JOBSHEET 10

Nama : Farid Aziz Wicaksono
Kelas : TI/1C
Absen : 14

A. PRAKTIKUM

No	Node.java
1	package minggu10;
2	
3	public class Node {
4	int data;
5	Node next;
6	
7	public Node(int data, Node next){
8	this.data = data;
9	this.next = next;
10	}
11	}

No	LinkedList.java
1	package minggu10;
2	
3	public class LinkedList {
4	Node head;
5	int size;
6	
7	public LinkedList() {
8	head = null;
9	size = 0;
10	}
11	
12	public boolean isEmpty() {
13	return head == null;
14	}
15	
16	public void addFirst(int item) {
17	head = new Node(item, head);
18	size++;
19	}
20	
21	public void add(int item, int index) throws Exception {
22	if (index < 0 || index > size) {
23	throw new Exception("Nilai index di luar batas");
24	}
25	if (isEmpty() || index == 0) {
26	addFirst(item);
27	} else {

```
            Node tmp = head;
            for (int i = 1; i < index; i++) {
                tmp = tmp.next;
            }
            Node next = (tmp == null) ? null : tmp.next;
            tmp.next = new Node(item, next);
        }
        size++;
    }

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

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

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

    public void remove(int index) throws Exception {
```

```
        if (isEmpty() || index >= size) {
            throw new Exception("Nilai index di luar batas");
        }
        if(!isEmpty()){
            Node tmp = head;
            head = tmp.next;
            tmp = null;
        }
        else{
            Node prev = head;
            Node cur = head.next;
            for (int i = 1; i < index; i++) {
                prev = cur;
                cur = prev.next;
            }
            prev.next = cur.next;
        }
        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();
        }
        else{
            System.out.println("LinkedList kosong");
        }
    }
}
```

No	LinkedListTest.java
1	package minggu10;
2	
3	public class LinkedListTest {
4	public static void main(String[] args) {
5	LinkedList data = new LinkedList();
6	
7	try{
8	data.addFirst(1);
9	data.print();
10	data.add(1, 2);
11	data.clear();
12	data.addFirst(1);
13	data.remove(0);
14	data.print();
15	}
16	catch(Exception e){
17	System.out.println(e.getMessage());
18	}
19	}
20	}

Output :

```
run:
1
Nilai index di luar batas
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. PERTANYAAN

1. Mengapa pada proses traverse nilai head perlu disimpan terlebih dahulu dalam variable tmp ?

   Jawab : Karena jika nilai head tidak di simpan di dalam tmp, maka data akan berubah dengan data inputan selanjutnya

2. Apa kekurangan implementasi single linked list tanpa petunjuk tail ?

   Jawab : Karena tail untuk menunjukkan node terakhir, tanpa adanya tail, maka node terakhir tidak akan terbaca

3. Tambahkan implementasi method add berdasarkan nilai yang dicari! Node baru akan ditambahkan setelah node yang ditemukan.

   Jawab :

```
public void addKey(int item) throws Exception{
    if (isEmpty()) {
        throw new Exception("Data Kosong!");
    }

    Node tmp = head;
    while(tmp != null){
     if (item == tmp.data){ while
        (tmp.next != null){
        tmp = tmp.next;
        }
        tmp.next = new Node(item, null);
        size++;
        break;
    }
   }
}
```

4. Tambahkan implementasi method remove berdasarkan nilai yang dicari !
   Jawab :

```
public void removebykey(int data) throws Exception{
    Node prev =head;
    Node cur = head.next;

    for (int i =0 ; i <=size ; i++ ){
        if(get(i)==data){
            int x = i;
            for(int a = 1; a < x; a++){
                prev = cur;
                cur = prev.next;
            }
            prev.next = cur.next;
            size--;
        }
    }
}
```

5. Tambahkan menu serta submenu dan input dinamis pada program percobaan tersebut!

Menu	Submenu	Submenu	Submenu	Submenu
Tambah	Tambah (First)	Tambah (index)	Tambah (key)	Tambah (Last)
Hapus	Hapus (index)	Hapus (key)	Clear	
Cari	Cari (index)	Cari (key)		
Keluar				

   Jawab :

```
   public static void menu() {
        System.out.println("*============================*");
        System.out.println("PILIHAN MENU");
        System.out.println("1. Tambah");
        System.out.println("2. Hapus");
        System.out.println("3. Cari");
        System.out.println("0. Keluar");
        System.out.println("*============================*");
      }

      public static void tambah() {
        System.out.println("*============================*");
        System.out.println("SUB MENU |");
        System.out.println("1. Tambah(First)");
        System.out.println("2. Tambah(Index)");
```

```
            System.out.println("3. Tambah(Key)");
            System.out.println("4. Tambah(Last)");
            System.out.println("0. Kembali");
            System.out.println("*============================*");
        }

        public static void hapus() {
            System.out.println("*============================*");
            System.out.println("PILIHAN MENU");
            System.out.println("1. Hapus(Index)");
            System.out.println("2. Hapus(Key)");
            System.out.println("3. Clear");
            System.out.println("0. Kembali");
            System.out.println("*============================*");
        }

        public static void cari() {
            System.out.println("*============================*");
            System.out.println("PILIHAN MENU");
            System.out.println("1. Cari(Index)");
            System.out.println("2. Cari(Key)");
            System.out.println("0. Kembali");
            System.out.println("*============================*");
    }
```

Main

```
package minggu10;
import java.util.*;
public class p5m {
    public static void main(String[] args) {
        Scanner sd = new Scanner(System.in);
        int pilih, sub;
        int dat, idx;
        LinkedList a = new LinkedList();
        try {
            do {
                a.menu();
                System.out.print("Masukkan Pilihan Anda :");
                pilih = sd.nextInt();
                switch (pilih) {
                    case 1:
                        do {
                            a.tambah();
                            System.out.print("Masukkan Pilihan Anda : ");
                            sub = sd.nextInt();
                            switch (sub) {
```

```
                    case 1:
                        System.out.println("|Masukkan Data : ");
                        dat = sd.nextInt();

System.out.println("*================================*");
                        a.addFirst(dat);
                        a.print();
                        break;
                    case 2:
                        System.out.println("|Masukkan Data : ");
                        dat = sd.nextInt();
                        System.out.println("|Masukkan Index : ");
                        idx = sd.nextInt();


System.out.println("*================================*");
                        a.add(dat, idx);
                        a.print();
                        break;
                    case 3:
                        System.out.println("|Masukkan Data : ");
                        dat = sd.nextInt();
                        a.addFirst(dat);
                        a.print();
                        break;
                    case 4:
                        System.out.println("|Masukkan Data : ");
                        dat = sd.nextInt();
                        a.addLast(dat);
                        a.print();
                        break;
                    }
                } while (sub != 0);
                break;
            case 2:
                do {
                    a.hapus();
                    System.out.print("Masukkan Pilihan Anda : ");
                    sub = sd.nextInt();
                    switch (sub) {
                        case 1:
                            System.out.println("|Masukkan Indeks : ");
                            idx = sd.nextInt();
                            a.remove(idx);
                            a.print();
                            break;
```

```
                    case 2:
                        a.removeFirst();
                        a.print();
                        break;
                    case 3:
                        a.clear();
                        a.print();
                        break;
                }
            } while (sub != 0);
            break;
        case 3:
            do {
                a.cari();
                System.out.print("Masukkan Pilihan Anda : ");
                sub = sd.nextInt();
                switch (sub) {
                    case 1:
                        System.out.println("|Masukkan Indeks : ");
                        idx = sd.nextInt();
                        a.remove(idx);
                        a.print();
                        break;
                    case 2:
                        System.out.println("|Masukkan Data : ");
                        dat = sd.nextInt();

                        a.removeFirst();
                        a.print();
                        break;
                }
            } while (sub != 0);
            break;
        }
    } while (pilih != 0);
  } catch (Exception e) {
    System.out.println(e.getMessage());
  }
 }
}
```

Output :

```
*==============================*
PILIHAN MENU
1. Tambah
2. Hapus
3. Cari
0. Keluar
*==============================*
Masukkan Pilihan Anda :1
*==============================*
SUB MENU |
1. Tambah(First)
2. Tambah(Index)
3. Tambah(Key)
4. Tambah(Last)
0. Kembali
*==============================*
Masukkan Pilihan Anda : 1
|Masukkan Data :
2
*================================*
2
*==============================*
SUB MENU |
1. Tambah(First)
2. Tambah(Index)
3. Tambah(Key)
4. Tambah(Last)
0. Kembali
*==============================*
Masukkan Pilihan Anda : 3
|Masukkan Data :
4
4       2
```

```
*==============================*
SUB MENU |
1. Tambah(First)
2. Tambah(Index)
3. Tambah(Key)
4. Tambah(Last)
0. Kembali
*==============================*
Masukkan Pilihan Anda : 0
*==============================*
PILIHAN MENU
1. Tambah
2. Hapus
3. Cari
0. Keluar
*==============================*
Masukkan Pilihan Anda :3
*==============================*
PILIHAN MENU
1. Cari(Index)
2. Cari(Key)
0. Kembali
*==============================*
Masukkan Pilihan Anda : 2
|Masukkan Data :
1
2
*==============================*
PILIHAN MENU
1. Cari(Index)
2. Cari(Key)
0. Kembali
*==============================*
Masukkan Pilihan Anda : 1
|Masukkan Indeks :
2
Nilai index di luar batas
```

C. TUGAS

1. Buatlah implementasi program daftar mahasiswa menggunakan LinkedList! Mahasiswa memiliki atribut NIM serta nama.

   Jawab :

No	NodeMahasiswa.java
1	package minggu10;
2	
3	public class NodeMahasiswa {
4	int data;
5	int nim;
6	String nama;
7	NodeMahasiswa next;
8	
9	public NodeMahasiswa(int nim, String nama, NodeMahasiswa next) {
10	this.next = next;
11	this.nama = nama;
12	this.nim = nim;
13	}
14	}

No	SingleLinkedListmhs.java
1	package minggu10;
2	
3	public class SingleLinkedListmhs {
4	NodeMahasiswa head;
5	int size;
6	
7	public SingleLinkedListmhs() {
8	head = null;
9	size = 0;
10	}
11	
12	public boolean isEmpty() {
13	return head == null;
14	}
15	
16	public void addFirst(int nim, String nama) {
17	head = new NodeMahasiswa(nim, nama, head);
18	size++;
19	}
20	
21	public void add(int nim, String nama, int index) throws Exception {
22	if (index < 0 || index > size) {
23	throw new Exception("Nilai index di luar batas!");
24	}
25	if (isEmpty() || index == 0) {

```
            addFirst(nim, nama);
        } else {
            NodeMahasiswa tmp = head;
            for (int i = 0; i < index; i++) {
                tmp = tmp.next;
            }
            NodeMahasiswa next = (tmp == null) ? null : tmp.next;
            tmp.next = new NodeMahasiswa(nim, nama, next);
        }
        size++;
    }

    public void addLast(int nim, String nama) {
        if (isEmpty()) {
            addFirst(nim, nama);
        } else {
            NodeMahasiswa tmp = head;
            while (tmp.next != null) {
                tmp = tmp.next;
            }
            tmp.next = new NodeMahasiswa(nim, nama, null);
            size++;
        }
    }

    public int getFirstNim() throws Exception {
        if (isEmpty()) {
            throw new Exception("LinkedList Kosong");
        }
        return head.nim;
    }

    public String getFirstNama() throws Exception {
        if (isEmpty()) {
            throw new Exception("LinkedList Kosong");
        }
        return head.nama;
    }

    public int getLastNim() throws Exception {
        if (isEmpty()) {
            throw new Exception("LinkedList Kosing");
        }
        NodeMahasiswa tmp = head;
        while (tmp.next != null) {
            tmp = tmp.next;
```

```
        }
        return tmp.nim;
    }

    public String getLastNama() throws Exception {
        if (isEmpty()) {
            throw new Exception("LinkedList Kosing");
        }
        NodeMahasiswa tmp = head;
        while (tmp.next != null) {
            tmp = tmp.next;
        }
        return tmp.nama;
    }

    public int getNim(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception("Nilai index di luar batas");
        }
        NodeMahasiswa tmp = head;
        for (int i = 0; i < index; i++) {
            tmp = tmp.next;
        }
        return tmp.nim;
    }

    public String getNama(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception("Nilai index di luar batas");
        }
        NodeMahasiswa tmp = head;
        for (int i = 0; i < index; i++) {
            tmp = tmp.next;
        }
        return tmp.nama;
    }

    public void remove(int index) throws Exception {
        if (isEmpty() || index >= size) {
            throw new Exception("Nilai index di luar batas");
        }
    }

    public void removeFirstNim() throws Exception {
        int tmp = getFirstNim();
        head = head.next;
```

```
        size--;
    }

    public void removeFirstNama() throws Exception {
        String tmp = getFirstNama();
        head = head.next;
        size--;
    }

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

    public void print() {
        if (!isEmpty()) {
            NodeMahasiswa tmp = head;
            while (tmp != null) {
                System.out.println("==================");
                System.out.println("NIM: " + tmp.nim);
                System.out.println("Nam: " + tmp.nama);
                tmp = tmp.next;
            }
            System.out.println("");
        } else {
            System.out.println("LinkedList kosong");
        }
    }
}
```

No	MainMahasiswa.java

```
package minggu10;
public class MainMahasiswa {
    public static void main(String[] args) {
        SingleLinkedListmhs data = new SingleLinkedListmhs();
        try {
            data.addFirst(1841720001, "andi");
            data.add(1841720002, "anda", 0);
            data.add(1841720003, "farid", 0);
            data.addLast(1841720004, "aziz");
            data.print();
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

Output :

```
run:
===================
NIM: 1841720003
Nam: farid
===================
NIM: 1841720002
Nam: anda
===================
NIM: 1841720001
Nam: andi
===================
NIM: 1841720004
Nam: aziz

BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Implementasikan Stack atau Queue (pilih salah satu dengan menggunakan konsep LinkedList!

Jawab :

No	NodeQueue.java
1	package minggu10;
2	
3	public class NodeQueue {
4	NodeQueue next;
5	int data;
6	
7	public NodeQueue(int data) {
8	this.data = data;
9	this.next = null;
10	}
11	}

No	LinkedListQueue.java
1	package minggu10;
2	
3	public class LinkedListQueue {
4	NodeQueue head, tail;
5	
6	public LinkedListQueue() {
7	this.head = this.tail = null;
8	}
9	
10	void enqueue(int data) {
11	NodeQueue tmp = new NodeQueue(data);
12	if (this.tail == null) {

```
            this.head = this.tail = tmp;
            return;
        }
        this.tail.next = tmp;
        this.tail = tmp;
    }

    NodeQueue dequeue() {
        if (this.head == null) {
            return null;
        }
        NodeQueue tmp = this.head;
        this.head = this.head.next;
        if (this.head == null) {
            this.tail = null;
        }
        return tmp;
    }

    void print() {
        NodeQueue tmp = head;
        while (tmp != null) {
            System.out.println(+tmp.data + "/");
            tmp = tmp.next;
        }
        System.out.println();
    }
}
```

No	MainQueue.java

```
package minggu10;

public class MainQueue {
    public static void main(String[] args) {
        LinkedListQueue Q = new LinkedListQueue();
        Q.enqueue(0);
        Q.enqueue(1);
        Q.enqueue(2);
        Q.enqueue(3);
        Q.print();
        Q.dequeue();
        Q.print();
    }
}
```

Output :

```
run:
0
1
2
3

1
2
3

BUILD SUCCESSFUL (total time: 0 seconds)
```