

Nama : Monica Tifani Zahara

NRP : 171 111 077 / TI

## Praktikum Pemrograman Dasar 2

### 1. Source code

```
01. public static void main(String[] args) {
02.     Scanner sc = new Scanner(System.in);
03.     LinkedList a = new LinkedList();
04.     int data = 0, bdata = 0, pilih;
05.     char ulang, ulangpr;
06.     ulang = 'y';
07.     ulangpr = 'y';
08.
09.     do {
10.         System.out.println("===== MENU =====");
11.         System.out.println("1. Masukkan Antrian ");
12.         System.out.println("2. Tampilkan Antrian ");
13.         System.out.println("3. Keluarkan Antrian dengan Stack ");
14.         System.out.println("4. Keluarkan Antrian dengan Queue");
15.         System.out.println("=====");
16.         do {
17.             System.out.print("Pilih proses\t: ");
18.             pilih = sc.nextInt();
19.             while (pilih < 1 || pilih > 4);
20.             switch (pilih) {
21.                 case 1:
22.                     do {
23.                         System.out.print("Masukkan banyak data\t: ");
24.                         bdata = sc.nextInt();
25.                         sc.nextLine();
26.                         System.out.println("-----");
27.                         for (int i = 1; i <= bdata; i++) {
28.                             System.out.print("Masukkan Data ke " + i + "\t: ");
29.                             data = sc.nextInt();
30.                             a.push(new LinkedListNode(data));
31.                         }
32.                         do {
33.                             System.out.print("Tambah antrian lagi? (Y / T)\t");
34.                             ulangpr = sc.next().charAt(0);
35.                             while (ulangpr != 't' && ulangpr != 'y');
36.                         } while (ulangpr == 'y');
37.                     } do {
38.                         System.out.print("Kembali ke menu awal ? (Y / T)\t");
39.                         ulang = sc.next().charAt(0);
40.                         while (ulang != 'y' && ulang != 't');
41.                     }
42.                     break;
43.             }
```

```

44.         case 2:
45.             System.out.println("===== Data Antrian =====");
46.             a.print();
47.             do {
48.                 System.out.print("Kembali ke menu awal ? (Y / T)");
49.                 ulang = sc.next().charAt(0);
50.             } while (ulang != 'y' && ulang != 't');
51.
52.             break;
53.
54.         case 3:
55.             if (a.head != null && a.tail != null) {
56.                 System.out.println("Antrian yang dihapus adalah : " + a.spop().data);
57.             } else {
58.                 System.out.println("Tidak dapat menghapus data. Antrian Kosong !!");
59.             }
60.             a.print();
61.             do {
62.                 System.out.print("Kembali ke menu awal ? (Y / T)");
63.                 ulang = sc.next().charAt(0);
64.             } while (ulang != 'y' && ulang != 't');
65.
66.             break;
67.
68.         case 4:
69.             if (a.head != null && a.tail != null) {
70.                 System.out.println("Antrian yang dihapus adalah : " + a.qpop().data);
71.             } else {
72.                 System.out.println("Tidak dapat menghapus data. Antrian Kosong !!");
73.             }
74.             a.print();
75.             do {
76.                 System.out.print("Kembali ke menu awal ? (Y / T)");
77.                 ulang = sc.next().charAt(0);
78.             } while (ulang != 'y' && ulang != 't');
79.             break;
80.         }
81.     } while (ulang == 'y' || ulang == 'Y');

```

```

01.  /*
02.  * To change this license header, choose License Headers in Project Properties.
03.  * To change this template file, choose Tools | Templates
04.  * and open the template in the editor.
05.  */
06.  package pertemuan3;
07.
08.  import java.util.Scanner;
09.
10.  public class LinkedList {
11.
12.      LinkedListNode head;
13.      LinkedListNode tail;
14.
15.      LinkedList() {
16.          this.head = null;
17.          this.tail = null;
18.      }
19.
20.      void print() {
21.          LinkedListNode current = this.head;
22.          while (current != null) {
23.              System.out.print(current.data + " ");
24.              current = current.next;
25.          }
26.          System.out.println("");
27.      }
28.
29.      void push(LinkedListNode new_node) {
30.          if (this.head == null && this.tail == null) {
31.              this.head = new_node;
32.              this.tail = new_node;
33.          } else {
34.              tail.next = new_node;
35.              new_node.prev = tail;
36.              this.tail = new_node;
37.          }
38.      }
39.
40.      void insert(LinkedListNode new_node) {
41.          if (this.head == null) {
42.              this.head = new_node;
43.              this.tail = new_node;
44.          } else if (new_node.data <= this.head.data) {
45.              this.head.set_prev(new_node);
46.              this.head = new_node;
47.          } else if (new_node.data >= this.tail.data) {
48.              this.tail.set_next(new_node);
49.              this.tail = new_node;
50.          } else {
51.              LinkedListNode position = head;
52.              while (position.data < new_node.data) {
53.                  position = position.next;
54.              }
55.              LinkedListNode previous_position = position.prev;
56.              new_node.set_prev(previous_position);
57.              new_node.set_next(position);
58.          }
59.      }
60.
61.  }
62.
63.

```

```

64.
65.     LinkedListNode find_node_by_data(int data) {
66.         LinkedListNode current = this.head;
67.         while (current != null) {
68.             if (current.data == data) {
69.                 return current;
70.             }
71.             current = current.next;
72.         }
73.         return null;
74.     }
75.
76.     LinkedListNode lastNode(LinkedListNode node) {
77.         while (node.next != null) {
78.             node = node.next;
79.         }
80.         // System.out.println("last : " + node.data);
81.         return node;
82.     }
83.
84.     void delete(LinkedListNode deleted) {
85.         if (deleted != null && this.head != null) {
86.             if (this.head == this.tail && deleted == this.head) {
87.                 this.head = null;
88.                 this.tail = null;
89.             } else if (deleted == this.head) {
90.                 LinkedListNode new_head = this.head.next;
91.                 this.head.set_next(null);
92.                 new_head.set_prev(null);
93.                 this.head = new_head;
94.             } else if (deleted == this.tail) {
95.                 LinkedListNode new_tail = this.tail.prev;
96.                 this.tail.set_prev(null);
97.                 new_tail.set_next(null);
98.                 this.tail = new_tail;
99.             } else {
100.                 LinkedListNode deleted_prev = deleted.prev;
101.                 LinkedListNode deleted_next = deleted.next;
102.                 deleted.set_prev(null);
103.                 deleted.set_next(null);
104.                 deleted_prev.set_next(deleted_next);
105.             }
106.         }
107.     }
108. }
109.

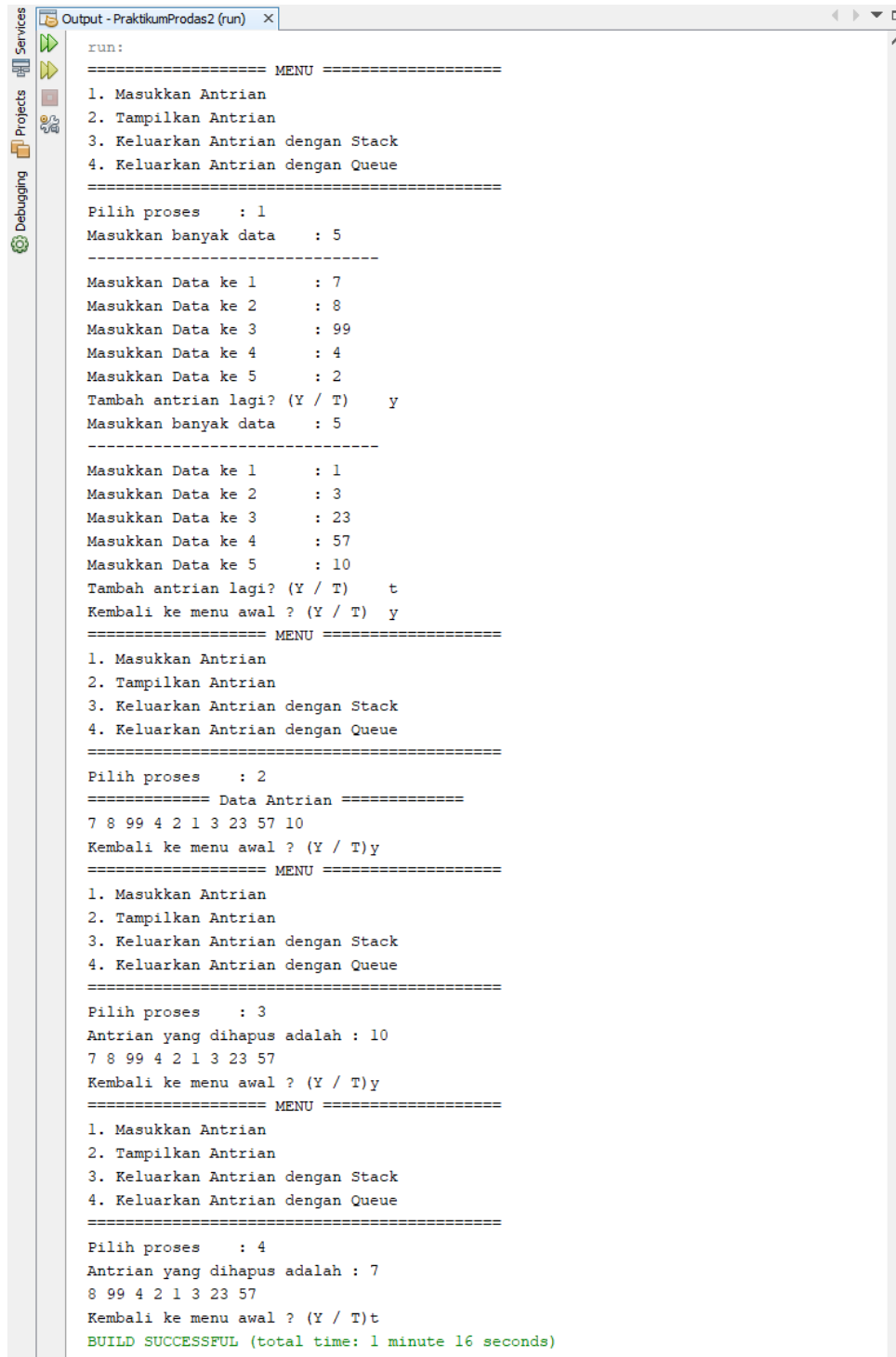
```

```

109.
110.     LinkedListNode qpop() {
111.         LinkedListNode taken = null;
112.         if (this.head == null && this.tail == null) {
113.             taken = null;
114.         } else if (this.head == this.tail) {
115.             taken = head;
116.             this.head = null;
117.             this.tail = null;
118.         } else {
119.             taken = head;
120.             head = head.next;
121.         }
122.         return taken;
123.     }
124. }
125.
126.     LinkedListNode spop() {
127.         LinkedListNode taken = null;
128.         if (this.head == null && this.tail == null) {
129.             taken = null;
130.         } else if (this.head == this.tail) {
131.             taken = tail;
132.             this.head = null;
133.             this.tail = null;
134.         } else {
135.             taken = tail;
136.             this.tail.prev.set_next(null);
137.             //tail.prev.next = null;
138.             this.tail = tail.prev;
139.         }
140.         return taken;
141.     }

```

## 2. Running program



```
run:
===== MENU =====
1. Masukkan Antrian
2. Tampilkan Antrian
3. Keluarkan Antrian dengan Stack
4. Keluarkan Antrian dengan Queue
=====
Pilih proses : 1
Masukkan banyak data : 5
-----
Masukkan Data ke 1 : 7
Masukkan Data ke 2 : 8
Masukkan Data ke 3 : 99
Masukkan Data ke 4 : 4
Masukkan Data ke 5 : 2
Tambah antrian lagi? (Y / T) y
Masukkan banyak data : 5
-----
Masukkan Data ke 1 : 1
Masukkan Data ke 2 : 3
Masukkan Data ke 3 : 23
Masukkan Data ke 4 : 57
Masukkan Data ke 5 : 10
Tambah antrian lagi? (Y / T) t
Kembali ke menu awal ? (Y / T) y
===== MENU =====
1. Masukkan Antrian
2. Tampilkan Antrian
3. Keluarkan Antrian dengan Stack
4. Keluarkan Antrian dengan Queue
=====
Pilih proses : 2
===== Data Antrian =====
7 8 99 4 2 1 3 23 57 10
Kembali ke menu awal ? (Y / T)y
===== MENU =====
1. Masukkan Antrian
2. Tampilkan Antrian
3. Keluarkan Antrian dengan Stack
4. Keluarkan Antrian dengan Queue
=====
Pilih proses : 3
Antrian yang dihapus adalah : 10
7 8 99 4 2 1 3 23 57
Kembali ke menu awal ? (Y / T)y
===== MENU =====
1. Masukkan Antrian
2. Tampilkan Antrian
3. Keluarkan Antrian dengan Stack
4. Keluarkan Antrian dengan Queue
=====
Pilih proses : 4
Antrian yang dihapus adalah : 7
8 99 4 2 1 3 23 57
Kembali ke menu awal ? (Y / T)t
BUILD SUCCESSFUL (total time: 1 minute 16 seconds)
```