NAMA : Adani Salsabila
NIM : 2341720123
KELAS : 1-G
MATERI : Double Linked Lists (Jobsheet 10)

1. Praktikum 1

src > P12 > Node.java > Node

```java
package P12;

public class Node {
    int data;
    Node prev, next;

    Node(Node prev, int data, Node next) {
        this.prev = prev;
        this.data = data;
        this.next = next;
    }
}
```

src > P12 > DoubleLinkedLists.java > DoubleLinkedLists > addFirst(int)

```java
package P12;

public class DoubleLinkedLists {
    Node head;
    int size;

    public DoubleLinkedLists() {
        head = null;
        size = 0;
    }

    public boolean isEmpty() {
        return head == null;
    }

    public void addFirst(int item) {
        if (isEmpty()) {
            head = new Node(prev:null, item, next:null);
        } else {
            Node newNode = new Node(prev:null, item, head);
            head.prev = newNode;
            head = newNode;
        }
        size++;
    }

    public void addLast(int item) {
        if (isEmpty()) {
            addFirst(item);
        } else {
            Node current = head;
            while (current.next != null) {
                current = current.next;
            }
            Node newNode = new Node(current, item, next:null);
            current.next = newNode;
            size++;
        }
    }
```

src > P12 > DoubleLinkedLists.java > DoubleLinkedLists > addFirst(int)

```java
public class DoubleLinkedLists {

    public void add(int item, int index) throws Exception {
        if (index < 0 || index > size) {
            throw new Exception(message:"Nilai indeks di luar batas");
        } else if (index == 0) {
            addFirst(item);
        } else if (index == size) {
            addLast(item);
        } else {
            Node current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }
            Node newNode = new Node(current.prev, item, current);
            current.prev.next = newNode;
            current.prev = newNode;
            size++;
        }
    }

    public int size() {
        return size;
    }

    public void clear() {
        head = null;
        size = 0;
    }

    public void print() {
        if (!isEmpty()) {
            Node tmp = head;
            while (tmp != null) {
                System.out.print(tmp.data + "\t");
                tmp = tmp.next;
            }
            System.out.println(x:"\nberhasil diisi");
        } else {
            System.out.println(x:"Linked Lists Kosong");
        }
    }
}
```

src > P12 > DoubleLinkedListsMain.java > DoubleLinkedListsMain

```java
package P12;

public class DoubleLinkedListsMain {
    Run | Debug
    public static void main(String[] args) throws Exception {
        DoubleLinkedLists dll = new DoubleLinkedLists();
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"==================================");
        dll.addFirst(item:3);
        dll.addLast(item:4);
        dll.addFirst(item:7);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"==================================");
        dll.add(item:40, index:1);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"==================================");
        dll.clear();
        dll.print();
        System.out.println("Size : " + dll.size());
    }
}
```

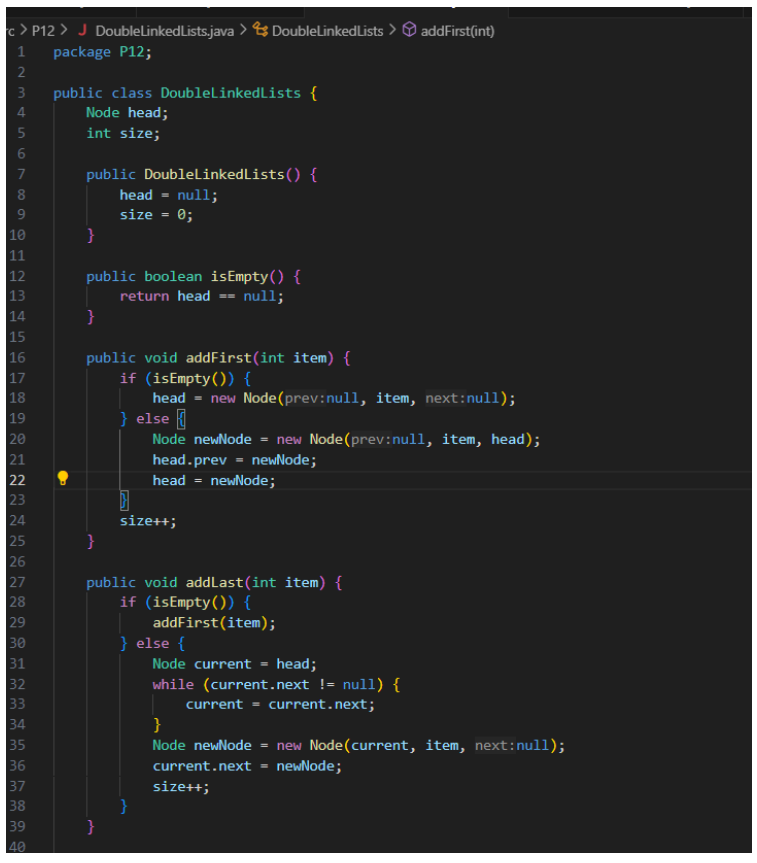
```
PS D:\college\semester 2\Algoritma_dan_Struktur_Data_1G_01>  d:; cd 'd:\co
oritma_dan_Struktur_Data_1G_01'; & 'C:\Program Files\Java\jdk-20\bin\java.
oritma_dan_Struktur_Data_1G_01'; & 'C:\Program Files\Java\jdk-20\bin\java.
DoubleLinkedListsMain'
Linked Lists Kosong
Size : 0
==================================
7       3       4
berhasil diisi
Size : 3
==================================
7       40      3       4
berhasil diisi
Size : 4
==================================
Linked Lists Kosong
Size : 0
PS D:\college\semester 2\Algoritma_dan_Struktur_Data_1G_01>
```

NAMA : Adani Salsabila
NIM : 2341720123
KELAS : 1-G
MATERI : Double Linked Lists (Jobsheet 10)

Pertanyaan!

1. Jelaskan perbedaan antara single linked list dengan double linked lists!
   Jawab : Perbedaan single linked list dengan double linked list berada di pointer, single linked list memiliki hanya satu pointer sedangkan double linked list memiliki dua pointer. Single linked list memiliki data dan next, sedangkan double linked list memiliki data, previous, dan next.
2. Perhatikan class Node, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?
   Jawab : Atribut next digunakan untuk referensi ke node berikutnya dalam linked list, sedangkan atribut prev adalah referensi ke node sebelumnya.
3. Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan inisialisasi atribut head dan size seperti pada gambar berikut ini?
   ```
   public DoubleLinkedLists() {
   head = null;
   size = 0;
   }
   ```
   Jawab : Kegunaan inisialisasi atribut head dan size pada gambar tersebut adalah untuk membuat linked list dalam keadaan kosong agar tidak ada operasi penambahan elemen yang dilakukan.
4. Pada method addFirst(), kenapa dalam pembuatan object dari konstruktor class Node prev dianggap sama dengan null?
   ```
   Node newNode = new Node(null, item, head);
   ```
   Jawab : Karena tidak ada node sebelumnya yang harus dihubungkan dengan elemen baru, menandakan elemen baru tidak memiliki node sebelumnya dalam linked list.
5. Perhatikan pada method addFirst(). Apakah arti statement head.prev = newNode ?
   Jawab : Arti statement tersebut pada method addFirst menghubungkan node sebelumnya dari head saat ini ke newNode, mengupdate referensi ke node sebelumnya di dalam struktur linked lists.
6. Perhatikan isi method addLast(), apa arti dari pembuatan object Node dengan mengisikan parameter prev dengan current, dan next dengan null?
   ```
   Node newNode = new Node(current, item, null);
   ```
   Jawab : Arti dari pembuatan object Node dengan mengisikan parameter prev dengan current, dan next dengan null bertujuan untuk menambahkan node baru di belakang linked list.
7. Pada method add(), terdapat potongan kode program sebagai berikut, jelaskan!
   ```
   if (current.prev == null) {
   Node newNode = new Node(null, item, current);
   current.prev = newNode;
   head = newNode;
   ```
   Jawab : Di dalam potongan kode tersebut, penambahan node ketika current adalah node pertama dalam linked list, dengan membuat node baru di head linked list dan menghubungkannya dengan node pertama yang sudah ada.

2. Praktikum 2



NAMA : Adani Salsabila
NIM : 2341720123
KELAS : 1-G
MATERI : Double Linked Lists (Jobsheet 10)

src > P12 > DoubleLinkedLists.java > DoubleLinkedLists > addFirst(int)

```java
public class DoubleLinkedLists {

    public void add(int item, int index) throws Exception {
        if (index < 0 || index > size) {
            throw new Exception(message:"Nilai indeks di luar batas");
        } else if (index == 0) {
            addFirst(item);
        } else if (index == size) {
            addLast(item);
        } else {
            Node current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }

            if (current.prev == null) {
                Node newNode = new Node(prev:null, item, current);
                current.prev = newNode;
                head = newNode;
            } else {
            Node newNode = new Node(current.prev, item, current);
            current.prev.next = newNode;
            current.prev = newNode;
            size++;
        }
    }
}

    public int size() {
        return size;
    }

    public void clear() {
        head = null;
        size = 0;
    }
```

src > P12 > DoubleLinkedLists.java > DoubleLinkedLists > addFirst(int)

```java
public class DoubleLinkedLists {
    public void print() {
        if (!isEmpty()) {
            Node tmp = head;
            while (tmp != null) {
                System.out.print(tmp.data + "\t");
                tmp = tmp.next;
            }
            System.out.println(x:"\nberhasil diisi");
        } else {
            System.out.println(x:"Linked Lists Kosong");
        }
    }

    public void removeFirst() throws Exception {
        if (isEmpty()) {
            throw new Exception(message:"Linked List masih kosong, tidak dapat dihapus!");
        } else if (size == 1) {
            removeLast();
        } else {
            head = head.next;
            head.prev = null;
            size --;
        }
    }

    public void removeLast() throws Exception {
        if (isEmpty()) {
            throw new Exception(message:"Linked List masih kosong, tidak dapat dihapus!");
        } else if (head.next == null) {
            head = null;
            size --;
            return;
        }
        Node current = head;
        while (current.next.next != null) {
            current = current.next;
        }
        current.next = null;
        size --;
    }
```

src > P12 > DoubleLinkedLists.java > DoubleLinkedLists > addFirst(int)

```java
public class DoubleLinkedLists {

    public void remove(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception(message:"Nilai index di luar batas");
        } else if (index == 0) {
            removeFirst();
        } else {
            Node current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }
            if (current.next == null) {
                current.prev.next = null;

            } else if (current.prev == null) {
                current = current.next;
                current.prev = null;
                head = current;
            } else {
                current.prev.next = current.next;
                current.next.prev = current.prev;
            }
            size--;
        }
    }
}
```

NAMA : Adani Salsabila
NIM : 2341720123
KELAS : 1-G
MATERI : Double Linked Lists (Jobsheet 10)

src > P12 > DoubleLinkedListsMain.java > DoubleLinkedListsMain > main(String[])

```java
package P12;

public class DoubleLinkedListsMain {
    Run | Debug
    public static void main(String[] args) throws Exception {
        DoubleLinkedLists dll = new DoubleLinkedLists();
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.addFirst(item:3);
        dll.addLast(item:4);
        dll.addFirst(item:7);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.add(item:40, index:1);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.clear();
        dll.print();
        System.out.println("Size : " + dll.size());
        dll.addLast(item:50);
        dll.addLast(item:40);
        dll.addLast(item:10);
        dll.addLast(item:20);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.removeFirst();
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.removeLast();
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.remove(index:1);
        dll.print();
        System.out.println("Size : " + dll.size());
    }
}
```

PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
 'P12.DoubleLinkedListsMain'
ta_1G_01\x5cbin' 'P12.DoubleLinkedListsMain' ;3e5555b0-32b9-47d3-82d0-a3e93d4581a1Linked Lists
 Kosong
Size : 0
====================================
7       3       4
berhasil diisi
Size : 3
====================================
7       40      3       4
berhasil diisi
Size : 4
====================================
Linked Lists Kosong
Size : 0
50      40      10      20
berhasil diisi
Size : 4
====================================
40      10      20
berhasil diisi
Size : 3
====================================
40      10
berhasil diisi
Size : 2
====================================
40
berhasil diisi
Size : 1
PS D:\college\semester 2\Algoritma_dan_Struktur_Data_1G_01>
```

Pertanyaan!

1. Apakah maksud statement berikut pada method removeFirst()?

```
head = head.next;
head.prev = null;
```

Jawab : Maksud dari statement tersebut pada method removeFirst() adalah untuk menghapus node pertama dari linked list dan memperbarui kepala (head) serta mengatur kembali referensi node sebelum head menjadi null sehingga linked list tetap dapat yerhubung dengan benar setelah operasi penghubungan.

2. Bagaimana cara mendeteksi posisi data ada pada bagian akhir pada method removeLast()?
   Jawab : Cara mendeteksi posisi data ada pada bagian akhir pada method removeLast() adalah pertama periksa apakah node saat ini adalah node terakhir atau bukan dengan cara iterasi linked list dari awal sampai akhir, ketika sudah di node terakhir tandanya adalah node berikutnya setelah node ini adalah null.
3. Jelaskan alasan potongan kode program di bawah ini tidak cocok untuk perintah remove!

```
Node tmp = head.next;
head.next = tmp.next;
tmp.next.prev = head;
```

Jawab : Alasan potongan kode program di atas tidak cocok untuk perintah remove adalah karena kode tersebut hanya menghapus node kedua dalam linked list, bukan node yang spesifik. Karena, method remove() secara umum digunakan untuk menghapus node mana pun sesuai dengan parameter yang sudah dimasukkan.

4. Jelaskan fungsi kode program berikut ini pada fungsi remove!

```
current.prev.next = current.next;
current.next.prev = current.prev;
```

Jawab : Fungsi kode program pada fungsi remove tersebut adalah untuk menghapus node current dari linked list tanpa mengganggu relasi antar node lain dalam struktur linked list.

NAMA : Adani Salsabila
NIM : 2341720123
KELAS : 1-G
MATERI : Double Linked Lists (Jobsheet 10)

3. Praktikum 3

```java
package P12;

public class DoubleLinkedLists {
    Node head;
    int size;

    public DoubleLinkedLists() {
        head = null;
        size = 0;
    }

    public boolean isEmpty() {
        return head == null;
    }

    public void addFirst(int item) {
        if (isEmpty()) {
            head = new Node(prev:null, item, next:null);
        } else {
            Node newNode = new Node(prev:null, item, head);
            head.prev = newNode;
            head = newNode;
        }
        size++;
    }

    public void addLast(int item) {
        if (isEmpty()) {
            addFirst(item);
        } else {
            Node current = head;
            while (current.next != null) {
                current = current.next;
            }
            Node newNode = new Node(current, item, next:null);
            current.next = newNode;
            size++;
        }
    }
```

```java
    public void add(int item, int index) throws Exception {
        if (index < 0 || index > size) {
            throw new Exception(message:"Nilai indeks di luar batas");
        } else if (index == 0) {
            addFirst(item);
        } else if (index == size) {
            addLast(item);
        } else {
            Node current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }

            if (current.prev == null) {
                Node newNode = new Node(prev:null, item, current);
                current.prev = newNode;
                head = newNode;
            } else {
            Node newNode = new Node(current.prev, item, current);
            current.prev.next = newNode;
            current.prev = newNode;
            size++;
        }
    }
}

    public int size() {
        return size;
    }

    public void clear() {
        head = null;
        size = 0;
    }
```

```java
    public void print() {
        if (!isEmpty()) {
            Node tmp = head;
            while (tmp != null) {
                System.out.print(tmp.data + "\t");
                tmp = tmp.next;
            }
            System.out.println(x:"\nberhasil diisi");
        } else {
            System.out.println(x:"Linked Lists Kosong");
        }
    }

    public void removeFirst() throws Exception {
        if (isEmpty()) {
            throw new Exception(message:"Linked List masih kosong, tidak dapat dihapus!");
        } else if (size == 1) {
            removeLast();
        } else {
            head = head.next;
            head.prev = null;
            size --;
        }
    }

    public void removeLast() throws Exception {
        if (isEmpty()) {
            throw new Exception(message:"Linked List masih kosong, tidak dapat dihapus!");
        } else if (head.next == null) {
            head = null;
            size --;
            return;
        }
        Node current = head;
        while (current.next.next != null) {
            current = current.next;
        }
        current.next = null;
        size --;
    }
```

NAMA : Adani Salsabila
NIM : 2341720123
KELAS : 1-G
MATERI : Double Linked Lists (Jobsheet 10)

src > P12 > DoubleLinkedLists.java > DoubleLinkedLists > addFirst(int)

```java
public class DoubleLinkedLists {

    public void remove(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception(message:"Nilai index di luar batas");
        } else if (index == 0) {
            removeFirst();
        } else {
            Node current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }
            if (current.next == null) {
                current.prev.next = null;

            } else if (current.prev == null) {
                current = current.next;
                current.prev = null;
                head = current;
            } else {
                current.prev.next = current.next;
                current.next.prev = current.prev;
            }
            size--;
        }
    }

    public int getFirst() throws Exception {
        if (isEmpty()) {
            throw new Exception(message:"Linked List kosong");
        }
        return head.data;
    }

    public int getLast() throws Exception {
        if (isEmpty()) {
            throw new Exception(message:"Linked List kosong");
        }
        Node tmp = head;
        while (tmp.next != null) {
            tmp = tmp.next;
        }
        return tmp.data;
    }

    public int get(int index) throws Exception {
        if (index < 0 || index >= size) {
            throw new Exception(message:"Nilai index di luar batas");
        }
        Node current = head;
        for (int i = 0; i < index; i++) {
            current = current.next;
        }
        return current.data;
    }

}
```

src > P12 > DoubleLinkedListsMain.java > DoubleLinkedListsMain > main(String[])

```java
package P12;

public class DoubleLinkedListsMain {
    public static void main(String[] args) throws Exception {
        DoubleLinkedLists dll = new DoubleLinkedLists();
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.addFirst(item:3);
        dll.addLast(item:4);
        dll.addFirst(item:7);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        dll.add(item:40, index:1);
        dll.print();
        System.out.println("Size : " + dll.size());
        System.out.println(x:"====================================");
        System.out.println("Data awal pada Linked Lists adalah : " + dll.getFirst());
        System.out.println("Data akhir pada Linked Lists adalah : " + dll.getLast());
        System.out.println("Data indeks ke-1 pada Linked Lists adalah : " + dll.get(index:1));
    }
}
```

```
PS D:\college\semester 2\Algoritma_dan_Struktur_Data_1G_01>
PS D:\college\semester 2\Algoritma_dan_Struktur_Data_1G_01>  d:; cd 'd:\colleg
oritma_dan_Struktur_Data_1G_01'; & 'C:\Program Files\Java\jdk-20\bin\java.exe'
oritma_dan_Struktur_Data_1G_01'; & 'C:\Program Files\Java\jdk-20\bin\java.exe'
DoubleLinkedListsMain'
Linked Lists Kosong
Size : 0
=====================================
7       3       4
berhasil diisi
Size : 3
=====================================
7       40      3       4
berhasil diisi
Size : 4
=====================================
Data awal pada Linked Lists adalah : 7
Data akhir pada Linked Lists adalah : 4
Data indeks ke-1 pada Linked Lists adalah : 40
PS D:\college\semester 2\Algoritma_dan_Struktur_Data_1G_01>
```

NAMA : Adani Salsabila
NIM : 2341720123
KELAS : 1-G
MATERI : Double Linked Lists (Jobsheet 10)

Pertanyaan!

1. Jelaskan method size() pada class DoubleLinkedLists!
   Jawab : Method size() pada class DoubleLinkedLists berfungsi untuk mengembalikan jumlah elemen atau ukuran dari linked list. Ketika dipanggil, method tersebut akan mengembalikan jumlah elemen dlm linked list saat itu.
2. Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke-1!
   Jawab : Cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke-1 adalah dengan mengubah posisi head menjadi indeks -1 dan kita harus menggunakan indeks yang sesuai dengan urutan node dalam linked list yang dimulai dari 1, bukan 0.
3. Jelaskan perbedaan karakteristik fungsi Add pada Double Linked Lists dan Single Linked Lists!
   Jawab : Dalam single linked lists, tiap node hanya memiliki satu pointer yaitu next, sehingga saat ingin menambahkan node baru hanya bisa menunjuk ke node baru, sedangkan di double linked lists pointernya ada dua yaitu next dan prev sehingga bisa menunjuk ke node sebelumnya atau berikutnya. Dan karena double linked lists memiliki dua pointer, maka bisa memiliki akses ke node prev, sehingga bisa menambahkan node baru di awal, tengah, atau akhir, berbeda dengan single linked list yang penambahan nodenya terbatas.
4. Jelaskan perbedaan logika dari kedua kode program di bawah ini!
   a.

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

   b.

```
public boolean isEmpty() {
return head == null;
```

   Jawab : Kode program a memeriksa kekosongan linked list berdasarkan pada jumlah elemen yang tersimpan, sengkan kode program b memeriksa kekosongan linked list berdasarkan pada apakah kepala linked list memiliki nilai null. Kemudian, kode program a memeriksa jumlah elemen secara langsung, sedangkan kode program b memeriksa keberadaan head linked list.