku E
Buku D
Buku C
Buku B
Buku A
--akhir tumpukan--

POP: Buku E
POP: Buku D
POP: Buku C

--TUMPUKAN:--
Buku B
Buku A
--akhir tumpukan--

Press any key to continue . . . _
```

penjelasan: buku yang ter-pop adalah Buku E, Buku D, dan Buku C. Mengapa demikian? Karena `System.out.println("POP: " + POP(tumpukan));` dideklarasikan sebanyak 3 kali. Yang artinya akan menghapus 3 data teratas di sebuah tumpukan. Kondisi “atas” sekarang telah ditempati oleh buku dibawah C yaitu Buku B.

PRAKTIK 5

```
1 public class program_tumpukan{
2     public static int N = 5;
3     public static int atas = -1;
4     public static void PUSH (String tumpukan[], String data){
5         if (atas == N-1) //jika tumpukan penuh
6         {
7             System.out.println("maap, tumpukan penuh, PUSH " + data+ " tidak dapat dilakukan");
8         }
9         else //jika tumpukan tidak penuh
10        {
11            atas = atas + 1;
12            tumpukan[atas] = data;
13            System.out.println("PUSH " + data + " berhasil..");
14        }
15    }
16    public static String POP (String tumpukan[]){
17        String hasil;
18        if (atas < 0 ) //jika tumpukan kosong
19        {
20            hasil = "TUMPUKAN KOSONG, POP GAGAL DILAKUKAN";
21        }
22        else //jika tumpukan tidak kosong
23        {
24            hasil = tumpukan[atas];
25            atas = atas - 1;
26        }
27        return (hasil);
28    }
29    public static void lihatTumpukan(String tumpukan[]){
30        System.out.println("");
31        System.out.println("--TUMPUKAN:--");
32        for (int i=atas; i>=0; i--){
33            System.out.println(tumpukan[i]);
34        }
35        System.out.println("--akhir tumpukan--");
36        System.out.println("");
37    }
38    public static void main (String[] args){
39        String tumpukan[] = new String[14];
40        PUSH (tumpukan, "Buku A");
41        PUSH (tumpukan, "Buku B");
42        PUSH (tumpukan, "Buku C");
43        PUSH (tumpukan, "Buku D");
44        PUSH (tumpukan, "Buku E");
45        PUSH (tumpukan, "Buku F");
46        PUSH (tumpukan, "Buku G");
47        lihatTumpukan(tumpukan);
48        System.out.println("POP: " + POP(tumpukan));
49        System.out.println("POP: " + POP(tumpukan));
50        System.out.println("POP: " + POP(tumpukan));
51        System.out.println("POP: " + POP(tumpukan));
52        System.out.println("POP: " + POP(tumpukan));
53        System.out.println("POP: " + POP(tumpukan));
54        lihatTumpukan(tumpukan);
55    }
56 }
```

```

C:\Windows\system32\cmd.exe
PUSH Buku A berhasil..
PUSH Buku B berhasil..
PUSH Buku C berhasil..
PUSH Buku D berhasil..
PUSH Buku E berhasil..
maap, tumpukan penuh, PUSH Buku F tidak dapat dilakukan
maap, tumpukan penuh, PUSH Buku G tidak dapat dilakukan

--TUMPUKAN:-
Buku E
Buku D
Buku C
Buku B
Buku A
--akhir tumpukan-

POP: Buku E
POP: Buku D
POP: Buku C
POP: Buku B
POP: Buku A
POP: TUMPUKAN KOSONG, POP GAGAL DILAKUKAN

--TUMPUKAN:-
--akhir tumpukan-

Press any key to continue . . . _

```

penjelasan: setelah saya tambahkan 3 pop lagi maka jumlah pop sekarang adalah 6. maka dari itu ada pop yang gagal dilakukan yaitu pop ke-6. Mengapa demikian? Hal tersebut dikarenakan pop hanya bisa dilakukan pada 5 data saja. Karena nilai $N = 5$.

PRAKTIK 6

```

1 public class program_antrian{
2     public static int N = 5;
3     public static int belakang = -1;
4     public static void ENQUEUE (String antrian[], String data){
5         if (belakang == N-1) //jika antrian penuh
6         {
7             System.out.println("maap, antrian penuh, ENQUEUE " + data + " tidak dapat dilakukan");
8         }
9         else //jika antrian tidak penuh
10        {
11            belakang = belakang + 1;
12            antrian[belakang] = data;
13            System.out.println("ENQUEUE " + data + " berhasil..");
14        }
15    }
16    public static String DEQUEUE (String antrian[]){
17        String hasil;
18        if (belakang < 0 ) //jika antrian kosong
19        {
20            hasil = "ANTRIAN KOSONG, DEQUEUE GAGAL DILAKUKAN";
21        }
22        else //jika antrian tidak kosong
23        {
24            hasil = antrian[0];
25            //----menggeser data kedua dst, maju selangkah ke depan
26            for (int i=0; i<=belakang-1; i++){
27                antrian[i] = antrian[i+1];
28            }
29            belakang = belakang - 1;
30        }
31        return (hasil);
32    }
33    public static void lihatAntrian(String antrian[]){
34        System.out.println("-----");
35        System.out.print("ISI ANTRIAN : (depan)");
36        for (int i=0; i<=belakang; i++){

```

```

37         System.out.print(" " + antrian[i]);
38     }
39     System.out.println(" (belakang)");
40     System.out.println("-----");
41 }
42 public static void main (String[] args){
43     String antrian[] = new String[5];
44     ENQUEUE (antrian, "Mobil A");
45     ENQUEUE (antrian, "Mobil B");
46     ENQUEUE (antrian, "Mobil C");
47     lihatAntrian(antrian);
48 }
49 }

```

```

C:\Windows\system32\cmd.exe
ENQUEUE Mobil A berhasil..
ENQUEUE Mobil B berhasil..
ENQUEUE Mobil C berhasil..
-----
ISI ANTRIAN : <depan> Mobil A Mobil B Mobil C <belakang>
-----
Press any key to continue . . .

```

penjelasan: mobil yang dapat di enqueue ke dalam antrian adalah Mobil A, Mobil B, dan Mobil C (semua Mobil). Posisi “belakang” saat ini berada di tempat Mobil C yaitu tempat terluar.

PRAKTIK 7

```

1 public class program_antrian{
2     public static int N = 5;
3     public static int belakang = -1;
4     public static void ENQUEUE (String antrian[], String data){
5         if (belakang == N-1) //jika antrian penuh
6         {
7             System.out.println("maap, antrian penuh, ENQUEUE "+ data + " tidak dapat dilakukan");
8         }
9         else //jika antrian tidak penuh
10        {
11            belakang = belakang + 1;
12            antrian[belakang] = data;
13            System.out.println("ENQUEUE " + data + " berhasil..");
14        }
15    }
16    public static String DEQUEUE (String antrian[]){
17        String hasil;
18        if (belakang < 0 ) //jika antrian kosong
19        {
20            hasil = "ANTRIAN KOSONG, DEQUEUE GAGAL DILAKUKAN";
21        }
22        else //jika antrian tidak kosong
23        {
24            hasil = antrian[0];
25            //---menggeser data kedua dst, maju selangkah ke depan
26            for (int i=0; i<=belakang-1; i++){
27                antrian[i] = antrian[i+1];
28            }
29            belakang = belakang - 1;
30        }
31        return (hasil);
32    }
33    public static void lihatAntrian(String antrian[]){
34        System.out.println("-----");
35        System.out.print("ISI ANTRIAN : (depan)");
36        for (int i=0; i<=belakang; i++){
37            System.out.print(" " + antrian[i]);
38        }
39        System.out.println(" (belakang)");
40        System.out.println("-----");
41    }
42    public static void main (String[] args){
43        String antrian[] = new String[5];
44        ENQUEUE (antrian, "Mobil A");
45        ENQUEUE (antrian, "Mobil B");
46        ENQUEUE (antrian, "Mobil C");
47        ENQUEUE (antrian, "Mobil D");
48        ENQUEUE (antrian, "Mobil E");
49        ENQUEUE (antrian, "Mobil F");
50        lihatAntrian(antrian);
51    }
52 }

```



```

C:\Windows\system32\cmd.exe
ENQUEUE Mobil A berhasil..
ENQUEUE Mobil B berhasil..
ENQUEUE Mobil C berhasil..
ENQUEUE Mobil D berhasil..
ENQUEUE Mobil E berhasil..
maap, antrian penuh, ENQUEUE Mobil F tidak dapat dilakukan
ISI ANTRIAN : <depan> Mobil A Mobil B Mobil C Mobil D Mobil E <belakang>
Press any key to continue . . . _

```

penjelasan: mobil yang ada dalam antrian adalah Mobil A, Mobil B, Mobil C, Mobil D, dan Mobil E. Ada satu mobil yang gagal dimasukkan ke antrian yaitu Mobil F. Mengapa demikian? Hal itu dikarenakan Mobil E merupakan data ke-6. Sementara antrian hanya bisa menampung 5 data karena $N = 5$. Posisi belakang saat ini ditempati oleh Mobil E. Perbedaan dengan praktik sebelumnya adalah posisi belakang sekarang diisi oleh Mobil E tetapi dengan F yang dicancel. Sementara sebelumnya posisi belakang diisi oleh Mobil C tanpa membuang 1 data pun.

PRAKTIK 8

```

1  public class program_antrian{
2  public static int N = 5;
3  public static int belakang = -1;
4  public static void ENQUEUE (String antrian[], String data){
5      if (belakang == N-1) //jika antrian penuh
6      {
7          System.out.println("maap, antrian penuh, ENQUEUE " + data + " tidak dapat dilakukan");
8      }
9      else //jika antrian tidak penuh
10     {
11         belakang = belakang + 1;
12         antrian[belakang] = data;
13         System.out.println("ENQUEUE " + data + " berhasil..");
14     }
15 }
16 public static String DEQUEUE (String antrian[]){
17     String hasil;
18     if (belakang < 0 ) //jika antrian kosong
19     {
20         hasil = "ANTRIAN KOSONG, DEQUEUE GAGAL DILAKUKAN";
21     }
22     else //jika antrian tidak kosong
23     {
24         hasil = antrian[0];
25         //---menggeser data kedua dst, maju selangkah ke depan
26         for (int i=0; i<=belakang-1; i++){
27             antrian[i] = antrian[i+1];
28         }
29         belakang = belakang - 1;
30     }
31     return (hasil);
32 }
33 public static void lihatAntrian(String antrian[]){
34     System.out.println("-----");
35     System.out.print("ISI ANTRIAN : (depan)");
36     for (int i=0; i<=belakang; i++){
37         System.out.print(" " + antrian[i]);
38     }
39     System.out.println(" (belakang)");
40     System.out.println("-----");
41 }
42 public static void main (String[] args){
43     String antrian[] = new String[5];
44     ENQUEUE (antrian, "Mobil A");
45     ENQUEUE (antrian, "Mobil B");
46     ENQUEUE (antrian, "Mobil C");
47     ENQUEUE (antrian, "Mobil D");
48     ENQUEUE (antrian, "Mobil E");
49     ENQUEUE (antrian, "Mobil F");
50     lihatAntrian(antrian);
51     System.out.println("deQueue: " + DEQUEUE(antrian));
52     lihatAntrian(antrian);
53 }
54 }

```

```
C:\Windows\system32\cmd.exe
ENQUEUE Mobil A berhasil..
ENQUEUE Mobil B berhasil..
ENQUEUE Mobil C berhasil..
ENQUEUE Mobil D berhasil..
ENQUEUE Mobil E berhasil..
maap, antrian penuh, ENQUEUE Mobil F tidak dapat dilakukan

ISI ANTRIAN : <depan> Mobil A Mobil B Mobil C Mobil D Mobil E <belakang>
deQueue: Mobil A
ISI ANTRIAN : <depan> Mobil B Mobil C Mobil D Mobil E <belakang>
Press any key to continue . . .
```

penjelasan: mobil yang ter-deQueue adalah Mobil A. Karena deQueue mengeluarkan mobil yang pertama masuk duluan. Ini merupakan konsep FIFO (First In First Out). Dan perubahan tentu terjadi. Yaitu B di paling depan dan E di paling belakang. Dengan kata lain jika kita mendeklarasikan deQueue lagi maka Mobil B lah yang akan dikeluarkan.

PRAKTIK 9

```
1 public class program_antrian{
2     public static int N = 5;
3     public static int belakang = -1;
4     public static void ENQUEUE (String antrian[], String data){
5         if (belakang == N-1) //jika antrian penuh
6         {
7             System.out.println("maap, antrian penuh, ENQUEUE "+ data + " tidak dapat dilakukan");
8         }
9         else //jika antrian tidak penuh
10        {
11            belakang = belakang + 1;
12            antrian[belakang] = data;
13            System.out.println("ENQUEUE " + data + " berhasil..");
14        }
15    }
16    public static String DEQUEUE (String antrian[]){
17        String hasil;
18        if (belakang < 0 ) //jika antrian kosong
19        {
20            hasil = "ANTRIAN KOSONG, DEQUEUE GAGAL DILAKUKAN";
21        }
22        else //jika antrian tidak kosong
23        {
24            hasil = antrian[0];
25            //---menggeser data kedua dst, maju selangkah ke depan
26            for (int i=0; i<=belakang-1; i++){
27                antrian[i] = antrian[i+1];
28            }
29            belakang = belakang - 1;
30        }
31        return (hasil);
32    }
33    public static void lihatAntrian(String antrian[]){
34        System.out.println("-----");
35        System.out.print("ISI ANTRIAN : (depan)");
36        for (int i=0; i<=belakang; i++){
37            System.out.print(" " + antrian[i]);
38        }
39        System.out.println(" (belakang)");
40        System.out.println("-----");
41    }
42    public static void main (String[] args){
43        String antrian[] = new String[5];
44        ENQUEUE (antrian, "Mobil A");
45        ENQUEUE (antrian, "Mobil B");
46        ENQUEUE (antrian, "Mobil C");
47        ENQUEUE (antrian, "Mobil D");
48        ENQUEUE (antrian, "Mobil E");
49        ENQUEUE (antrian, "Mobil F");
50        lihatAntrian(antrian);
51        System.out.println("deQueue: " + DEQUEUE (antrian));
52        System.out.println("deQueue: " + DEQUEUE (antrian));
53        System.out.println("deQueue: " + DEQUEUE (antrian));
54        System.out.println("deQueue: " + DEQUEUE (antrian));
55        lihatAntrian(antrian);
56    }
57 }
```

```

C:\Windows\system32\cmd.exe
ENQUEUE Mobil A berhasil..
ENQUEUE Mobil B berhasil..
ENQUEUE Mobil C berhasil..
ENQUEUE Mobil D berhasil..
ENQUEUE Mobil E berhasil..
maap, antrian penuh, ENQUEUE Mobil F tidak dapat dilakukan
-----
ISI ANTRIAN : <depan> Mobil A Mobil B Mobil C Mobil D Mobil E <belakang>
-----
deQueue: Mobil A
deQueue: Mobil B
deQueue: Mobil C
deQueue: Mobil D
-----
ISI ANTRIAN : <depan> Mobil E <belakang>
-----
Press any key to continue . . . _

```

penjelasan: saya menambahkan 3 deQueue lagi yang menjadikan jumlah deQueue menjadi 4. dengan itu maka ada 4 mobil yang ter-deQueue yaitu Mobil A, Mobil B, Mobil C, Mobil D. Tentu kondisi nya beda dengan praktik sebelumnya. Kali ini hanya tersisa Mobil E yang menjadi mobil di depan sekaligus mobil di belakang.

PRAKTIK 10

```

1 public class program_antrian{
2     public static int N = 5;
3     public static int belakang = -1;
4     public static void ENQUEUE (String antrian[], String data){
5         if (belakang == N-1) //jika antrian penuh
6         {
7             System.out.println("maap, antrian penuh, ENQUEUE " + data + " tidak dapat dilakukan");
8         }
9         else //jika antrian tidak penuh
10        {
11            belakang = belakang + 1;
12            antrian[belakang] = data;
13            System.out.println("ENQUEUE " + data + " berhasil..");
14        }
15    }
16    public static String DEQUEUE (String antrian[]){
17        String hasil;
18        if (belakang < 0 ) //jika antrian kosong
19        {
20            hasil = "ANTRIAN KOSONG, DEQUEUE GAGAL DILAKUKAN";
21        }
22        else //jika antrian tidak kosong
23        {
24            hasil = antrian[0];
25            //----menggeser data kedua dst, maju selangkah ke depan
26            for (int i=0; i<=belakang-1; i++){
27                antrian[i] = antrian[i+1];
28            }
29            belakang = belakang - 1;
30        }
31        return (hasil);
32    }
33    public static void lihatAntrian(String antrian[]){
34        System.out.println("-----");
35        System.out.print("ISI ANTRIAN : (depan)");
36        for (int i=0; i<=belakang; i++){
37            System.out.print(" " + antrian[i]);
38        }
39        System.out.println(" (belakang)");
40        System.out.println("-----");
41    }
42    public static void main (String[] args){
43        String antrian[] = new String[5];
44        ENQUEUE (antrian, "Mobil A");
45        ENQUEUE (antrian, "Mobil B");
46        ENQUEUE (antrian, "Mobil C");
47        ENQUEUE (antrian, "Mobil D");
48        ENQUEUE (antrian, "Mobil E");
49        ENQUEUE (antrian, "Mobil F");
50        lihatAntrian(antrian);
51        System.out.println("deQueue: " + DEQUEUE (antrian));
52        System.out.println("deQueue: " + DEQUEUE (antrian));
53        System.out.println("deQueue: " + DEQUEUE (antrian));
54        System.out.println("deQueue: " + DEQUEUE (antrian));
55        System.out.println("deQueue: " + DEQUEUE (antrian));
56        System.out.println("deQueue: " + DEQUEUE (antrian));
57        lihatAntrian(antrian);
58    }
59 }

```

```
C:\Windows\system32\cmd.exe
ENQUEUE Mobil A berhasil..
ENQUEUE Mobil B berhasil..
ENQUEUE Mobil C berhasil..
ENQUEUE Mobil D berhasil..
ENQUEUE Mobil E berhasil..
maap, antrian penuh, ENQUEUE Mobil F tidak dapat dilakukan
-----
ISI ANTRIAN : <depan> Mobil A Mobil B Mobil C Mobil D Mobil E <belakang>
-----
deQueue: Mobil A
deQueue: Mobil B
deQueue: Mobil C
deQueue: Mobil D
deQueue: Mobil E
deQueue: ANTRIAN KOSONG. DEQUEUE GAGAL DILAKUKAN
-----
ISI ANTRIAN : <depan> <belakang>
-----
Press any key to continue . . . _
```

penjelasan: ada proses deQueue yang tidak bisa dilakukan. Hal itu dikarenakan deQueue hanya bisa menghapus 5 data yaitu Mobil A, Mobil B, Mobil C, Mobil D, dan Mobil E. Mengapa hanya 5 data? Karena nilai N adalah 5 ($N=5$)

KESIMPULAN

Tumpukan atau stack adalah teknik peletakan data dimana seolah-olah data yang satu diletakkan 'diatas' data yang lain. Dalam tumpukan semua penyisipan dan penghapusan data dilakukan hanya melalui satu pintu yang disebut top (puncak) tumpukan. Tumpukan dapat dibayangkan seperti tumpukan buku dimana hanya buku teratas yang dapat diperoleh kembali dengan satu langkah. Buku-buku yang terdapat dibawah hanya dapat diambil setelah buku yang berada diatasnya diambil (dikeluarkan) terlebih dahulu. Dengan bentuk yang demikian tumpukan dapat dikatakan sebagai struktur data yang bersifat LIFO (Last In First Out). Dalam tumpukan terdapat 2 buah operasi data yaitu (1) operasi push dan (2) operasi pop. Operasi push adalah operasi untuk menambahkan sebuah data ke dalam tumpukan, sedangkan operasi pop adalah operasi untuk mengambil (menghapus) data dari dalam tumpukan.

Antrian atau Queue adalah teknik peletakan data dimana seolah-olah data yang satu diletakkan 'dibelakang' data yang lain. Dalam antrian semua penyisipan dan penghapusan data dilakukan melalui dua pintu yaitu belakang dan depan. Pintu belakang digunakan sebagai jalan masuk data, sedangkan pintu depan digunakan sebagai jalan keluar data. Dengan bentuk yang demikian antrian dapat dikatakan sebagai struktur data yang bersifat FIFO (First In First Out) artinya pengantri yang datang paling awal akan keluar paling dahulu. Dalam antrian terdapat 2 buah operasi data yaitu (1) operasi enqueue dan (2) operasi dequeue. Operasi enqueue adalah operasi untuk menambahkan sebuah data ke dalam antrian dan dilakukan pada pintu belakang, sedangkan operasi dequeue adalah operasi untuk mengambil (menghapus) data dari dalam antrian dan dilakukan pada pintu depan.