JOBSHEET 12

Double Linked Lists

|  |
| --- |
| **Nama : Muhammad Abhinaya Zurfa** |
| **Kelas : SIB 1 C / 20** |

# Tujuan Praktikum

Setelah melakukan praktikum ini, mahasiswa mampu:

1. memahami algoritma double linked lists;
2. membuat dan mendeklarasikan struktur algoritma double linked lists;
3. menerapkan algoritma double linked lists dalam beberapa *study case*.

# Kegiatan Praktikum 1

* + 1. **Percobaan 1**

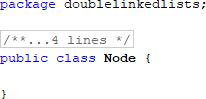
Pada percobaan 1 ini akan dibuat class Node dan class DoubleLinkedLists yang didalamnya terdapat operasi-operasi untuk menambahkan data dengan beberapa cara (dari bagian depan linked list, belakang ataupun indeks tertentu pada linked list).

1. Perhatikan diagram class Node dan class DoublelinkedLists di bawah ini! Diagram class ini yang selanjutnya akan dibuat sebagai acuan dalam membuat kode program DoubleLinkedLists.

|  |
| --- |
| Node |
| data: int prev: Node  next: Node |
| Node(prev: Node, data:int, next:Node) |

|  |
| --- |
| DoubleLinkedLists |
| head: Node  size : int |
| DoubleLinkedLists() isEmpty(): boolean addFirst (): void addLast(): void  add(item: int, index:int): void size(): int  clear(): void  print(): void |

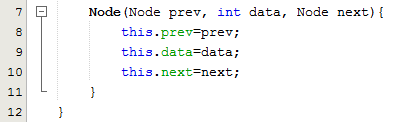
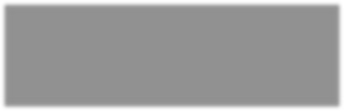
1. Buat paket baru dengan nama **doublelinkedlists**
2. Buat class di dalam paket tersebut dengan nama **Node**



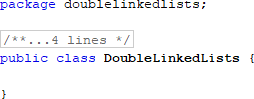
1. Di dalam class tersebut, deklarasikan atribut sesuai dengan diagram class di atas.



1. Selanjutnya tambahkan konstruktor default pada class Node sesuai diagram di atas.



1. Buatlah sebuah class baru bernama DoubleLinkedLists pada package yang sama dengan node seperti gambar berikut:



1. Pada class DoubleLinkedLists tersebut, deklarasikan atribut sesuai dengan diagram class di atas.



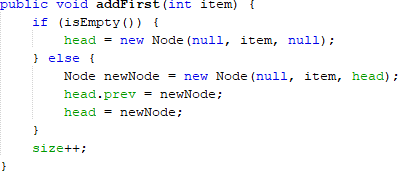
1. Selajuntnya, buat konstruktor pada class DoubleLinkedLists sesuai gambar berikut.

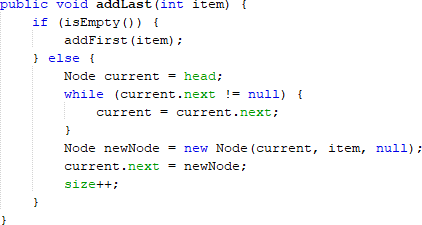


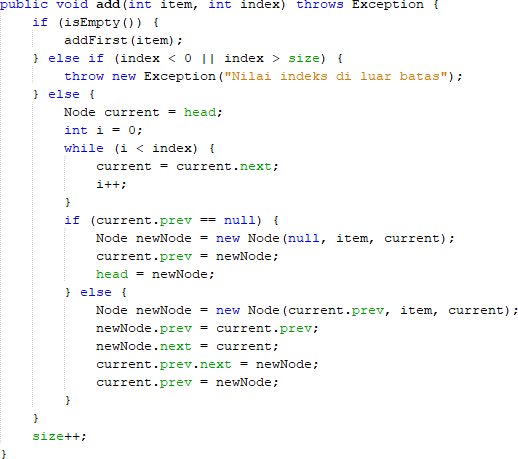
1. Buat method **isEmpty()**. Method ini digunakan untuk memastikan kondisi linked list kosong.



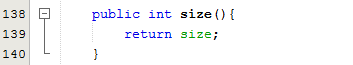
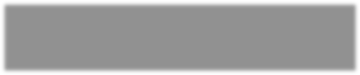
1. Kemudian, buat method **addFirst().** Method ini akan menjalankan penambahan data di bagian depan linked list.



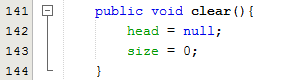
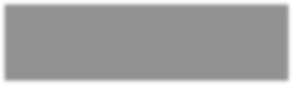
1. Selain itu pembuatan method **addLast()** akan menambahkan data pada bagian belakang linked list.
2. Untuk menambahkan data pada posisi yang telah ditentukan dengan indeks, dapat dibuat dengan method **add(int item, int index)**



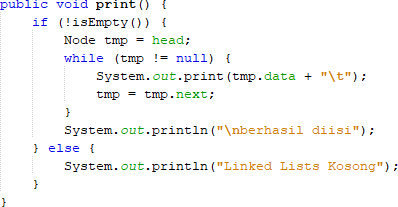
1. Jumlah data yang ada di dalam linked lists akan diperbarui secara otomatis, sehingga dapat dibuat method **size()** untuk mendapatkan nilai dari size.



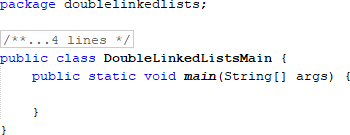
1. Selanjutnya dibuat method **clear()** untuk menghapus semua isi linked lists, sehingga linked lists dalam kondisi kosong.



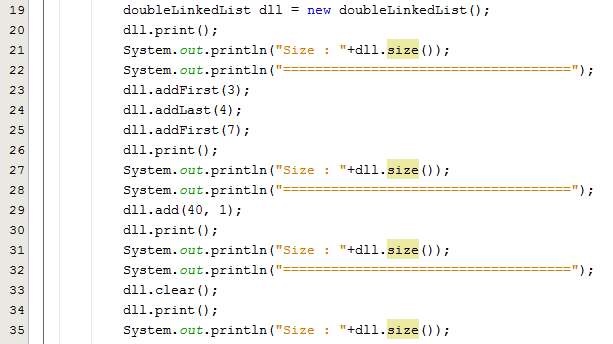
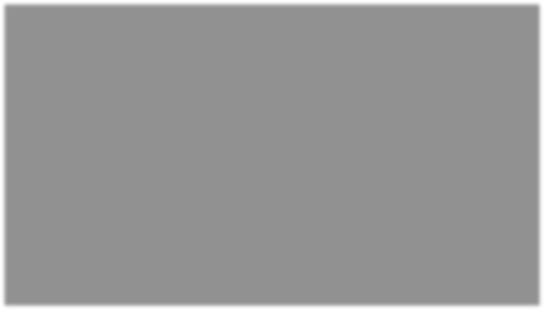
1. Untuk mencetak isi dari linked lists dibuat method **print().** Method ini akan mencetak isi linked lists berapapun size-nya. Jika kosong akan dimunculkan suatu pemberitahuan bahwa linked lists dalam kondisi kosong.



1. Selanjutya dibuat class Main DoubleLinkedListsMain untuk mengeksekusi semua method yang ada pada class DoubleLinkedLists.

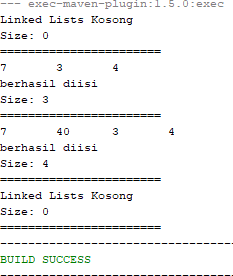


1. Pada main class pada langkah 16 di atas buatlah object dari class DoubleLinkedLists kemudian eksekusi potongan program berikut ini.



## Verifikasi Hasil Percobaan

Verifikasi hasil kompilasi kode program Anda dengan gambar berikut ini.



package doublelinkedlists;

/\*\*

 \* @author Abhinaya

 \*/

public class Node {

    int data;

    Node prev;

    Node next;

    public Node(Node prev, int data, Node next) {

        this.prev = prev;

        this.data = data;

        this.next = next;

    }

}

package doublelinkedlists;

/\*\*

 \* @author Abhinaya

 \*/

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(null, item, null);

        } else {

            Node newNode = new Node(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, null);

            current.next = newNode;

            size++;

        }

    }

    public void add(int item, int index) throws Exception {

        if (isEmpty()) {

            addFirst(item);

        } else if (index < 0 || index > size) {

            throw new Exception("Nilai indeks di luar batas");

        } else {

            Node current = head;

            int i = 0;

            while (i < index) {

                current = current.next;

                i++;

            }

            if (current.prev == null) {

                Node newNode = new Node(null, item, current);

                current.prev = newNode;

                head = newNode;

            } else {

                Node newNode = new Node(current.prev, item, current);

                newNode.prev = current.prev;

                newNode.next = 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("\nBerhasil diisi");

        } else {

            System.out.println("Linked Lists kosong");

        }

    }

}

package doublelinkedlists;

/\*\*

 \* @author Abhinaya

 \*/

public class DoubleLinkedListMain {

    public static void main(String[] args) {

        DoubleLinkedLists dll = new DoubleLinkedLists();

        dll.print();

        System.out.println("Size : " + dll.size());

        System.out.println("==================================");

        dll.addFirst(3);

        dll.addLast(4);

        dll.addFirst(7);

        dll.print();

        System.out.println("Size : " + dll.size());

        System.out.println("==================================");

        dll.add(40, 1);

        dll.print();

        System.out.println("Size : " + dll.size());

        System.out.println("==================================");

        dll.clear();

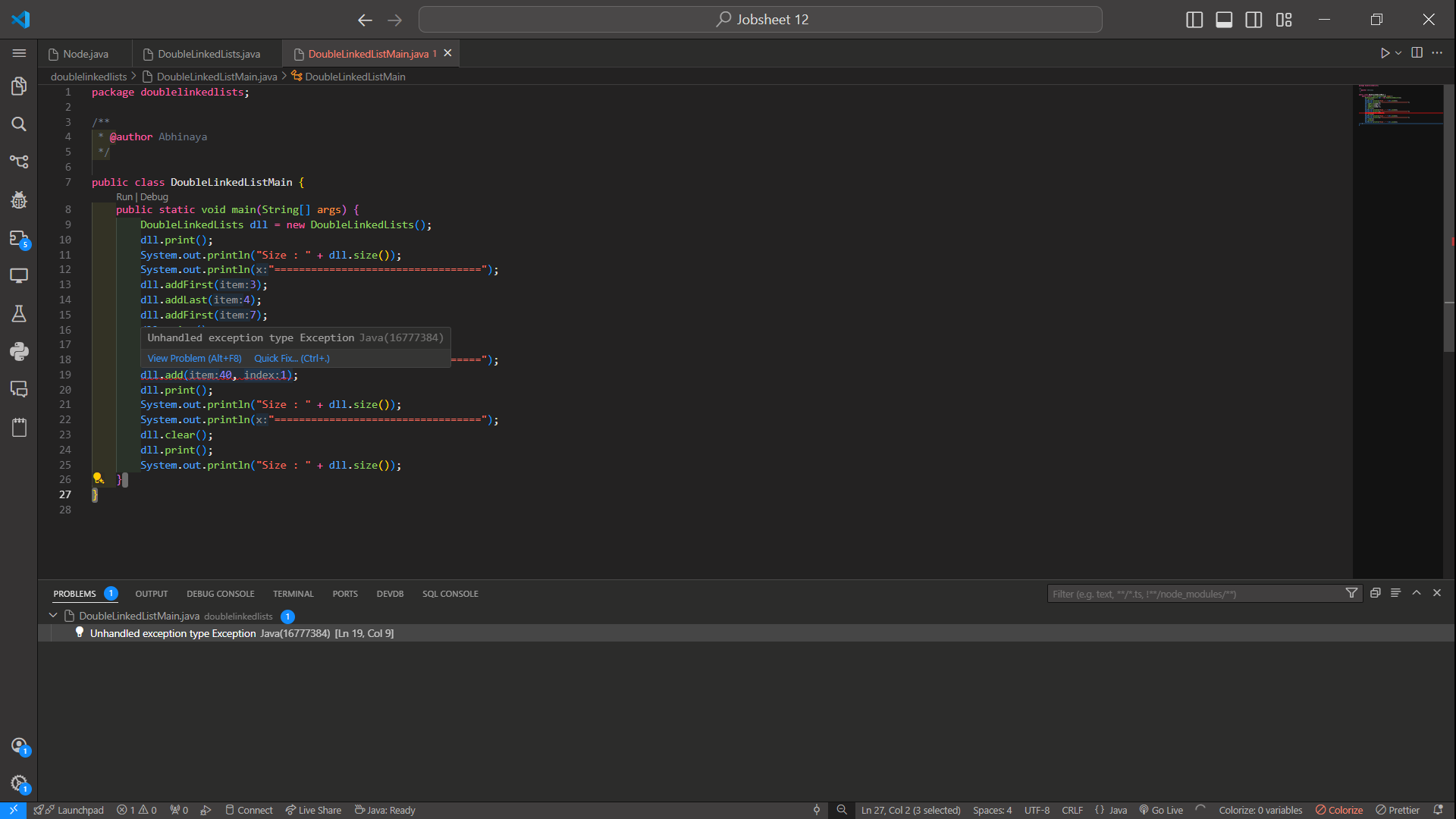
        dll.print();

        System.out.println("Size : " + dll.size());

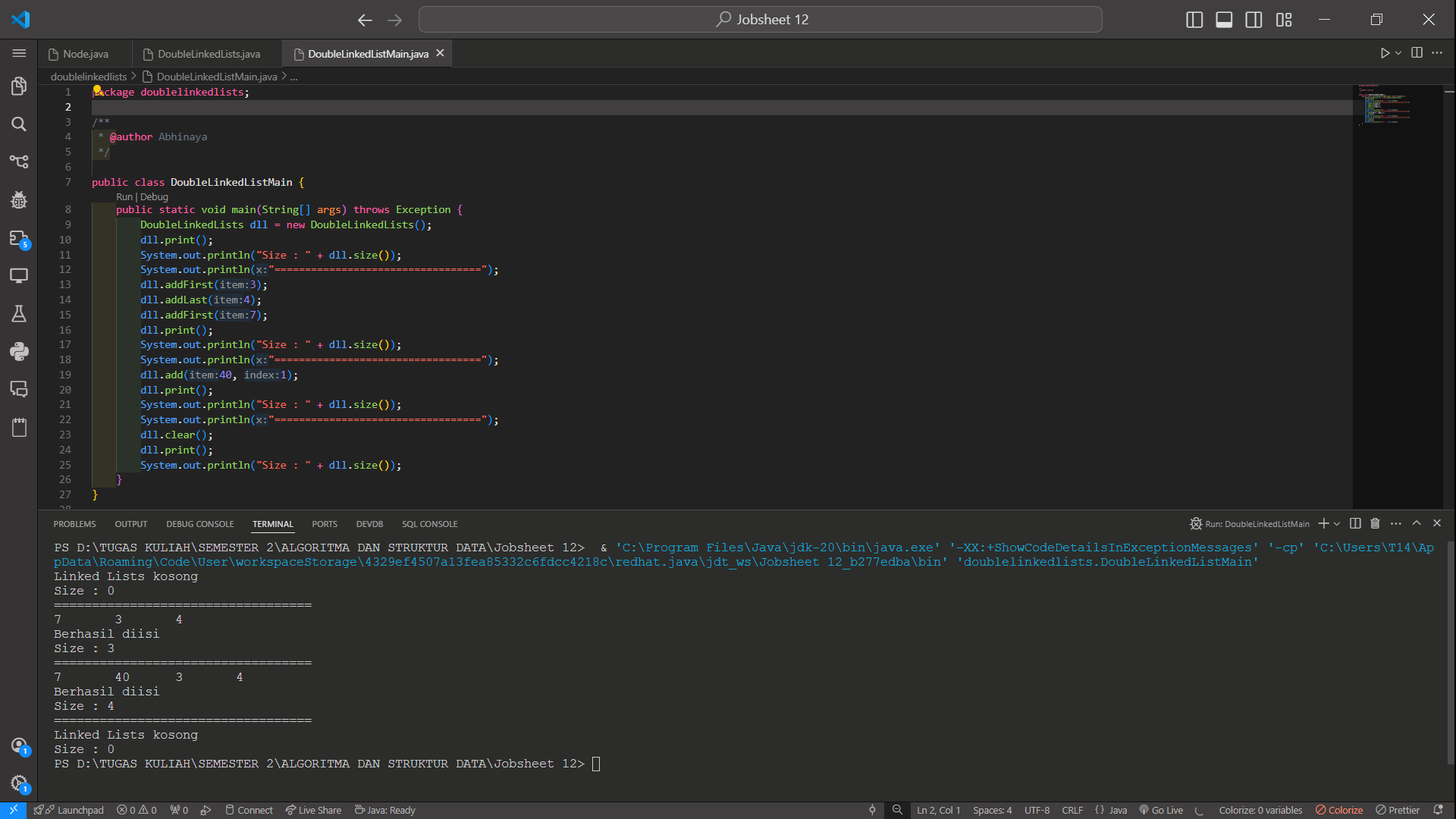
    }

}

Hasilnya akan Error



Agar tidak error bisa dirubah

public static void main(String[] args) throws Exception {

