NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

2. 1 Percobaan 1

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

```java
package P11;

public class Node {
    int data;
    Node next;

    Node(int nilai, Node berikut) {
        data = nilai;
        next = berikut;
    }
}

```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

```java
package P11;

public class SingleLinkedList {
    Node head, tail;

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

    void print() {
        if (!isEmpty()) {
            Node tmp = head;
            System.out.print(s:"Isi Linked List : ");
            while (tmp != null) {
                System.out.print(tmp.data + "\t");
                tmp = tmp.next;
            }
            System.out.println();
        } else {
            System.out.println(x:"Linked List Kosong.");
        }
    }

    void addFirst(int input) {
        Node ndInput = new Node(input, head);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
            // ndInput.next = head;
            // head = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }

    void addLast(int input) {
        Node ndInput = new Node(input, berikut:null);
        if (isEmpty()) {
            head = ndInput;
            // tail.next = ndInput;
            tail = ndInput;
        } else {
            tail.next = ndInput;
            tail = tail.next;
        }
    }
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

```
void insertAfter(int key, int input) {
    Node ndInput = new Node(input, berikut:null);
    Node temp = head;
    do {
        if (temp.data == key) {
            ndInput.next = temp.next;
            temp.next = ndInput;
            if (ndInput.next != null) {
                tail = ndInput;
                break;
            }
        }
        temp = temp.next;
    } while (temp == null);
}

void insertAt(int index, int input) {
    //Node ndInput = new Node(input, null);
    if (index < 0) {
        System.out.println(x:"Masukkan index dengan benar.");
    } else if (index == 0) {
        addFirst(input);
    } else {
        Node temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new Node(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

```
package P11;

public class SLLMain {
    Run | Debug
    public static void main(String[] args) {
        SingleLinkedList singLL = new SingleLinkedList();

        singLL.print();
        singLL.addFirst(input:890);
        singLL.print();
        singLL.addLast(input:760);
        singLL.print();
        singLL.addFirst(input:700);
        singLL.print();
        singLL.insertAfter(key:700, input:999);
        singLL.print();
        singLL.insertAt(index:3, input:833);
        singLL.print();
    }
}
```

```
Linked List Kosong.
Isi Linked List : 890
Isi Linked List : 890   760
Isi Linked List : 700   890     760
Isi Linked List : 700   999     890     760
Isi Linked List : 700   999     890     833     760
```

Pertanyaan :

1. *Mengapa hasil compile kode program di baris pertama menghasilkan "Linked List Kosong"?*
   Jawab : Karena method isEmpty() memberikan return true apabila head tidak null, sedangkan kondisi kosong terjadi ketika head adalah null.

2. *Jelaskan kegunaan variable temp secara umum pada setiap method!*
   Jawab : Temp digunakan untuk berpindah tempat dari satu node ke node berikutnya, sehingga dapat memberikan hasil yang dibutuhkan

3. *Perhatikan class SingleLinkedList, pada method insertAt Jelaskan kegunaan kode berikut*

   ```
   if(temp.next.next==null) tail=temp.next;
   ```

   Jawab : Kode digunakan untuk menentukan tail dari LinkedList, sehingga tail adalah node temp yang berada sebelum null. Jika 2 langkah setelah temp adalah null, maka 1 langkah di depan temp adalah tail.

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

2.2 Percobaan 2

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

```
int getData(int index) {
    Node tmp = head;
    for (int i = 0; i < index - 1; i++) {
        tmp = tmp.next;
    }
    return tmp.next.data;
}

int indexOf(int key) {
    Node tmp = head;
    int index = 0;
    while (tmp != null && tmp.data != key) {
        tmp = tmp.next;
        index++;
    }
    if (tmp != null) {
        return 1;
    } else {
        return index;
    }
}

void removeFirst() {
    if (isEmpty()) {
        System.out.println(x:"Linked List Masih Kosong.");
    } else if (head == tail) {
        head = tail = null;
    } else {
        head = head.next;
    }
}

void removeLast() {
    if (isEmpty()) {
        System.out.println(x:"Linked List Masih Kosong.");
    } else if (head == tail) {
        head = tail = null;
    } else {
        Node temp = head;
        while (temp.next.next != null) {
            temp = temp.next;
        }
        temp.next = null;
        tail = temp;
    }
}
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

```
void remove(int key) {
    if (isEmpty()) {
        System.out.println(x:"Linked List Masih Kosong.");
    } else {
        Node temp = head;
        while (temp.next != null) {
            if (temp.data == key && temp == head) {
                removeFirst();
                break;
            } else if (temp.next.data == key) {
                temp.next = temp.next.next;
                if (temp.next == null) {
                    tail = temp;
                }
                break;
            }
            temp = temp.next;
        }
    }
}

void removeAt(int index) {
    if (index == 0) {
        removeFirst();
    } else {
        Node temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = temp.next.next;
        if (temp.next == null) {
            tail = temp;
        }
    }
}
```

```
Linked List Kosong.
Isi Linked List : 890
Isi Linked List : 890   760
Isi Linked List : 700   890     760
Isi Linked List : 700   999     890     760
Isi Linked List : 700   999     890     833     760
Data pada indeks ke-1 : 999
Data 3 berada pada indeks ke-1
Isi Linked List : 700   890     833     760
Isi Linked List : 890   833     760
Isi Linked List : 833   760
Isi Linked List : 833
```

NAMA : OKTAVIAN EKA RAMADHAN
NIM : 2341720117
KELAS : 1G
MATERI : LINKED LIST

Pertanyaan :

1. *Mengapa digunakan keyword break pada fungsi remove? Jelaskan!*
   Jawab : break diperlukan agar loop while dapat berhenti ketika key sudah ditemukan, sehingga program tidak menjalankan loop lagi.

2. *Jelaskan kegunaan kode dibawah pada method remove*

```
else if (temp.next.data == key) {
  temp.next = temp.next.next;
```

   Jawab : Kode tersebut dapat diartikan, jika data selanjutnya sama dengan key(data yang akan dihapus), maka ganti data selanjutnya menjadi data selanjutnya lagi. Sehingga data setelah temp akan langsung lompat ke 2 data selanjutnya dengan mengabaikan data yang dihapus atau data key.