

# **LAPORAN PRAKTIKUM**

**UAS**

**MATA KULIAH STRUKTUR  
DATA**



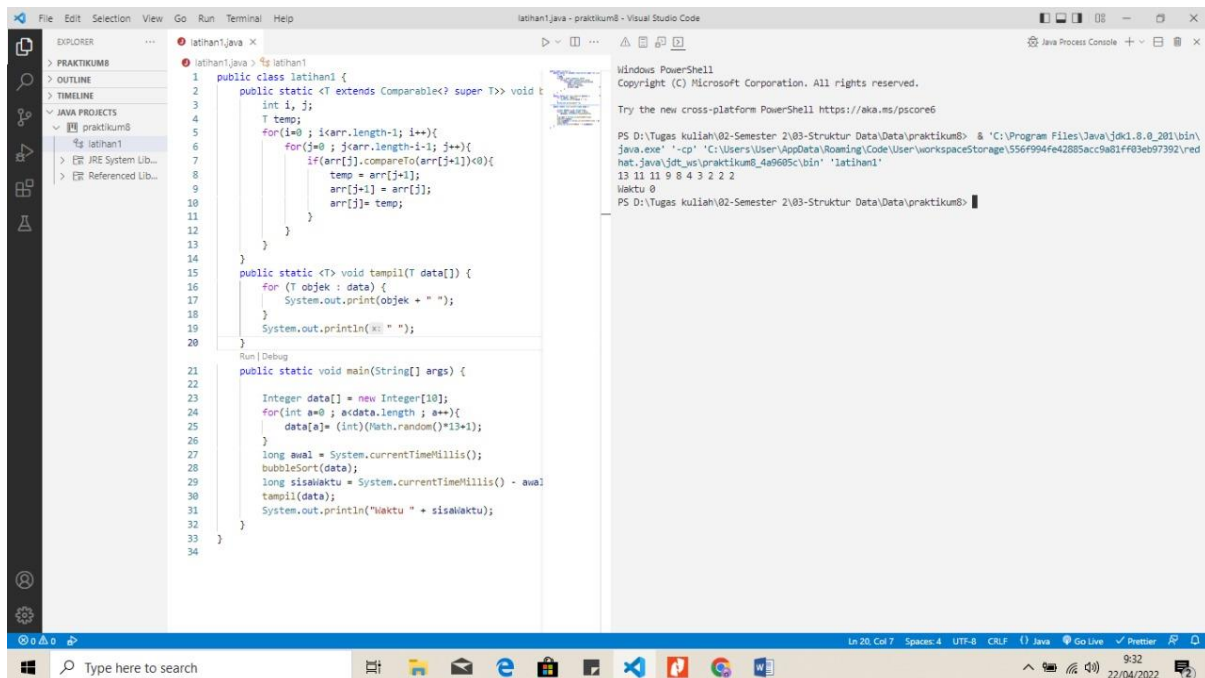
**Selamat Maulana**

**362155401129**

**PROGRAM STUDI DIPLOMA III TEKNIK INFORMATIKA  
POLITEKNIK NEGERI BANYUWANGI 2022**

## 1. Latihan Program Buble shell

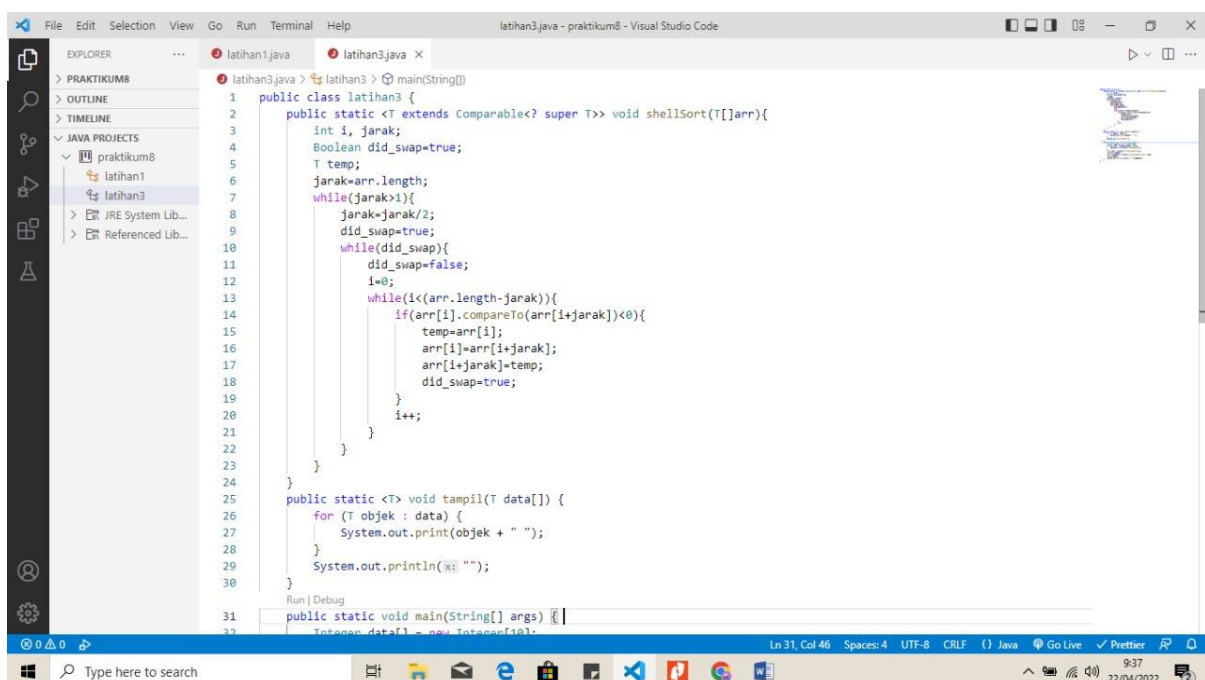
A. Dari percobaan 1 tambahkan method untuk melakukan pengurutan bubble sort secara descending.



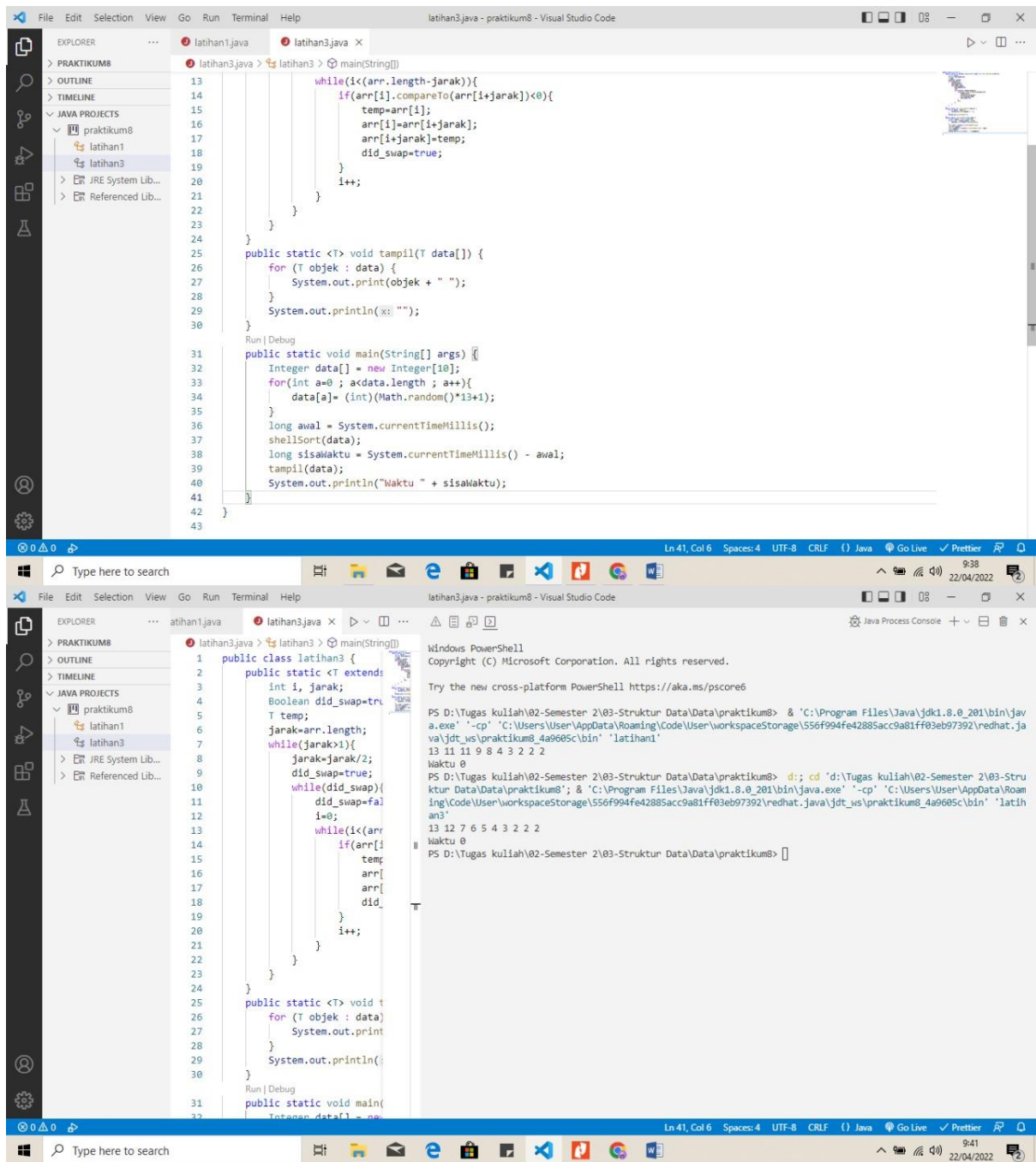
```
1 public class latihan1 {
2     public static <T extends Comparable<? super T>> void bubbleSort(T[] arr) {
3         int i, j;
4         T temp;
5         for(i=0; i<arr.length-1; i++){
6             for(j=0; j<arr.length-1-i; j++){
7                 if(arr[j].compareTo(arr[j+1])<0){
8                     temp = arr[j+1];
9                     arr[j+1] = arr[j];
10                    arr[j] = temp;
11                }
12            }
13        }
14    }
15
16    public static <T> void tampil(T data[]) {
17        for (T objek : data) {
18            System.out.print(objek + " ");
19        }
20        System.out.println("\n");
21    }
22
23    public static void main(String[] args) {
24        Integer data[] = new Integer[10];
25        for(int a=0; a<data.length; a++){
26            data[a] = (int)(Math.random()*13+1);
27        }
28        long awal = System.currentTimeMillis();
29        bubbleSort(data);
30        long sisalaktu = System.currentTimeMillis() - awal;
31        tampil(data);
32        System.out.println("Waktu : " + sisalaktu);
33    }
34 }
```

• Analisa : dari percobaan diatas dapat dianalisa bahwa bubblesort adalah mengurutkan data dengan data selanjutnya maka untuk mengurutkan perlu menggunakan 2for atau 2 perulangan dan diberi inisiasi jika array saat ini (arr[j]) dibandingkan dengan (compareTo) array saat ini ditambah1(arr[j+1]) dikurang 1 lebih besar dai nol maka ditukar dengan cara (temp = arr[j+1]; arr[j+1] = arr[j]; arr[j]= temp;), fungsi dari System.currentTimeMillis() adalah untuk meberikan waktu

B. Dari percobaan 3 tambahkan method untuk melakukan pengurutan shell sort secara descending.



```
1 public class latihan3 {
2     public static <T extends Comparable<? super T>> void shellSort(T[] arr){
3         int i, jarak;
4         Boolean did_swap=true;
5         T temp;
6         jarak=arr.length;
7         while(jarak>1){
8             jarak=jarak/2;
9             did_swap=true;
10            while(did_swap){
11                did_swap=false;
12                i=0;
13                while(i<(arr.length-jarak)){
14                    if(arr[i].compareTo(arr[i+jarak])<0){
15                        temp=arr[i];
16                        arr[i]=arr[i+jarak];
17                        arr[i+jarak]=temp;
18                        did_swap=true;
19                    }
20                    i++;
21                }
22            }
23        }
24    }
25
26    public static <T> void tampil(T data[]) {
27        for (T objek : data) {
28            System.out.print(objek + " ");
29        }
30        System.out.println("\n");
31    }
32
33    public static void main(String[] args) {
34        Integer data[] = new Integer[10];
35        for(int a=0; a<data.length; a++){
36            data[a] = (int)(Math.random()*13+1);
37        }
38        long awal = System.currentTimeMillis();
39        shellSort(data);
40        long sisalaktu = System.currentTimeMillis() - awal;
41        tampil(data);
42        System.out.println("Waktu : " + sisalaktu);
43    }
44 }
```



• Analisa : dari percobaan diatas dapat dianalisa bahwa metode shell sort “jarak ditentukan dengan  $N/2$  dimana  $N$  adalah jumlah array”. Dalam kasus di program ini dibuat yang menjadi  $N$  adalah jarak lalu jarak dibagi 2 sampai setelah itu diurutkan dengan Hasil compile jarak tersebut begitu seterusnya sampai jarak sama dengan 1, jika jarak sudah bernilai satu maka sama dengan pengurutan bubble sort. Di dalam kodingan yang kami buat ada 10 data array acak lalu data tersebut dibagi 2 mejadi jaraknya 5, data dengan jarak 5 akan saling diurutkan, setelah diurutkan dibagi 2 lagi lalu diurutkan lagi. Begitu seterusnya sampai jarak = 1. Jika sudah satu akan diurutkan dengan sebelumnya

## 2. Latihan Program Quick dan Marge

A. Dari percobaan 1 tambahkan method untuk melakukan pengurutan quick sort secara descending.

The image displays two screenshots of a Visual Studio Code editor window, showing Java code for a QuickSort algorithm. The editor is titled "Quickjava - praktikum9 - Visual Studio Code".

**Top Screenshot:** The code editor shows the following code:

```
1 public class Quick {
2     public static <T extends Comparable <?super T>> int Partition(T[]arr,int p,int r){
3         int i,j;
4         T pivot, temp;
5         pivot=arr[p];
6         i=p;
7         j=r;
8         while (i<=j){
9             while (pivot.compareTo(arr[j])<0)
10                j--;
11             while (pivot.compareTo(arr[i])<0)
12                i++;
13             if(i<j){
14                 temp=arr[i];
15                 arr[i]=arr[j];
16                 arr[j]=temp;
17                 j--;
18                 i++;
19             }
20             else
21                 return j;
22         }
23         return j;
24     }
25     private static <T extends Comparable <?super T>> void Quicksort(T[]arr,int p,int r){
26         int q;
27         if(p<r){
28             q=Partition(arr, p, r);
29             Quicksort(arr, q+1, r);
30         }
31     }
32     public static <T> void tampilkan(T data[]) {
33         for (T objek : data){
```

**Bottom Screenshot:** The code editor shows the following code:

```
23         return j;
24     }
25     private static <T extends Comparable <?super T>> void Quicksort(T[]arr,int p,int r){
26         int q;
27         if(p<r){
28             q=Partition(arr, p, r);
29             Quicksort(arr, q+1, r);
30         }
31     }
32     public static <T> void tampilkan(T data[]) {
33         for (T objek : data){
34             System.out.print(objek + " ");
35         }
36         System.out.println(" ");
37     }
38
39     public static void main(String[] args) {
40         Integer data[] = new Integer[10];
41         for (int a=0 ; a<data.length ; a++)
42         {
43             data[a]= (int)(Math.random()*13+1);
44         }
45         System.out.println("Sebelum Quick Sort :");
46         tampilkan(data);
47         long awal = System.currentTimeMillis();
48         Quicksort(data, 0, data.length-1);
49         long sisaWaktu = System.currentTimeMillis() -awal;
50         System.out.println("Setelah Quick Sort Descending :");
51         tampilkan(data);
52         System.out.println("Waktu " + sisaWaktu);
53     }
54 }
```

```

12  i++;
13  if(i < j){
14      temp = arr[i];
15      arr[i] = arr[j];
16      arr[j] = temp;
17      j--;
18      i++;
19  }
20  else
21      return j;
22  }
23  return j;
24  }
25  private static <T extends Comparable<T>> void quickSort(T[] arr) {
26      int q;
27      if(arr.length > 1){
28          q = partition(arr, 0, arr.length - 1);
29          quickSort(arr, 0, q - 1);
30          quickSort(arr, q + 1, arr.length - 1);
31      }
32  }
33  public static <T> void main(String[] args) {
34      for (T objek : data)
35          System.out.print(objek + " ");
36      System.out.println();
37  }
38  }
39  public static void main(String[] args) {
40      Integer data[] = new Integer[10];
41      for (int a = 0; a < data.length; a++)
42          data[a] = (int) (Math.random() * 100);
43  }

```

Windows PowerShell  
Copyright (C) Microsoft Corporation. All rights reserved.  
Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS D:\Tugas kuliah\02-Semester 2\03-Struktur Data\Data\praktikum9> & 'C:\Program Files\Java\jdk1.8.0\_201\bin\java.exe' -cp 'C:\Users\User\AppData\Roaming\Code\User\workspaceStorage\9afa641dfbdc1bea36989faf4995c8\redhat.java\jdt\_ws\praktikum9\_4a9e05d\bin' Quick  
Sebelum Quick Sort :  
11 11 11 8 3 12 8 2 6 4  
Setelah Quick Sort Descending :  
4 6 2 8 3 11 11 11 8 12  
Waktu 0  
PS D:\Tugas kuliah\02-Semester 2\03-Struktur Data\Data\praktikum9>

- Analisa Pemrograman diatas adalah program untuk penggunaan Quick Sort secara descending yaitu keterbalikan dari ascending, dimana program akan mengurutkan data/angka dari kanan ke kiri/dari terkecil sampai terbesar

**B. Buatlah class Mahasiswa dengan variable nrp dan nama yang memiliki tipe String! Class Mahasiswa mengimplementasikan interface Comparable, selanjutnya implementasikan fungsi abstract compareTo(), untuk membandingkan dua objek mahasiswa berdasarkan nrp.**

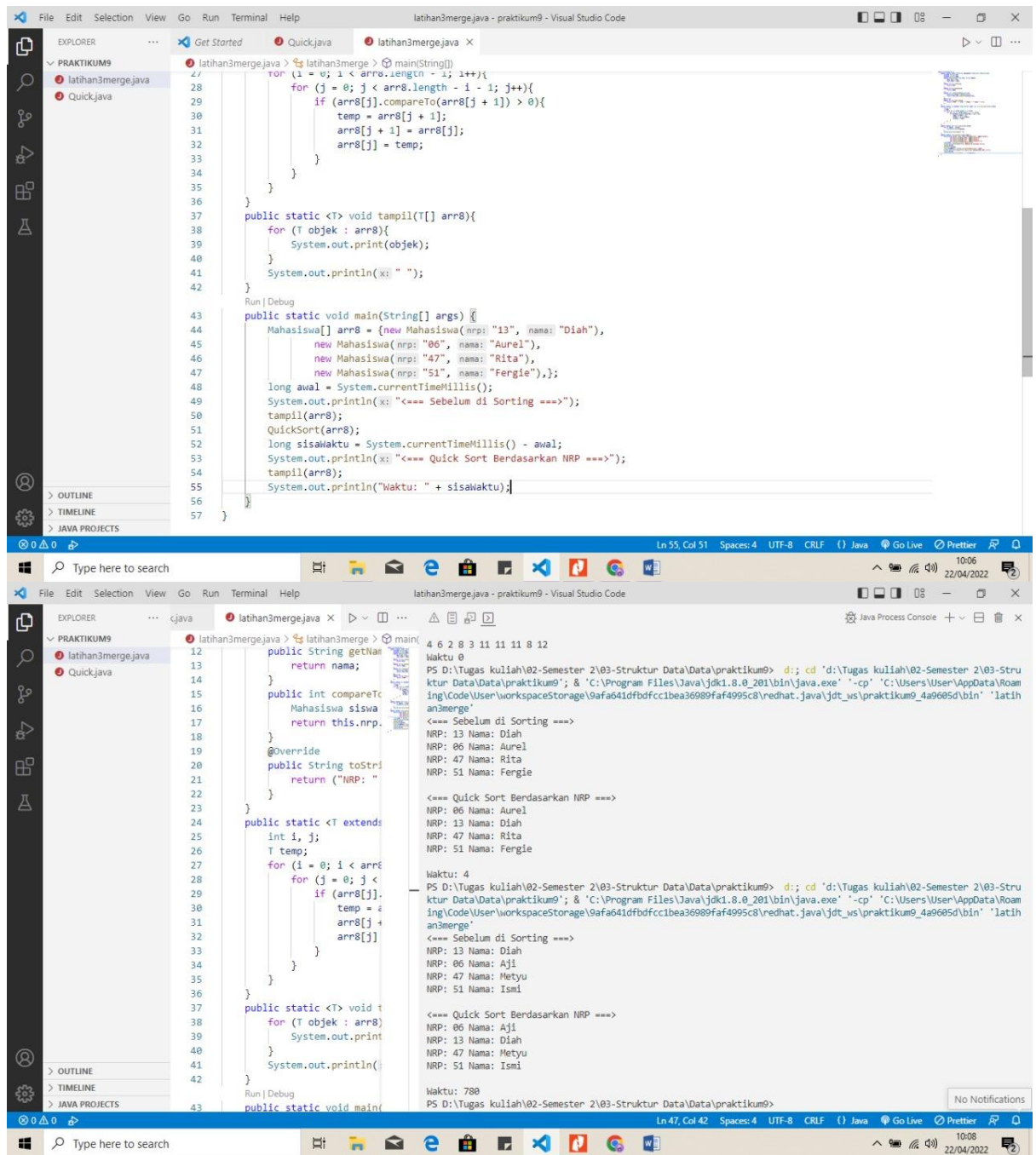
```

1  class latihan3merge {
2      public static class Mahasiswa implements Comparable<Mahasiswa>{
3          private String nrp;
4          private String nama;
5          public Mahasiswa(String nrp, String nama){
6              this.nrp = nrp;
7              this.nama = nama;
8          }
9          public String getNrp(){
10             return nrp;
11         }
12         public String getNama(){
13             return nama;
14         }
15         public int compareTo(Mahasiswa o){
16             Mahasiswa siswa = (Mahasiswa) o;
17             return this.nrp.compareTo(siswa.nrp);
18         }
19         @Override
20         public String toString(){
21             return ("NRP: " + nrp + "\tNama: " + nama + "\n");
22         }
23     }
24     public static <T extends Comparable<T>> void QuickSort(T[] arr8){
25         int i, j;
26         T temp;
27         for (i = 0; i < arr8.length - 1; i++){
28             for (j = 0; j < arr8.length - i - 1; j++){
29                 if (arr8[j].compareTo(arr8[j + 1]) > 0){
30                     temp = arr8[j + 1];
31                     arr8[j + 1] = arr8[j];
32                     arr8[j] = temp;
33                 }
34             }
35         }
36     }
37 }

```

Ln 29, Col 57 Spaces: 4 UTF-8 CRLF {} Java Go Live Prettier





```
27 for (i = 0; i < arr8.length - 1; i++){
28     for (j = 0; j < arr8.length - i - 1; j++){
29         if (arr8[j].compareTo(arr8[j + 1]) > 0){
30             temp = arr8[j + 1];
31             arr8[j + 1] = arr8[j];
32             arr8[j] = temp;
33         }
34     }
35 }
36
37 public static <T> void tampil(T[] arr8){
38     for (T objek : arr8){
39         System.out.print(objek);
40     }
41     System.out.println("\n");
42 }
43
44 public static void main(String[] args) {
45     Mahasiswa[] arr8 = {new Mahasiswa(nrp: "13", nama: "Diah"),
46                         new Mahasiswa(nrp: "06", nama: "Aji"),
47                         new Mahasiswa(nrp: "47", nama: "Metu"),
48                         new Mahasiswa(nrp: "51", nama: "Ismi")};
49     long awal = System.currentTimeMillis();
50     System.out.println("\n<=== Sebelum di Sorting ===>");
51     tampil(arr8);
52     QuickSort(arr8);
53     long sisaWaktu = System.currentTimeMillis() - awal;
54     System.out.println("\n<=== Quick Sort Berdasarkan NRP ===>");
55     tampil(arr8);
56     System.out.println("Waktu: " + sisaWaktu);
57 }
```

- Analisa Pemrograman diatas adalah program untuk melakukan pengurutan NRP dari 4 nama mahasiswa secara ascending/dari angka yang terkecil sampai terbesar, menggunakan Quick Sort.