NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Linked List

LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

2. 1. Pembuatan Single Linked List

Node.java

src > P11 > SingleLinkedList > Node13.java

```java
package P11.SingleLinkedList;

Codeium: Refactor | Explain
public class Node13 {
    int data;
    Node13 next;


    public Node13(int nilai, Node13 berikutnya) {
        data = nilai;
        next = berikutnya;
    }
}

```

SingelLinkList.java

src > P11 > SingleLinkedList > SingleLinkedList13.java > SingleLinkedList13 > print()

```java
package P11.SingleLinkedList;

Codeium: Refactor | Explain
public class SingleLinkedList13 {
    Node13 head, tail;

    Codeium: Refactor | Explain | Generate Javadoc | ×
    boolean isEmpty() {
        return head == null;
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    void print() {
        if (!isEmpty()) {
            Node13 tmp = head;
            System.out.print(s:"Isi Linked List:  ");
            while (tmp != null) {
                System.out.print(tmp.data + "\t");
                tmp = tmp.next;
            }
            System.out.println(x:"");
        } else {
            System.out.println(x:"Linked List kosong");
        }
    }

    Codeium: Refactor | Explain | Generate Javadoc | ×
    void addFirst(int input) {
        Node13 ndInput = new Node13(input, berikutnya:null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }

```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Linked List

```java
Codeium: Refactor | Explain | Generate Javadoc | ×
void insertAt(int index, int input) {
    Node13 ndInput = new Node13(input, berikutnya:null);
    if (index < 0) {
        System.out.println("perbaiki logikanya!"
                + "kalau indeksnya -1 bagaimana???");
    } else if (index == 0) {
        addFirst(input);
    } else {
        Node13 temp = head;
        for (int i = 0; i < index - 1; i++) {
            temp = temp.next;
        }
        temp.next = new Node13(input, temp.next);
        if (temp.next.next == null) {
            tail = temp.next;
        }
    }
}
}
```

```java
void addLast(int input) {
    Node13 ndInput = new Node13(input, berikutnya:null);
    if (isEmpty()) {
        head = ndInput;
        tail = ndInput;
    } else {
        tail.next = ndInput;
        tail = ndInput;
    }
}

Codeium: Refactor | Explain | Generate Javadoc | ×
void insertAfter(int key, int input) {
    Node13 ndinput = new Node13(input, berikutnya:null);
    Node13 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);
}
```

SLLMain.java

```java
src > P11 > SingleLinkedList > J SLLMain13.java > ...
package P11.SingleLinkedList;

Codeium: Refactor | Explain
public class SLLMain13 {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) {
        SingleLinkedList13 singLL = new SingleLinkedList13();
        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();
    }
}
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Linked List

Output :

```
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
PS E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```

Question :

1. Mengapa hasil compile kode program di baris pertama menghasilkan “Linked List Kosong”?
   Jawab : Karena data pada SingleLinkedList masih kosong dan pada kelas SLLMain sudah dipanggil atau di tampilkan terlebih dahulu, sehingga akan menghasilkan “Linked List Kosong”.

2. Jelaskan kegunaan variable temp secara umum pada setiap method!
   Jawab : Untuk menyimpan data sementara pada node dalam linked list tanpa harus merubah struktur data utamanya.

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

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

   Jawab : Untuk memastikan jika node baru dimasukkan dan menjadi node terakhir maka pointer tail diperbarui dan menunjukkan ke node yang baru dimasukkan.

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Linked List

2.2 Modifikasi Elemen pada Single Linked List

```
81  int getData(int index) {
82      Node13 tmp = head;
83      for (int i = 0; i < index - 1; i++) {
84          tmp = tmp.next;
85      }
86      return tmp.next.data;
87  }
88
    Codeium: Refactor | Explain | Generate Javadoc | ×
89  int indexOf(int key) {
90      Node13 tmp = head;
91      int index = 0;
92      while (tmp != null && tmp.data != key) {
93          tmp = tmp.next;
94          index++;
95      }
96      if (tmp == null) {
97          return 1;
98      } else {
99          return index;
100     }
101 }
102
    Codeium: Refactor | Explain | Generate Javadoc | ×
103 void removeFirst() {
104     if (isEmpty()) {
105         System.out.println("Linked List masih kosong,"
106                 + "tidak dapat dihapus");
107     } else if (head == tail) {
108         head = null;
109     } else {
110         head = head.next;
111     }
112 }
113
```

```
114 void removeLast() {
115     if (isEmpty()) {
116         System.out.println("Linked List masih kosong,"
117                 + "tidak dapat dihapus");
118     } else if (head == tail) {
119         tail = null;
120     } else {
121         Node13 temp = head;
122         while (temp.next == null) {
123             temp = temp.next;
124         }
125         temp.next = null;
126         tail = temp.next;
127     }
128 }
129
    Codeium: Refactor | Explain | Generate Javadoc | ×
130 void remove(int key) {
131     if (isEmpty()) {
132         System.out.println("Linked List masih kosong,"
133                 + "tidak dapat dihapus");
134     } else {
135         Node13 temp = head;
136         while (temp != null) {
137             if (temp.data == key && temp == head) {
138                 removeFirst();
139                 break;
140             } else if (temp.next.data == key) {
141                 temp.next = temp.next.next;
142                 if (temp.next == null) {
143                     tail = temp;
144                 }
145                 break;
146             }
147             temp = temp.next;
148         }
149     }
150 }
151
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Linked List

```java
    public void removeAt(int index) {
        if (index == 0) {
            removeFirst();
        } else {
            Node13 temp = head;
            for (int i = 0; i < index; i++) {
                temp = temp.next;
            }
            temp.next = temp.next.next;
            if (temp.next == null) {
                tail = temp;
            }
        }
    }
}
```

SLLMain.java

```java
package P11.SingleLinkedList;

Codeium: Refactor | Explain
public class SLLMain13 {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) {
        SingleLinkedList13 singLL = new SingleLinkedList13();
        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();

        System.out.println("Data pada indeks ke-1 = " + singLL.getData(index:1));
        System.out.println("Data 3 berada pada indeks ke- " + singLL.indexOf(key:760));

        singLL.remove(key:999);
        singLL.print();
        singLL.removeAt(index:0);
        singLL.print();
        singLL.removeFirst();
        singLL.print();
        singLL.removeLast();
        singLL.print();
    }
}
```

Output :

```
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- 4
Isi Linked List:  700   890     833     760
Isi Linked List:  890   833     760
Isi Linked List:  833   760
Isi Linked List:  833
PS E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```

Question :

1. Mengapa digunakan keyword break pada fungsi remove? Jelaskan!
   Jawab : Untuk menghentikan loop while jika data yang dicari ditemukan dan dihapus.

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Linked List

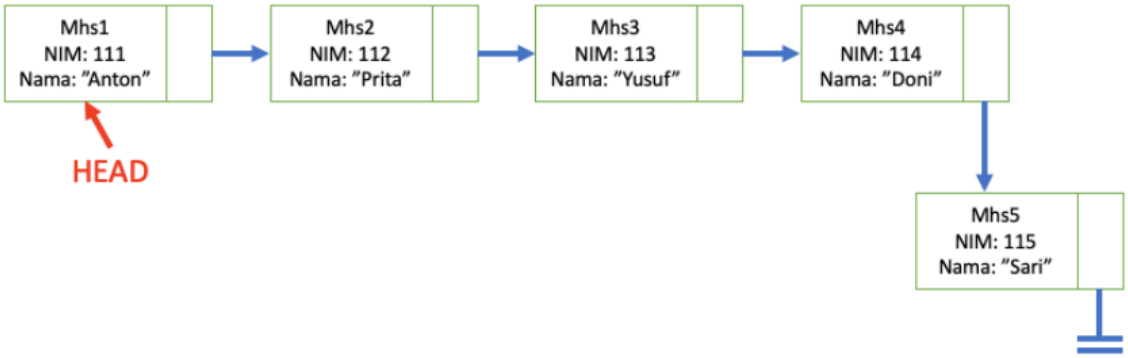
2. Jelaskan kegunaan kode dibawah pada method remove

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

Jawab : Kondisi jika key sesuai dan ditemukan maka node tersebut akan dihapus dan akan menghubungkan node sebelumnya dengan node setelah node yang dihapus.

3. Tugas

1. Implementasikan ilustrasi Linked List Berikut. Gunakan 4 macam penambahan data yang telah dipelajari sebelumnya untuk menginputkan data.



Jawab :

Objeck Mahasiswa :

```
src > P11 > Tugas_P11 > J Mahasiswa13.java > ...
1   package P11.Tugas_P11;
2
    Codeium: Refactor | Explain
3   public class Mahasiswa13 {
4       int nim;
5       String nama;
6
7       public Mahasiswa13(int nim, String nama) {
8           this.nim = nim;
9           this.nama = nama;
10      }
11  }
12
```

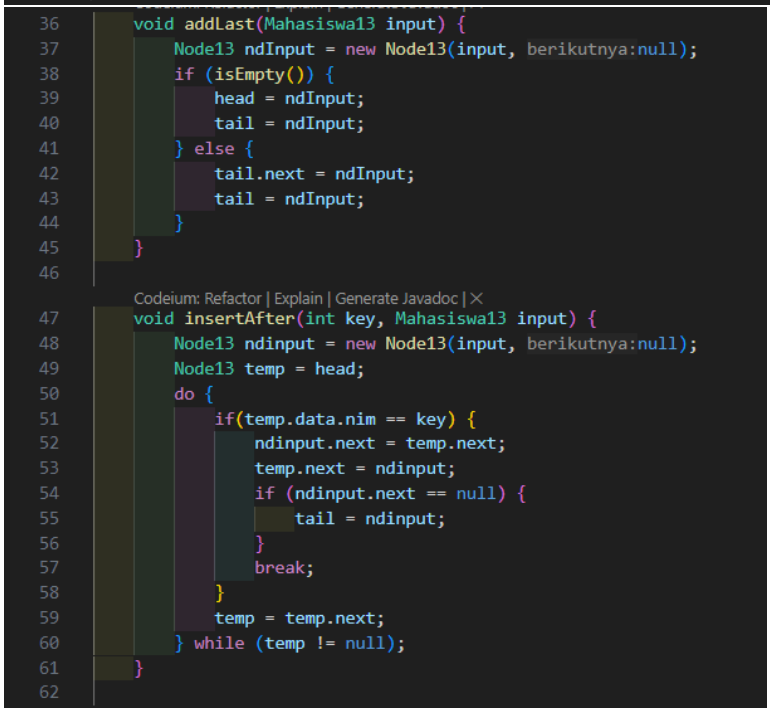
Class Node :

```
src > P11 > Tugas_P11 > J Node13.java > Node13
1   package P11.Tugas_P11;
2
    Codeium: Refactor | Explain
3   public class Node13 {
4       Mahasiswa13 data;
5       Node13 next;
6
7       public Node13(Mahasiswa13 nilai, Node13 berikutnya) {
8           data = nilai;
9           next = berikutnya;
10      }
11  }
12
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Linked List

Class Single Linked List :

```
src > P11 > Tugas_P11 > J SLLMhs13.java > SLLMhs13
1   package P11.Tugas_P11;
2
    Codeium: Refactor | Explain
3   public class SLLMhs13 {
4       Node13 head, tail;
5
        Codeium: Refactor | Explain | Generate Javadoc | ×
6       boolean isEmpty() {
7           return head == null && tail == null;
8       }
9
        Codeium: Refactor | Explain | Generate Javadoc | ×
10      void print() {
11          if (!isEmpty()) {
12              Node13 tmp = head;
13              System.out.println(x:"Isi Linked List:  ");
14              while (tmp != null) {
15                  System.out.println("NIM : " + tmp.data.nim);
16                  System.out.println("Nama : " + tmp.data.nama);
17                  tmp = tmp.next;
18              }
19              System.out.println(x:"");
20          } else {
21              System.out.println(x:"Linked List kosong");
22          }
23      }
24
        Codeium: Refactor | Explain | Generate Javadoc | ×
25      void addFirst(Mahasiswa13 input) {
26          Node13 ndInput = new Node13(input, berikutnya:null);
27          if (isEmpty()) {
28              head = ndInput;
29              tail = ndInput;
30          } else {
31              ndInput.next = head;
32              head = ndInput;
33          }
34      }
35
```



```
36      void addLast(Mahasiswa13 input) {
37          Node13 ndInput = new Node13(input, berikutnya:null);
38          if (isEmpty()) {
39              head = ndInput;
40              tail = ndInput;
41          } else {
42              tail.next = ndInput;
43              tail = ndInput;
44          }
45      }
46
        Codeium: Refactor | Explain | Generate Javadoc | ×
47      void insertAfter(int key, Mahasiswa13 input) {
48          Node13 ndinput = new Node13(input, berikutnya:null);
49          Node13 temp = head;
50          do {
51              if(temp.data.nim == key) {
52                  ndinput.next = temp.next;
53                  temp.next = ndinput;
54                  if (ndinput.next == null) {
55                      tail = ndinput;
56                  }
57                  break;
58              }
59              temp = temp.next;
60          } while (temp != null);
61      }
62
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Linked List

```
63      void insertAt(int index, Mahasiswa13 input) {
64          Node13 ndInput = new Node13(input, berikutnya:null);
65          if (index < 0) {
66              System.out.println("perbaiki logikanya!"
67                      + "kalau indeksnya -1 bagaimana???");
68          } else if (index == 0) {
69              addFirst(input);
70          } else {
71              Node13 temp = head;
72              for (int i = 0; i < index - 1; i++) {
73                  temp = temp.next;
74              }
75              temp.next = new Node13(input, temp.next);
76              if (temp.next.next == null) {
77                  tail = temp.next;
78              }
79          }
80      }
81  }
82
```

Class Main :

```
src > P11 > Tugas_P11 > J SLLMhsMain13.java > ...
1   package P11.Tugas_P11;
2
    Codeium: Refactor | Explain
3   public class SLLMhsMain13 {
        Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | X
4       public static void main(String[] args) {
5           SLLMhs13 singLL = new SLLMhs13();
6
7           Mahasiswa13 Mhs1 = new Mahasiswa13(nim:111, nama:"Anton");
8           Mahasiswa13 Mhs2 = new Mahasiswa13(nim:112, nama:"Prita");
9           Mahasiswa13 Mhs3 = new Mahasiswa13(nim:113, nama:"Yusuf");
10          Mahasiswa13 Mhs4 = new Mahasiswa13(nim:114, nama:"Doni");
11          Mahasiswa13 Mhs5 = new Mahasiswa13(nim:115, nama:"Sari");
12
13          singLL.print();
14          singLL.addFirst(Mhs2);
15          singLL.print();
16          singLL.addLast(Mhs4);
17          singLL.print();
18          singLL.addFirst(Mhs1);
19          singLL.print();
20          singLL.insertAfter(key:114, Mhs5);
21          singLL.print();
22          singLL.insertAt(index:2, Mhs3);
23          singLL.print();
24      }
25  }
26
```

Output :

```
Linked List kosong
Isi Linked List:
NIM : 112
Nama : Prita

Isi Linked List:
NIM : 112
Nama : Prita
NIM : 114
Nama : Doni

Isi Linked List:
NIM : 111
Nama : Anton
NIM : 112
Nama : Prita
NIM : 114
Nama : Doni
```

```
Isi Linked List:
NIM : 111
Nama : Anton
NIM : 112
Nama : Prita
NIM : 114
Nama : Doni
NIM : 115
Nama : Sari

Isi Linked List:
NIM : 111
Nama : Anton
NIM : 112
Nama : Prita
NIM : 113
Nama : Yusuf
NIM : 114
Nama : Doni
NIM : 115
Nama : Sari

PS E:\KULIAH 2\Pratikum Al
```

NAMA : Gilang Purnomo
NIM : 2341720042
NO ABSEN : 13
KELAS : 1F
MATERI : Linked List

2. Buatlah implementasi program antrian layanan unit kemahasiswaan sesuai dengan kondisi yang ditunjukkan pada soal nomor 1! Ketentuan

a. Implementasi antrian menggunakan Queue berbasis Linked List!

b. Program merupakan proyek baru, bukan modifikasi dari soal nomor 1

Jawab :

Class MhsQueue :

src > P11 > Tugas_P11 > LayananUnitKemahasiswaan > MhsQueue13.java > ...

```java
package P11.Tugas_P11.LayananUnitKemahasiswaan;

Codeium: Refactor | Explain
public class MhsQueue13 {
    int nim;
    String nama;

    public MhsQueue13(int nim, String nama) {
        this.nim = nim;
        this.nama = nama;
    }
}
```

Class NodeQueue :

src > P11 > Tugas_P11 > LayananUnitKemahasiswaan > NodeQueue13.java > ...

```java
package P11.Tugas_P11.LayananUnitKemahasiswaan;

Codeium: Refactor | Explain
public class NodeQueue13 {
    MhsQueue13 dataMhs;
    NodeQueue13 next;

    public NodeQueue13(MhsQueue13 nilai, NodeQueue13 berikutnya) {
        dataMhs = nilai;
        next = berikutnya;
    }
}
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Linked List

Class SLLQueue :

```
src > P11 > Tugas_P11 > LayananUnitKemahasiswaan > SLLMhsQueue13.java > SLLMhsQueue13 > print()
1  package P11.Tugas_P11.LayananUnitKemahasiswaan;
2
   Codeium: Refactor | Explain
3  public class SLLMhsQueue13 {
4      NodeQueue13 front, rear;
5
       Codeium: Refactor | Explain | Generate Javadoc | ×
6      boolean isEmpty() {
7          return front == null && rear == null;
8      }
9
       Codeium: Refactor | Explain | Generate Javadoc | ×
10     void print() {
11         if (!isEmpty()) {
12             NodeQueue13 tmp = front;
13             System.out.println(x:"Data Antrian:   ");
14             while (tmp != null) {
15                 System.out.println("NIM : " + tmp.dataMhs.nim);
16                 System.out.println("Nama : " + tmp.dataMhs.nama);
17                 tmp = tmp.next;
18             }
19             System.out.println(x:"");
20         } else {
21             System.out.println(x:"Antrian masih kosong");
22         }
23     }
24
       Codeium: Refactor | Explain | Generate Javadoc | ×
25     void Enqueue(MhsQueue13 input) {
26         NodeQueue13 ndInput = new NodeQueue13(input, berikutnya:null);
27         if (isEmpty()) {
28             front = ndInput;
29             rear = ndInput;
30         } else {
31             rear.next = ndInput;
32             rear = ndInput;
33         }
34     }
35
```

```
36     void Dequeue() {
37         if (isEmpty()) {
38             System.out.println(x:"Data masih kosong");
39             return;
40         } else if (front == rear) {
41             System.out.println("Data yang keluar = NIM : " + front.dataMhs.nim + " | "
42                                + "| Nama : " + front.dataMhs.nama);
43             front = rear = null;
44         } else {
45             MhsQueue13 tmp = front.dataMhs;
46             front = front.next;
47             System.out.println("Data yang keluar = NIM : " + tmp.nim + " | "
48                                + "Nama : " + tmp.nama);
49         }
50     }
51 }
52
```

NAMA	: Gilang Purnomo
NIM	: 2341720042
NO ABSEN	: 13
KELAS	: 1F
MATERI	: Linked List

Class Main :

src > P11 > Tugas_P11 > LayananUnitKemahasiswaan > SLLMhsQueueMain13.java > ...

```java
package P11.Tugas_P11.LayananUnitKemahasiswaan;

Codeium: Refactor | Explain
public class SLLMhsQueueMain13 {
    Run | Debug | Codeium: Refactor | Explain | Generate Javadoc | ×
    public static void main(String[] args) {
        SLLMhsQueue13 antrian = new SLLMhsQueue13();

        MhsQueue13 Mhs1 = new MhsQueue13(nim:111, nama:"Anton");
        MhsQueue13 Mhs2 = new MhsQueue13(nim:112, nama:"Prita");
        MhsQueue13 Mhs3 = new MhsQueue13(nim:113, nama:"Yusuf");
        MhsQueue13 Mhs4 = new MhsQueue13(nim:114, nama:"Doni");
        MhsQueue13 Mhs5 = new MhsQueue13(nim:115, nama:"Sari");

        antrian.print();
        antrian.Enqueue(Mhs2);
        antrian.print();
        antrian.Enqueue(Mhs4);
        antrian.print();
        antrian.Dequeue();
        antrian.Enqueue(Mhs1);
        antrian.print();
        antrian.Enqueue(Mhs5);
        antrian.print();
        antrian.Enqueue(Mhs3);
        antrian.print();
        antrian.Dequeue();
        antrian.Dequeue();
        antrian.print();
    }
}
```

Output :

```
Antrian masih kosong
Data Antrian:
NIM : 112
Nama : Prita

Data Antrian:
NIM : 112
Nama : Prita
NIM : 114
Nama : Doni

Data yang keluar = NIM : 112 | Nama : Prita
Data Antrian:
NIM : 114
Nama : Doni
NIM : 111
Nama : Anton

Data Antrian:
NIM : 114
Nama : Doni
NIM : 111
Nama : Anton
NIM : 115
Nama : Sari

Data Antrian:
NIM : 114
Nama : Doni
NIM : 111
Nama : Anton
NIM : 115
Nama : Sari
NIM : 113
Nama : Yusuf

Data yang keluar = NIM : 114 | Nama : Doni
Data yang keluar = NIM : 111 | Nama : Anton
Data Antrian:
NIM : 115
Nama : Sari
NIM : 113
Nama : Yusuf

PS E:\KULIAH 2\Pratikum Algoritma dan Struktur Data\PrakASD_1F_13>
```