NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

2. 1 Percobaan 1

```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;
    }
}
```

```java
package P12;

public class DoubleLinkedLists {
    Node head;
    int size;

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

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

    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++;
    }

    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++;
        }
    }
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

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

int size() {
    return size;
}

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

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 List Kosong");
    }
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

```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());
        System.out.println(x:"==================================");
    }
}
```

```
Linked List Kosong
Size : 0
==================================
7       3       4
Berhasil diisi
Size : 3
==================================
7       40      3       4
Berhasil diisi
Size : 4
==================================
Linked List Kosong
Size : 0
```

Pertanyaan :

1. *Jelaskan perbedaan antara single linked list dengan double linked lists!*
   Jawab : Pada single linked list node hanya dapat bergerak satu arah(next), dan tidak bisa kembali(previous)

2. *Perhatikan class Node, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?*
   Jawab : Sebagai penunjuk node setelah dan sebelumnya, sehingga struktur data dapat dimanipulasi.

3. *Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan inisialisasi atribut head dan size seperti pada gambar berikut ini?*

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

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

Jawab : inisialisasi head = null untuk menunjukkan bahwa linkedlist yang dibuat adalah linkedlist kosong.

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 jika kita menambahkan objek yang berada di depan maka previous atau node sebelumnya dari node tersebut adalah null, bukan node yang lain

5. *Perhatikan pada method addFirst(). Apakah arti statement head.prev = newNode ?*

   Jawab : Node baru ditempatkan di depan Head, sehingga node baru berada di posisi paling depan

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 : Pengisian parameter tersebut ditujukan agar node current atau yang saat ini ditunjuk menjadi node yang berada di depan node baru, dan node setelah node baru adalah null karena ia adalah node terakhir

7. *Pada method add(), terdapat potongan kode program sebagai berikut:*

```
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;
}
```

   jelaskan maksud dari bagian yang ditandai dengan kotak kuning.

   Jawab : Jika di depan node current adalah null, maka lakukan add dengan menambahkan node baru dengan prev null(berada di depan), dan node selanjutnya adalah node current atau node pertama yang ditunjuk

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

2.2 Percobaan 2

```
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--;
    }
}

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--;
}
```

```
void remove(int index) throws Exception {
    if (isEmpty() || index >= size) {
        throw new Exception(message:"Nilai indeks 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 : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

```
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();
System.out.println("Size : " + dll.size());
System.out.println(x:"==================================");
dll.remove(index:1);
dll.print();
System.out.println("Size : " + dll.size());
System.out.println(x:"==================================");
```

```
50      40      10      20
Berhasil diisi
Size : 4
==================================
40      10      20
Berhasil diisi
Size : 3
==================================
Size : 2
==================================
40
Berhasil diisi
Size : 1
==================================
```

Pertanyaan :

1. *Apakah maksud statement berikut pada method removeFirst()?*
   head = head.next;
   head.prev = null
   Jawab : Memindahkan head data pada node di belakangnya, dan menjadikan node previous pada head baru menjadi null

2. *Bagaimana cara mendeteksi posisi data ada pada bagian akhir pada method removeLast()?*
   Jawab :

```
while (current.next.next != null) {
    current = current.next;
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

Dengan menggunakan perulangan dan Node temp(current) sehingga dapat berpindah tempat dan berhenti ketika 2 node setelahnya adalah null, dan dapat menghapus node setelah current yang merupakan data akhir

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 : Karena kode program tersebut hanya digunakan untuk menghapus data pada indeks setelah head atau indeks 1 saja

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

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

Jawab : Mengganti pointer pada node sebelum dan setelah node yang akan dihapus sehingga melewati node yang dihapus

2.3 Percobaan 3

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

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;
}

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

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

```
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));
```

```
Linked List 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:\Vian\kuliah\sem-2\AlgoritmaStrukturData>
```

Pertanyaan :

1. *Jelaskan method size() pada class DoubleLinkedLists!*
   Jawab : Untuk memberikan return berupa int jumlah data yang disimpan pada linked list

2. *Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke-1!*
   Jawab : Untuk mengganti indeks cukup dengan mengganti value size ketika pertama kali di deklarasi menjadi 1

3. Jelaskan perbedaan karakteristik fungsi Add pada Double Linked Lists dan Single Linked Lists!
   Jawab : Fungsi add pada double linked list dapat dilakukan melalui depan ataupun belakang mengingat double linked list yang dapat bergerak dua

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : Double Linked List

arah. Sedangkan single linked list hanya dapat menggunakan fungsi traversal sehingga harus mencari dari depan ke belakang

4. *Jelaskan perbedaan logika dari kedua kode program di bawah ini!*

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

(a)

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

(b)

Jawab : Perbedaan kedua kode program tersebut hanya terdapat pada penggunaan variabel dan jumlah baris program.
Kode program a akan melakukan pengecekan pada variabel size, jika size bervalue 0 maka akan menghasilkan true, dan menghasilkan false jika size tidak sama dengan 0.
Sedangkan kode program b akan melakukan pengecekan pada variabel head, jika head memiliki value null, maka akan menghasilkan true, jika head tidak sama dengan null maka akan menghasilkan false