NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

Percobaan 1

Node15.java

```java
package P11;

public class Node15 {
    int data;
    Node15 prev, next;

    Node15(Node15 prev, int data, Node15 next) {
        this.prev = prev;
        this.data = data;
        this.next = next;
    }
}
```

```java
package P11;

public class DoubleLinkedList {
    Node15 head;
    int size;

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

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

    public void addFirst(int item) {
        if (isEmpty()) {
            head = new Node15(prev:null, item, next:null);
        } else {
            Node15 newNode = new Node15(prev:null, item, head);
            head.prev = newNode;
            head = newNode;
        }
        size++;
    }

    public void addLast(int item) {
        if (isEmpty()) {
            addFirst(item);
        } else {
            Node15 current = head;
            while (current.next != null) {
                current = current.next;
            }
            Node15 newNode = new Node15(current, item, next:null);
            current.next = newNode;
            size++;
        }
    }
```

```java
    public 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 {
            Node15 current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }
            if (current.prev == null) {
                Node15 newNode = new Node15(prev:null, item, current);
                current.prev = newNode;
                head = newNode;
            } else {
                Node15 newNode = new Node15(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()) {
            Node15 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");
        }
    }
}
```

NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

```
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:\Kuliah\Semester 2\Praktek Algoritma Struktur Data\Algoritma_Struktur_Data_1G_15>
```

Pertanyaan

1. Jelaskan perbedaan antara single linked list dengan double linked lists!

   Jawab : Single linked list merupakan linked list yang hanya mempunyai satu variabel pointer yaitu pointer yang menunjuk node selanjutnya, sedangkan double linked list merupakan linked list yang mempunyai dua variabel pointer yaitu pointer yang menunjuk ke node selanjutnya dan pointer yang menunjuk ke node sebelumnya.

2. Perhatikan class Node, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?

   Jawab : Atribut next menunjuk ke node berikutnya dalam double linked list, memungkinkan traversal maju, dan atribut prev menunjuk ke node sebelumnya dalam double linked list, memungkinkan traversal mundur.

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 : Fungsi dari menginisialisasi head dengan nilai null menandakan bahwa linked list yang baru dibuat masih kosong dan belum ada node di dalamnya, dan fungsi menginisialisasi size dengan nilai 0 menandakan bahwa jumlah elemen dalam linked list adalah nol. Oleh karena itu, inisialisasi atribut head dan size dalam konstruktor kelas DoubleLinkedList bertujuan untuk memastikan bahwa linked list baru dimulai dalam keadaan kosong dan siap untuk digunakan tanpa error, menjadikan linked list ini mudah dipelihara dan dioperasikan.

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 : Pada method addFirst() atribut prev diatur ke null dalam konstruktor class Node ketika menambahkan node baru di awal linked list karena node tersebut akan menjadi node pertama dalam list. Sebagai node pertama, ia tidak memiliki node sebelumnya, sehingga prev harus null. Ini menjaga

NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

konsistensi struktur double linked list, di mana node pertama selalu memiliki prev yang menunjuk ke null.

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

   Jawab : Pada metode addFirst(), pernyataan head.prev = newNode digunakan untuk menetapkan referensi node sebelumnya dari node pertama dalam linked list ke node baru yang baru saja ditambahkan.

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 : Pada method addLast() pembuatan object Node dengan mengisikan parameter prev dengan current, dan next dengan null digunakan untuk membuat node baru yang akan ditambahkan di akhir linked list dengan prev menunjuk ke node terakhir saat ini (current) dan next diatur ke null karena node baru ini akan menjadi node terakhir dalam list.

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 : Kode tersebut menambahkan node baru di awal double linked list. Jika current adalah node pertama (head) dari linked list, maka node baru (newNode) dibuat dan ditempatkan di depan current. Setelah itu, newNode diatur sebagai head baru dari linked list.

NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

Percobaan 2

```
88      void removeFirst() throws Exception {
89          if (isEmpty()) {
90              throw new Exception(message:"LinkedList masih kosong, tidak dapat dihapus");
91          } else if (size == 1) {
92              removeLast();
93          } else {
94              head = head.next;
95              head.prev = null;
96              size--;
97          }
98      }
99
100     void removeLast() throws Exception {
101         if (isEmpty()) {
102             throw new Exception(message:"Linked List masih kosong, tidak dapat dihapus");
103         } else if (head.next == null){
104             head = null;
105             size--;
106             return;
107         }
108         Node15 current = head;
109         while (current.next.next != null) {
110             current = current.next;
111         }
112         current.next = null;
113         size--;
114     }
```

```
116     void remove(int index) throws Exception {
117         if (isEmpty() || index >= size) {
118             throw new Exception(message:"Nilai indeks di luar batas");
119         } else if (index == 0) {
120             removeFirst();
121         } else {
122             Node15 current = head;
123             int i = 0;
124             while (i < index) {
125                 current = current.next;
126                 i++;
127             }
128             if (current.next == null) {
129                 current.prev.next = null;
130             } else if (current.prev == null) {
131                 current = current.next;
132                 current.prev = null;
133                 head = current;
134             } else {
135                 current.prev.next = current.next;
136                 current.next.prev = current.prev;
137             }
138             size--;
139         }
140     }
141
142 }
143
```

NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

```
23          dll.addLast(item:50);
24          dll.addLast(item:40);
25          dll.addLast(item:10);
26          dll.addLast(item:20);
27          dll.print();
28          System.out.println("Size : " + dll.size());
29          System.out.println(x:"====================================");
30          dll.removeFirst();
31          dll.print();
32          System.out.println("Size : " + dll.size());
33          System.out.println(x:"====================================");
34          dll.removeLast();
35          dll.print();
36          System.out.println("Size : " + dll.size());
37          System.out.println(x:"====================================");
38          dll.remove(index:1);
39          dll.print();
40          System.out.println("Size : " + dll.size());
41      }
42  }
43
```

```
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:\Kuliah\Semester 2\Praktek Algoritma Struktur Data\Algoritma_Struktur_Data_1G_15>
```

Pertanyaan

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

   head = head.next;

   head.prev = null;

   jawab : Pada method removeFirst() dalam kelas DoubleLinkedList, pernyataan head = head.next dan head.prev = null digunakan untuk menghapus node pertama dari linked list, dimana pernyataan head = head.next digunakan untuk menggeser head ke node kedua dalam linked list setelah node pertama dihapus, sedangkan pernyataan head.prev = null digunakan untuk memastikan bahwa prev dari node baru yang menjadi head diatur ke null untuk menjaga konsistensi struktur double linked list

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

   Jawab : Untuk mendeteksi posisi data pada bagian akhir dalam metode removeLast(), kita menggunakan iterasi melalui linked list sampai kita mencapai node kedua terakhir. Setelah kita menemukan node kedua terakhir, kita mengatur next dari node tersebut menjadi null, sehingga node terakhir dihapus dari linked list.

   Memeriksa apakah linked list kosong

NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

```
if (isEmpty()) {
    throw new Exception(message:"Linked List masih kosong, tidak dapat dihapus");
```

Memeriksa apakah head adalah node terakhir

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

Iterasi untuk menemukan node terakhir sebelum node terakhir

```
Node15 current = head;
while (current.next.next != null) {
    current = current.next;
}
```

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 : Potongan kode program tersebut tidak cocok untuk perintah remove karena potongan kode tersebut tidak memperhatikan situasi dimana node yang akan dihapus adalah node terakhir dalam linked list sehingga dapat menyebabkan error atau kesalahan dalam manipulasi linked list.

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

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

Jawab : Potongan kode tersebut digunakan untuk menghapus node yang terletak di posisi index dari linked list. Perintah current.prev.next = current.next; menghubungkan node sebelumnya dengan node setelahnya, dan current.next.prev = current.prev; menghubungkan node setelahnya dengan node sebelumnya. Dengan demikian, node current dihapus dari linked list.

NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

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");
        }
        Node15 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.");
        }
        Node15 tmp = head;
        for (int i = 0; i < index; i++) {
            tmp = tmp.next;
        } return tmp.data;
    }

}
```

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

```
Linked Lists Kosong
Size : 0
================================================
7       3       4
Berhasil diisi
Size : 3
================================================
7       40      3       4
Berhasil diisi
Size : 4
================================================
Data awal pada Linked Losts adalah : 7
Data akhir pada Linked Lists adalah : 4
Data indeks ke-1 pada Linked Lists adalah : 40
================================================
PS D:\Kuliah\Semester 2\Praktek Algoritma Struktur Data\Algoritma_Struktur_Data_1G_15>
```

Pertanyaan

1. Jelaskan method size() pada class DoubleLinkedLists!

   Jawab : Method size() pada kelas DoubleLinkedList mengembalikan jumlah elemen dalam linked list, yang disimpan dalam variabel size. Ini berguna untuk mengetahui ukuran linked list pada saat tertentu.

NAMA : HIZKIA ELSADANTA
NIM : 2341720253
KELAS : TI-1G
MATERI : DOUBLE LINKED LIST

2. Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke1!

   Jawab : Untuk memulai indeks pada double linked lists dari 1, kita dapat mengubah implementasi metode add(int item, int index) dengan menambahkan 1 pada nilai index yang diterima sebagai argumen. Dengan ini, indeks akan dimulai dari 1 saat diakses, meskipun secara internal dimulai dari 0.

3. Jelaskan perbedaan karakteristik fungsi Add pada Double Linked Lists dan Single Linked Lists!

   Jawab : Pada double linked lists, saat menambahkan node baru, baik node sebelumnya maupun node berikutnya akan diperbarui, memungkinkan akses ke node-node dalam kedua arah. Sementara pada single linked lists, hanya node berikutnya yang diperbarui, membatasi kemampuan akses ke depan saja dalam linked list.

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 : Kode program (a) memeriksa apakah variabel size sama dengan 0 untuk menentukan apakah struktur data kosong, sedangkan kode program (b) memeriksa apakah variabel head adalah null. Kode program (b) lebih langsung dan jelas dalam menentukan kekosongan struktur data.

Link github :

https://github.com/Kiaakk/Algoritma_Struktur_Data_1G_15.git