

JOBSHEET VI SEARCHING

6.1. Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Menjelaskan mengenai algoritma Searching.
2. Membuat dan mendeklarasikan struktur algoritma Searching.
3. Menerapkan dan mengimplementasikan algoritma Searching.

6.2. Searching / Pencarian Menggunakan Algoritma Sequential Search

6.2.1 Sequential Search Menggunakan Array

1. Buat folder baru dengan nama Praktikum06. Buat file dengan nama Sorting.java
2. Tambahkan method `sequentialSearch()` yang melakukan pencarian data bertipe integer di dalam array of integer

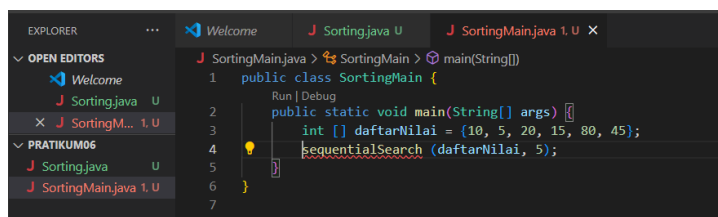
```
public static void sequentialSearch(int[] arr, int key) {  
    for (int i = 0; i < arr.length; i++) {  
        if (i == key) {  
            System.out.println("Data ditemukan pada indeks ke-" + i);  
        }  
    }  
  
    System.out.println("Data tidak ditemukan");  
}
```

3. Tambahkan fungsi main sebagai berikut

```
Run | Debug  
public static void main(String[] args) {  
    int[] daftarNilai = { 10, 5, 20, 15, 80, 45 };  
    sequentialSearch(daftarNilai, 5);  
}
```

4. Compile dan run program

- Terjadi error pada bagian `sequentialSearch` pada class Main, dan terdapat kekurangan pada kode program sehingga hasil output tidak sesuai dengan hasil.



Maka, diperbaiki dengan menambah **Sorting.sequentialSearch**, menambah **return**, dan perubahan dari **if (i == key)** menjadi **if (arr[i] == key)**.

Class Sorting

```
J Sorting.java > Sorting > sequentialSearch(int[], int)
3  /*
4  public class Sorting {
5
6      public static void sequentialSearch (int[] arr, int key){
7          for (int i = 0; i < arr.length; i++) {
8              if (arr[i] == key) {
9                  System.out.println("Data ditemukan pada indeks ke-" + i);
10                 return;
11             }
12         }
13         System.out.println(x:"Data tidak ditemukan");
14     }
15 }
```

Class Main

```
J SortingMain.java > SortingMain
1  public class SortingMain {
2      Run | Debug
3      public static void main(String[] args) {
4          int [] daftarNilai = {10, 5, 20, 15, 80, 45};
5          Sorting.sequentialSearch (daftarNilai, key:5);
6      }
7  }
```

Hasil Output :

```
PS C:\Users\TOSHIBA\Semester 2\Pratikum06> c::; cd 'c:\Us
s\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExcept
ode\User\workspaceStorage\c0d72cb89ab7151342d47d32f2b5b7d
in'
Data ditemukan pada indeks ke-1
PS C:\Users\TOSHIBA\Semester 2\Pratikum06>
```

6.2.2 Sequential Search Menggunakan Array of Object

Perhatikan diagram class Mahasiswa di bawah ini! Diagram class ini yang selanjutnya akan dibuat sebagai acuan dalam membuat kode program class Mahasiswa.

Mahasiswa
Nim: int nama: String umur: int ipk: double
Mahasiswa(ni:int, n: String, u: int, i: double) tampil(): void

Berdasarkan class diagram di atas, akan dibuat class Mahasiswa yang berfungsi untuk membuat objek mahasiswa yang akan dimasukkan ke dalam sebuah array. Terdapat sebuah konstruktor berparameter dan juga fungsi tampil() untuk menampilkan semua attribute yang ada.

PencarianMhs
listMhs: Mahasiswa[5] idx: int
tambah(mhs: Mahasiswa): void tampil(): void FindSeqSearch(int cari): int Tampilposisi(int x,int pos): void TampilData(int x,int pos) :void

Selanjutnya class diagram di atas merupakan representasi dari sebuah class yang berfungsi untuk melakukan operasi-operasi dari objek array mahasiswa, misalkan untuk menambahkan objek mahasiswa, menampilkan semua data mahasiswa, untuk melakukan pencarian berdasarkan NIM menggunakan algoritma Sequential Search, menampilkan posisi dari data yang dicari, serta menampilkan data mahasiswa yang dicari.

❖ Langkah-langkah Percobaan Sequential Search

1. Buatlah Project baru pada Netbeans dengan nama **TestSearching**
2. Kemudian buat packages baru dengan nama **minggu7**.
3. Buat class **Mahasiswa**, kemudian deklarasikan atribut berikut ini:

```
public class Mahasiswa {
    int nim;
    String nama;
    int umur;
    double ipk;
```

4. Buatlah konstruktor dengan nama **Mahasiswa** dengan parameter (**int ni**, **String n**, **int u**, **double i**) kemudian Isi konstruktor tersebut dengan kode berikut!

```
Mahasiswa(int ni, String n, int u, double i){
    nim = ni;
    nama = n;
    umur = u;
    ipk = i;
}
```

5. Buatlah method **tampil** bertipe void.

```
void tampil(){
    System.out.println("Nim = " + nim);
    System.out.println("Nama = " + nama);
    System.out.println("Umur = " + umur);
    System.out.println("IPK = " + ipk);
}
```

6. Buat class baru dengan nama **PencarianMhs** seperti di bawah ini!

```
public class PencarianMhs {
    Mahasiswa listMhs[] = new Mahasiswa[5];
    int idx;
}
```

7. Tambahkan method **tambah()** di dalam class tersebut! Method **tambah()** digunakan untuk menambahkan objek dari class Mahasiswa ke dalam atribut listMhs.

```
void tambah(Mahasiswa m) {
    if(idx < listMhs.length){
        listMhs[idx] = m;
        idx++;
    } else {
        System.out.println("Data sudah penuh !!");
    }
}
```

8. Tambahkan method **tampil()** di dalam class **PencarianMhs**! Method **tampil()** digunakan untuk menampilkan semua data mahasiswa-mahasiswa yang ada di dalam class tersebut!

Perhatikan penggunaan sintaks for yang agak berbeda dengan for yang telah dipelajari sebelumnya, meskipun secara konsep sebenarnya mirip.

```
void tampil() {
    for (Mahasiswa m : listMHs) {
        m.tampil();
        System.out.println("-----");
    }
}
```

9. Tambahkan method **FindSeqSearch** bertipe integer dengan parameter **cari** bertipe integer. Kemudian Deklarasikan isi method **FindSeqSearch** dengan algoritma pencarian data menggunakan teknik sequential searching.

```
public int FindSeqSearch(int cari) {
    int posisi = -1;
    for (int j = 0; j < listMHs.length; j++) {
        if (listMHs[j].nim==cari) {
            posisi = j;
            break;
        }
    }
    return posisi;
}
```

10. Buatlah method **Tampilpoisisi** bertipe void dan Deklarasikan isi dari method **Tampilpoisisi**.

```
public void Tampilpoisisi(int x,int pos)
{
    if (pos!= -1) {
        System.out.println("data : " + x + "ditemukan pada indeks " + pos);
    } else {
        System.out.println("data " + x + "tidak ditemukan");
    }
}
```

11. Buatlah method **TampilData** bertipe void dan Deklarasikan isi dari method **TampilData**.

```
public void TampilData(int x,int pos)
{
    if (pos!= -1) {
        System.out.println("Nim\t : " + x );
        System.out.println("Nama\t : "+listMHs[pos].nama);
        System.out.println("Umur\t : "+listMHs[pos].umur);
        System.out.println("IPK\t : "+listMHs[pos].ipk);
    } else {
        System.out.println("data " + x + "tidak ditemukan");
    }
}
```

12. Buatlah class baru dengan nama **MahasiswaMain** tambahkan method **main** seperti pada gambar berikut!

```
public class MahasiswaMain {
    public static void main(String[] args) {

    }
}
```

13. Di dalam method **main()** , buatlah sebuah objek PencarianMhs dan buatlah 5 objek mahasiswa kemudian tambahkan semua objek mahasiswa tersebut dengan memanggil fungsi tambah pada objek PencarianMhs.

```

Scanner s = new Scanner(System.in);
Scanner sl = new Scanner(System.in);

PencarianMhs data = new PencarianMhs();
int jumMhs = 5;

System.out.println("-----");
System.out.println("Masukkan data mahasiswa secara Urut dari Nim Terkecil");
for(int i = 0; i < jumMhs; i++){
    System.out.println("-----");
    System.out.print("Nim\t: ");
    int nim = s.nextInt();
    System.out.print("Nama\t : ");
    String nama = sl.nextLine();
    System.out.print("Umur\t : ");
    int umur = s.nextInt();
    System.out.print("IPK\t : ");
    double ipk = s.nextDouble();

    Mahasiswa m = new Mahasiswa(nim, nama, umur, ipk);
    data.tambah(m);
}

```

14. Panggil method **tampil()** untuk melihat semua data yang telah dimasukan.

```

System.out.println("-----");
System.out.println("Data keseluruhan Mahasiswa : ");
data.tampil();

```

15. Untuk melakukan pencarian berdasarkan NIM mahasiswa. Buatlah variable **cari** yang dapat menampung masukan dari keyboard lalu panggil method **FindSeqSearch** dengan isi parameternya adalah variable cari.

```

System.out.println("-----");
System.out.println("-----");
System.out.println("Pencarian Data : ");
System.out.println("Masukkan Nim Mahasiswa yang dicari: ");
System.out.print("NIM : ");
int cari = s.nextInt();
System.out.println("menggunakan sequential Search");
int posisi = data.FindSeqSearch(cari);

```

16. Lakukan pemanggilan method **Tampilposisi** dari class **PencarianMhs**.

```
data.Tampilpoisisi(cari, posisi);
```

17. Lakukan pemanggilan method **TampilData** dari class **PencarianMhs**.

```
data.TampilData(cari, posisi);
```

18. Jalankan dan amati hasilnya.

6.2.2. Verifikasi Hasil Percobaan

Cocokkan hasil kode program anda dengan gambar berikut ini.

Masukkan data mahasiswa secara Urut dari Nim Terkecil :

```
-----
Nim      : 2017
Nama     : Dewi Lestari
Umur     : 23
IPK      : 3.5
-----
```

```
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4
-----
```

```
Nim      : 2019
Nama     : Danang Adi
Umur     : 22
IPK      : 3.7
-----
```

```
Nim      : 2020
Nama     : Budi Prakarsa
Umur     : 20
IPK      : 2.9
-----
```

```
Nim      : 2021
Nama     : Vania Siti
Umur     : 20
IPK      : 3.0
-----
```

Data keseluruhan Mahasiswa :

```
Nim = 2017
Nama = Dewi Lestari
Umur = 23
IPK = 3.5
-----
```

```
Nim = 2018
Nama = Sinta Sanjaya
Umur = 22
IPK = 4.0
-----
```

```
Nim = 2019
Nama = Danang Adi
Umur = 22
IPK = 3.7
-----
```

```
Nim = 2020
Nama = Budi Prakarsa
Umur = 20
IPK = 2.9
-----
```

```
Nim = 2021
Nama = Vania Siti
Umur = 20
IPK = 3.0
-----
```



```
Pencarian Data :
Masukkan Nim Mahasiswa yang dicari:
NIM : 2018
menggunakan sequential Search
data : 2018ditemukan pada indeks 1
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4.0
```

➤ Hasil Input :

Class Mahasiswa

```
minggu7 > J Mahasiswa06.java > Mahasiswa06 > Mahasiswa06(int, String, int, double)
1  /**
2   * Mahasiswa06
3   */
4  package minggu7;
5  public class Mahasiswa06 {
6      int nim;
7      String nama;
8      int umur;
9      double ipk;
10
11      Mahasiswa06 (int ni, String n, int u, double i){
12          nim = ni;
13          nama = n;
14          umur = u;
15          ipk = i;
16      }
17      //Method Tampil
18      void tampil (){
19          System.out.println("Nim = " +nim);
20          System.out.println("Nama = "+nama);
21          System.out.println("Umur = "+umur);
22          System.out.println("IPK = "+ipk);
23      }
24  }
```

Class Pencarian Mahasiswa

```
6  public class PencarianMhs {
7      Mahasiswa06 listMhs[] = new Mahasiswa06 [5];
8      int idx;
9      void tampil(){
10         for(Mahasiswa06 m : listMhs){
11             m.tampil();
12             System.out.println(x:"-----");
13         }
14     }
15     //Method Tambah
16     void Tambah (Mahasiswa06 m) {
17         if (idx < listMhs.length){
18             listMhs[idx] = m;
19             idx++;
20         }
21         else {
22             System.out.println(x:"Data Sudah Penuh! ");
23         }
24     }
25     //Algoritma pencarian data Sequential Searching
26     public int FindSeqSearch (int cari) {
27         int posisi = -1;
28         for (int j = 0; j < listMhs.length; j++) {
29             if (listMhs[j].nim == cari) {
30                 posisi = j;

```

```

30         posisi = j;
31         break;
32     }
33 }
34 return posisi;
35 }
36 public void Tampilposisi (int x, int pos) {
37     if (pos != -1) {
38         System.out.println("Data : " + x + "ditemukan pada indeks " + pos);
39     } else {
40         System.out.println("Data " + x + "tidak ditemukan");
41     }
42 }
43 public void TampilData (int x, int pos){
44     if (pos != -1) {
45         System.out.println("Nim \t : " + x);
46         System.out.println("Nama \t : " +listMhs[pos].nama);
47         System.out.println("Umur \t : " +listMhs[pos].umur);
48         System.out.println("IPK \t : " +listMhs[pos].ipk);
49     } else {
50         System.out.println("Data " + x + "tidal ditemukan");
51     }
52 }
53 }

```

Class Main

```

2  import java.util.Scanner;
3  public class MahasiswaMain {
4      Run | Debug
5      public static void main(String[] args) {
6          Scanner s = new Scanner (System.in);
7          Scanner s1 = new Scanner (System.in);
8
9          PencarianMhs data = new PencarianMhs ();
10         int jumMhs = 5;
11
12         System.out.println(x:"-----");
13         System.out.println(x:"Masukan data Mahasiswa secara urut daru Nim Terkecil ");
14         for (int i = 0; i < jumMhs; i++) {
15             System.out.println(x:"-----");
16             System.out.print(s:"Nim \t : ");
17             int nim = s.nextInt ();
18             System.out.print(s:"Nama \t : ");
19             String nama = s1.nextLine ();
20             System.out.print(s:"Umur \t : ");
21             int umur = s.nextInt ();
22             System.out.print(s:"IPK \t : ");
23             double ipk= s.nextDouble ();
24
25             Mahasiswa06 m = new Mahasiswa06 (nim, nama, umur, ipk);
26             data.Tambah(m);
27         }
28     }
29 }

```

```

27
28     System.out.println(x:"-----");
29     System.out.println(x:"Data keseluruhan Mahasiswa : ");
30     data.tampil();
31
32     System.out.println(x:"-----");
33     System.out.println(x:"");
34     System.out.println(x:"Pencarian Data : ");
35     System.out.println(x:"Masukkan Nim Mahasiswa yang dicari : ");
36     System.out.print(s:"NIM : ");
37     int cari = s.nextInt();
38     System.out.println(x:"Menggunakan sequential Search");
39     int posisi = data.FindSeqSearch(cari);
40     data.Tampilposisi(cari, posisi);
41     data.TampilData (cari, posisi);
42 }
43 }

```

➤ Hasil Output :

```

Masukan data Mahasiswa secara urut dari Nim Terkecil
-----
Nim      : 2017
Nama     : Dewi Lestari
Umur     : 23
IPK      : 3.5
-----
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4
-----
Nim      : 2019
Nama     : Danang Adi
Umur     : 22
IPK      : 3.7
-----
Nim      : 2020
Nama     : Budi Prakarsa
Umur     : 20
IPK      : 2.9
-----
Nim      : 2021
Nama     : Vania Siti
Umur     : 20
IPK      : 2
    
```

```

-----
Data keseluruhan Mahasiswa :
Nim = 2017
Nama = Dewi Lestari
Umur = 23
IPK = 3.5
-----
Nim = 2018
Nama = Sinta Sanjaya
Umur = 22
IPK = 4.0
-----
Nim = 2019
Nama = Danang Adi
Umur = 22
IPK = 3.7
-----
Nim = 2020
Nama = Budi Prakarsa
Umur = 20
IPK = 2.9
-----
Nim = 2021
Nama = Vania Siti
Umur = 20
IPK = 2.0
-----
    
```

```

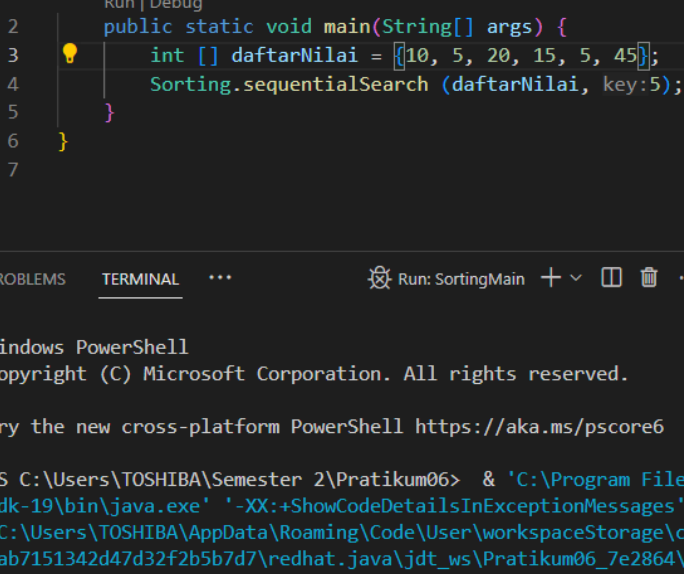
-----
Pencarian Data :
Masukkan Nim Mahasiswa yang dicari :
NIM : 2018
Menggunakan sequential Search
Data : 2018ditemukan pada indeks 1
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4.0
    
```

6.2.3. Pertanyaan

1. Lakukan perubahan array daftarNilai pada fungsi main().

```

Run | Debug
public static void main(String[] args) {
    int[] daftarNilai = { 10, 5, 20, 15, 5, 45 };
    sequentialSearch(daftarNilai, 5);
}
    
```



The screenshot shows an IDE with a Java file named `SortingMain.java`. The code defines a `SortingMain` class with a `main` method. Inside the `main` method, an array `daftarNilai` is initialized with the values `{10, 5, 20, 15, 5, 45}`, and the `Sorting.sequentialSearch` method is called with `daftarNilai` and `key:5`. The IDE interface includes a toolbar with icons for Run, Debug, and other actions. Below the code editor, there is a terminal window titled "Windows PowerShell" showing the command prompt output. The terminal displays the command to run the Java program, the output "Data ditemukan pada indeks ke-1", and the prompt for the next command.

```

J SortingMain.java > SortingMain > main(String[])
1 public class SortingMain {
    Run | Debug
2 public static void main(String[] args) {
3     int [] daftarNilai = {10, 5, 20, 15, 5, 45};
4     Sorting.sequentialSearch (daftarNilai, key:5);
5 }
6 }
7

```

PROBLEMS TERMINAL ... Run: SortingMain + - □ ✕ ... ^ ✕

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

```

PS C:\Users\TOSHIBA\Semester 2\Pratikum06> & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\TOSHIBA\AppData\Roaming\Code\User\workspaceStorage\c0d72cb89ab7151342d47d32f2b5b7d7\redhat.java\jdt_ws\Pratikum06_7e2864\bin' 'SortingMain'
Data ditemukan pada indeks ke-1
PS C:\Users\TOSHIBA\Semester 2\Pratikum06>

```

- Jelaskan perbedaan metod **TampilData** dan **TampilPosisi** pada class PencarianMhs
 - **TampilData** digunakan untuk menampilkan detail data yang ditemukan array sedangkan **TampilPosisi** digunakan untuk menampilkan pesan kepada pengguna dengan hasil pencarian suatu data dalam array
- Jelaskan fungsi **break** pada kode program dibawah ini!

```
if (listMhs[j].nim==cari) {  
    posisi = j;  
    break;  
}
```

 - Untuk menghentikan iterasi loop setelah data yang dicari ditemukan
- Jika Data Nim yang dimasukkan tidak terurut dari kecil ke besar. Apakah program masih dapat berjalan? Apakah hasil yang dikeluarkan benar? Mengapa demikian!
 - Akan tetap berjalan meskipun data NIM tidak terurut dari kecil ke besar. Namun, hasil yang dihasilkan mungkin tidak selalu benar jika data tidak terurut. Karena algoritma pencarian sequential bergantung pada urutan data untuk memeriksa setiap elemen dalam urutan yang diberikan

6.3. Searching / Pencarian Menggunakan Binary Search

6.3.1. Langkah-langkah Percobaan Binary Search menggunakan Array

1. Tambahkan method `binarySearchAsc()` pada file `Sorting.java`

```
public static int binarySearchAsc(int[] arr, int key) {
    int start = 0, end = arr.length - 1;

    while (start <= end) {
        int mid = start + (end - start) / 2;

        if (arr[mid] == key) {
            return mid;
        }

        if (arr[mid] < key) {
            start = mid + 1;
        }
        else {
            end = mid - 1;
        }
    }

    return -1;
}
```

2. Tambahkan baris program untuk menguji method `binarySearchAsc()` pada fungsi `main()`

```
int[] sortedNilai = { 5, 5, 10, 20, 30, 40, 50 };
int index = binarySearchAsc(sortedNilai, 5);

if (index != -1) {
    System.out.println("Data ditemukan pada indeks ke-" + index);
}
else {
    System.out.println("Data tidak ditemukan");
}
```

3. Run dan compile program

Class Sorting

```
J Sorting.java > Sorting > binarySearchAsc(int[], int)
4 public class Sorting {
15
16     public static int binarySearchAsc (int[] arr, int key) {
17         int start = 0, end = arr.length - 1;
18         while (start <= end) {
19             int mid = start + (end - start) / 2;
20             if (arr[mid] == key) {
21                 return mid;
22             }
23             if (arr[mid] < key) {
24                 start = mid + 1;
25             } else {
26                 end = mid - 1;
27             }
28         }
29         return -1;
30     }
31 }
```

Class Main

```
Welcome J SortingMain.java U J SortingMain.java U
J SortingMain.java > SortingMain > main(String[])
1 public class SortingMain {
2     Run | Debug
3     public static void main(String[] args) {
4         //int [] daftarNilai = {10, 5, 20, 15, 5, 45};
5         //Sorting.sequentialSearch (daftarNilai, 5);
6         int [] sortedNilai = {5, 5, 10, 20, 30, 40, 50};
7         int index = Sorting.binarySearchAsc(sortedNilai, key:5);
8
9         if (index != -1) {
10             System.out.println("Data ditemukan pada indeks ke- " +index);
11         } else {
12             System.out.println(x:"Data tidak ditemukan");
13         }
14     }
15 }
16 }
```

Hasil Output

```
PS C:\Users\TOSHIBA\Semester 2\Pratikum06> c::; c
-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C
\redhat.java\jdt_ws\Pratikum06_7e2864\bin' 'Sorti
Data ditemukan pada indeks ke- 1
PS C:\Users\TOSHIBA\Semester 2\Pratikum06>
```

6.3.2. Langkah-langkah Percobaan Binary Search menggunakan Array of Object

1. Pada percobaan 6.2.2 (sequential search) tambahkan method **FindBinarySearch** bertipe integer pada class **PencarianMhs**. Kemudian Deklarasikan isi method **FindBinarySearch** dengan algoritma pencarian data menggunakan teknik binary searching.

```
public int FindBinarySearch(int cari, int left, int right) {
    int mid;
    if (right >= left) {
        mid = (left + right) / 2;
        if (cari == listMHs[mid].nim) {
            return (mid);
        } else if (listMHs[mid].nim > cari) {
            return FindBinarySearch(cari, left, mid - 1);
        } else {
            return FindBinarySearch(cari, mid + 1, right);
        }
    }
    return -1;
}
```

2. Panggil method **FindBinarySearch** terdapat pada class **PencarianMhs** di kelas **Mahasiswamain**. Kemudia panggil method **tampilposisi** dan **tampilData**

```
System.out.println("=====");
System.out.println("menggunakan binary Search");
posisi = data.FindBinarySearch(cari, 0, jumMhs - 1);
data.Tampilpoisisi(cari, posisi);
data.TampilData(cari, posisi);
```

3. Jalankan dan amati hasilnya.

6.3.2. Verifikasi Hasil Percobaan

Cocokkan hasil kode program anda dengan gambar berikut ini.

Masukkan data mahasiswa secara Urut dari Nim Terkecil :

 Nim : 2017
 Nama : Dewi Lestari
 Umur : 23
 IPK : 3.5

Nim : 2018
 Nama : Sinta Sanjaya
 Umur : 22
 IPK : 4

Nim : 2019
 Nama : Danang Adi
 Umur : 22
 IPK : 3.7

Nim : 2020
 Nama : Budi Prakarsa
 Umur : 20
 IPK : 2.9

Nim : 2021
 Nama : Vania Siti
 Umur : 20
 IPK : 3.0

Data keseluruhan Mahasiswa :

Nim = 2017
 Nama = Dewi Lestari
 Umur = 23
 IPK = 3.5

Nim = 2018
 Nama = Sinta Sanjaya
 Umur = 22
 IPK = 4.0

Nim = 2019
 Nama = Danang Adi
 Umur = 22
 IPK = 3.7

Nim = 2020
 Nama = Budi Prakarsa
 Umur = 20
 IPK = 2.9

Nim = 2021
 Nama = Vania Siti
 Umur = 20
 IPK = 3.0

```

Pencarian Data :
Masukkan Nim Mahasiswa yang dicari:
NIM : 2018
menggunakan sequential Search
data : 2018ditemukan pada indeks 1
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4.0
=====
menggunakan binary Search
data : 2018ditemukan pada indeks 1
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4.0
    
```

➤ Hasil Input :

Tambahan pada Class Pencarian Mahasiswa

```

35
36 //Algoritma pencarian data Binary Searching
37 public int FindBinarySearch (int cari, int left, int right) {
38     int mid;
39     if (right >= left) {
40         mid = (left + right) / 2;
41         if (cari == listMhs[mid].nim) {
42             return (mid);
43         } else if (listMhs[mid].nim > cari ) {
44             return FindBinarySearch(cari, left, mid - 1);
45         } else {
46             return FindBinarySearch(cari, mid + 1, right);
47         }
48     }
49     return -1;
50 }
    
```

Tambahan pada Class Main

```

System.out.println(x:"=====");
System.out.println(x:"Menggunakan Binary Search");
posisi = data.FindBinarySearch(cari, left:0, jumMhs - 1);
data.Tampilposisi(cari, posisi);
data.TampilData (cari, posisi);
}
    
```

➤ Hasil Output :

```

=====
Masukan data Mahasiswa secara urut dari Nim Terkecil
=====
Nim      : 2017
Nama     : Dewi Lestari
Umur     : 23
IPK      : 3.5
=====
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4
=====
Nim      : 2019
Nama     : Danang Adi
Umur     : 22
IPK      : 3.7
=====
Nim      : 2020
Nama     : Budi Prakarsa
Umur     : 20
IPK      : 2.9
=====
Nim      : 2021
Nama     : Vania Siti
Umur     : 20
IPK      : 2
=====
    
```

```

=====
Data keseluruhan Mahasiswa :
Nim = 2017
Nama = Dewi Lestari
Umur = 23
IPK = 3.5
=====
Nim = 2018
Nama = Sinta Sanjaya
Umur = 22
IPK = 4.0
=====
Nim = 2019
Nama = Danang Adi
Umur = 22
IPK = 3.7
=====
Nim = 2020
Nama = Budi Prakarsa
Umur = 20
IPK = 2.9
=====
Nim = 2021
Nama = Vania Siti
Umur = 20
IPK = 2.0
=====
    
```

```

=====
Pencarian Data :
Masukkan Nim Mahasiswa yang dicari :
NIM : 2018
Menggunakan sequential Search
Data : 2018 ditemukan pada indeks 1
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4.0
=====
Menggunakan Binary Search
Data : 2018 ditemukan pada indeks 1
Nim      : 2018
Nama     : Sinta Sanjaya
Umur     : 22
IPK      : 4.0
    
```

6.3.3. Pertanyaan

1. Tunjukkan pada kode program yang mana proses divide dijalankan!

```

37     public int FindBinarySearch (int cari, int left,
38         int mid;
39         if (right >= left) {
40             mid = (left + right) / 2;
    
```

Baris ini, mencari nilai tengah dari rentang pencarian dengan membagi nilai left dan right menjadi dua

2. Tunjukkan pada kode program yang mana proses conquer dijalankan!

```

37     public int FindBinarySearch (int cari, int left, int right) {
38         int mid;
39         if (right >= left) {
40             mid = (left + right) / 2;
41             if (cari == listMhs[mid].nim) {
42                 return (mid);
43             } else if (listMhs[mid].nim > cari) {
44                 return FindBinarySearch(cari, left, mid - 1);
45             } else {
46                 return FindBinarySearch(cari, mid + 1, right);
47             }
48         }
49         return -1;
    
```

```
1) kondisi if (cari == listMhs[mid].nim) {
    return (mid);
```

Memeriksa apakah elemen yang dicari sama dengan elemen di posisi tengah array. Jika iya, mengembalikan indeks tengah sebagai hasil pencarian

```
2) else if (listMhs[mid].nim > cari )
    { return FindBinarySearch (cari, left, mid - 1);
    } else { return FindBinarySearch(cari, mid + 1, right);
```

Membagi rentang pencarian menjadi dua bagian berdasarkan elemen yang berada di posisi tengah. Jika elemen yang dicari lebih kecil dari elemen tengah, akan mencari di setengah kiri array, dan jika lebih besar, akan mencari di setengah kanan array. Ini merupakan proses untuk menemukan elemen yang dicari

3. Jika data Nim yang dimasukkan tidak urut. Apakah program masih dapat berjalan? Mengapa demikian!

➤ **Jika data NIM yang dimasukkan tidak dalam keadaan terurut, program masih akan berjalan, tetapi hasilnya mungkin tidak dapat diandalkan. Algoritma pencarian binary biasanya hanya berfungsi dengan benar pada data yang telah diurutkan**

4. Jika Nim yang dimasukkan dari NIM terbesar ke terkecil (missal : 20215, 20214, 20212, 20211, 20210) dan elemen yang dicari adalah 20210. Bagaimana hasil dari binary search? Apakah sesuai? Jika tidak sesuai maka ubahlah kode program binary seach agar hasilnya sesuai

➤ **Hasil pada Binary Search Data tidak ditemukan, karena NIM diurutkan dari NIM terbesar ke terkecil, tetapi masih menggunakan algoritma pencarian binary standar, hasilnya mungkin tidak sesuai sebab algoritma tersebut mengasumsikan bahwa data diurutkan dari terkecil ke terbesar**

```
=====
Pencarian Data :
Masukkan Nim Mahasiswa yang dicari :
NIM : 20210
Menggunakan sequential Search
Data : 20210ditemukan pada indeks 4
Nim      : 20210
Nama     : Dera
Umur     : 20
IPK      : 4.0
=====
Menggunakan Binary Search
Data 20210tidak ditemukan
```

➤ **Maka, diperbaiki pada kode program sebagai berikut :**

```

36 //Algoritma pencarian data Binary Searching
37 public int FindBinarySearch (int cari, int left, int right) {
38     int mid;
39     if (right >= left) {
40         mid = (left + right) / 2;
41         if (listMhs[mid].nim == cari) {
42             //if (cari == listMhs[mid].nim) {
43                 return (mid);
44             } else if (listMhs[mid].nim > cari) {
45                 return FindBinarySearch(cari, mid + 1, right);
46             //return FindBinarySearch(cari, left, mid - 1);
47             } else {
48                 return FindBinarySearch(cari, left, mid - 1);
49             //return FindBinarySearch(cari, mid + 1, right);
50             }
51         }
52     return -1;
53 }

```

Hasil Output :

```

=====
Pencarian Data :
Masukkan Nim Mahasiswa yang dicari :
NIM : 20210
Menggunakan sequential Search
Data : 20210 ditemukan pada indeks 4
Nim      : 20210
Nama     : Fadya
Umur     : 18
IPK      : 3.0
=====
Menggunakan Binary Search
Data : 20210 ditemukan pada indeks 4
Nim      : 20210
Nama     : Fadya
Umur     : 18
IPK      : 3.0

```

5. Modifikasilah program diatas yang mana jumlah mahasiswa yang di inputkan sesuai dengan masukan dari keyboard.

➤ Bagian Modifikasi

Class Pencarian Mahasiswa :

```

2
3 /**
4  * PencarianMhs
5  */
6 public class PencarianMhs {
7     Mahasiswa06 listMhs[];
8     int idx;
9     PencarianMhs (int size){
10         listMhs = new Mahasiswa06[size];
11         idx = 0;
12     }
13 }

```

Class Main :

```

4     public static void main(String[] args) {
5         Scanner s = new Scanner (System.in);
6         Scanner s1 = new Scanner (System.in);
7
8         System.out.println(x:"Masukkan Jumlah Mahasiswa : ");
9         int jumMhs = s.nextInt();
10        s.nextLine();
11
12        PencarianMhs data = new PencarianMhs (jumMhs);
13
14        System.out.println(x:"-----");
15        System.out.println(x:"Masukkan data Mahasiswa secara urut daru Nim Terkecil ");
16        for (int i = 0; i < jumMhs; i++) {
17            System.out.println(x:"-----");
18            System.out.print(s:"Nim \t : ");
19            int nim = s.nextInt ();
20            System.out.print(s:"Nama \t : ");
21            String nama = s1.nextLine ();
22            System.out.print(s:"Umur \t : ");
23            int umur = s.nextInt ();
24            System.out.print(s:"IPK \t : ");
25            double ipk= s.nextDouble ();

```

➤ Hasil Output :

```
Masukkan Jumlah Mahasiswa :
2
-----
Masukan data Mahasiswa secara urut daru Nim Terkecil
-----
Nim      : 2022
Nama     : Dea
Umur     : 19
IPK      : 4
-----
Nim      : 2021
Nama     : Nissa
Umur     : 18
IPK      : 4
-----
Data keseluruhan Mahasiswa :
Nim = 2022
Nama = Dea
Umur = 19
IPK = 4.0
-----
Nim = 2021
Nama = Nissa
Umur = 18
IPK = 4.0
-----
```

```
-----
Pencarian Data :
Masukkan Nim Mahasiswa yang dicari :
NIM : 2021
Menggunakan sequential Search
Data : 2021ditemukan pada indeks 1
Nim      : 2021
Nama     : Nissa
Umur     : 18
IPK      : 4.0
=====
Menggunakan Binary Search
Data : 2021ditemukan pada indeks 1
Nim      : 2021
Nama     : Nissa
Umur     : 18
IPK      : 4.0
```

6.4. Percobaan Pengayaan Divide and Conquer

6.4.1. Langkah-langkah Percobaan Merge Sort

1. Buatlah Package baru pada NetBeans dengan nama **MergeSortTest**
7. Tambahkan class **MergeSorting** pada package tersebut
8. Pada class **MergeSorting** buatlah method **mergeSort** yang menerima parameter data array yang akan diurutkan

```
public void mergeSort(int[] data) {
```

9. Buatlah method **merge** untuk melakukan proses penggabungan data dari bagian kiri dan kanan.

```
private void merge(int data[], int left, int middle, int right) {
```

10. Implementasikan proses **merge** sebagai berikut.

```
public void merge(int data[], int left, int middle, int right) {
    int[] temp = new int[data.length];
    for (int i = left; i <= right; i++) {
        temp[i] = data[i];
    }
    int a = left;
    int b = middle + 1;
    int c = left;

    //membandingkan setiap bagian
    while (a <= middle && b <= right) {
        if (temp[a] <= temp[b]) {
            data[c] = temp[a];
            a++;
        } else {
            data[c] = temp[b];
            b++;
        }
        c++;
    }
    int s = middle - a;
    for (int i = 0; i <= s; i++) {
        data[c + i] = temp[a + i];
    }
}
```

11. Buatlah method **sort**

```
private void sort(int data[], int left, int right) {
```

12. Implementasikan kode berikut pada method **sort**

```
//membagi menjadi 2 bagian dan dibagi kembali hingga tidak dapat dibagi kembali
private void sort(int data[], int left, int right) {
    if (left < right) {
        int middle = (left + right) / 2;
        sort(data, left, middle);
        sort(data, middle + 1, right);
        merge(data, left, middle, right);
    }
}
```

- 13 Pada method **mergeSort**, panggil method **sort** dengan parameter data yang ingin diurutkan serta range data awal sampai dengan akhir.

- 14 Tambahkan method **printArray**

```
public void printArray(int arr[]){
    int n= arr.length;
    for (int i=0; i<n;i++)
    {
        System.out.print(arr[i]+" ");
    }
    System.out.println();
}
```

- 15 Sebagai langkah terakhir, deklarasikan data yang akan diurutkan kemudian panggil proses sorting pada class **SortMain**

```
class SortMain {

    public static void main(String[] args) {
        int data[] ={10,40,30,50,70,20,100,90};
        System.out.println("sorting dengan merge sort");
        MergeSorting mSort= new MergeSorting();
        System.out.println("data awal");
        mSort.printArray(data);
        mSort.mergeSort(data);
        System.out.println("setelah diurutkan");
        mSort.printArray(data);
    }
}
```

6.4.2. Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```

sorting dengan merge sort
data awal
10 40 30 50 70 20 100 90
setelah diurutkan
10 20 30 40 50 70 90 100
BUILD SUCCESSFUL (total time: 1 second)
    
```

➤ Hasil Input :

Class Merge Sorting

```

1  public class MergeSorting {
2      public void mergeSort (int[] data) {
3          sort (data, left:0, data.length- 1);
4      }
5      public void merge (int data[], int left, int middle, int right) {
6          int [] temp = new int [data.length];
7          for (int i = left; i <= right; i++) {
8              temp[i] = data [i];
9          }
10         int a = left;
11         int b = middle + 1;
12         int c = left;
13
14         //Membandingkan setiap bagian
15         while (a<= middle && b <= right) {
16             if (temp[a] <= temp[b]) {
17                 data [c] = temp [a];
18                 a++;
19             } else {
20                 data [c] = temp [b];
21                 b++;
22             }
23             c++;
24         }
25
26         int s = middle - a;
27         for (int i = 0; i <= s; i++) {
28             data[c + i] = temp [a + i];
29         }
30         //Membagi menjadi 2 bagian dan dibagi kembali hingga tidak dapat dibagi kembali
31         private void sort (int [] data, int left, int right) {
32             if (left < right) {
33                 int middle = (left + right) / 2;
34                 sort (data, left, middle);
35                 sort (data, middle + 1, right);
36                 merge (data, left, middle, right);
37             }
38         }
39         public void printArray (int arr[]) {
40             int n = arr.length;
41             for (int i = 0; i < n; i++) {
42                 System.out.print(arr[i]+ " ");
43             }
44             System.out.println();
45         }
46     }
    
```


Class Main

```
J SortingMain.java > SortingMain > main(String[])
1 public class SortingMain {
    Run | Debug
2     public static void main(String[] args) {
3         int data [] = {10, 40, 30, 50, 70, 20, 100, 90};
4         System.out.println(x:"Sorting dengan Merge Sort");
5         MergeSorting mSort= new MergeSorting ();
6         System.out.println(x:"Data Awal");
7         mSort.printArray (data);
8         mSort.mergeSort (data);
9         System.out.println(x:"Setelah Diurutkan");
10        mSort.printArray (data);
    }
```

➤ Hasil Output :

```
Sorting dengan Merge Sort
Data Awal
10 40 30 50 70 20 100 90
Setelah Diurutkan
10 20 30 40 50 70 90 100
PS C:\Users\TOSHIBA\Semester 2\Pratikum06> |
```

6.5. Latihan Praktikum

1. Modifikasi percobaan searching diatas yang menggunakan Searching array of object dengan ketentuan berikut ini
 - Pencarian dilakukan berdasarkan Nama Mahasiswa (gunakan Algoritma binary Search)
 - Buat aturan untuk mendeteksi hasil pencarian lebih dari 1 hasil dalam bentuk kalimat peringatan!

➤ Bagian Modifikasi

Class PencarianMhs

//Pada bagian method FindBinarySearch

```
minggu7 > J PencarianMhs.java > PencarianMhs > FindSeqSearch(int)
6 public class PencarianMhs {
42 //Algoritma pencarian data Binary Searching
43 public int FindBinarySearch (String cari, int left, int right) {
44     while (left <= right) {
45         int mid = left + (right - left) / 2;
46         int res = cari.compareToIgnoreCase(listMhs[mid].nama);
47         if (res == 0)
48             return mid;
49         if (res > 0)
50             left = mid + 1;
51         else
52             right = mid - 1;
53     }
54     return -1;
55 }
```

Class Main

//Merubah pada bagian pencarian menggunakan nama & memberi peringatan

```
47 //System.out.println("Menggunakan Binary Search");
48 System.out.print(s:"Nama : ");
49 String cari = s1.next();
50 System.out.println(x:"Menggunakan binary Search");
51 int posisi = data.FindBinarySearch(cari, left:0, jumMhs - 1);
52 if (posisi != -1) {
53     System.out.println("Data dengan nama '" + cari + "' ditemukan pada indeks " + posisi);
54     // Tampilkan data mahasiswa yang ditemukan
55     data.listMhs[posisi].tampil();
56     // Cek apakah ada data dengan nama yang sama setelahnya
57     int i = posisi + 1;
58     boolean multipleResults = false;
59     while (i < jumMhs && data.listMhs[i].nama.equalsIgnoreCase(cari)) {
60         multipleResults = true;
61         i++;
62     }
63     if (multipleResults) {
64         System.out.println(x:"Catatan: Ada lebih dari satu hasil dengan nama yang sama.");
65     }
66 } else {
67     System.out.println("Data dengan nama '" + cari + "' tidak ditemukan");
68 }
```

➤ Hasil Output :

```
Masukkan Jumlah Mahasiswa :
3
-----
Masukan data Mahasiswa secara urut dari Nim Terkecil
-----
Nim      : 218
Nama     : John
Umur     : 17
IPK      : 3
-----
Nim      : 219
Nama     : Dea
Umur     : 19
IPK      : 4
-----
Nim      : 220
Nama     : Dea
Umur     : 18
IPK      : 4
-----
```

```
-----
Data keseluruhan Mahasiswa :
Nim = 218
Nama = John
Umur = 17
IPK = 3.0
-----
Nim = 219
Nama = Dea
Umur = 19
IPK = 4.0
-----
Nim = 220
Nama = Dea
Umur = 18
IPK = 4.0
-----
-----
Pencarian Data :
Masukkan Nim Mahasiswa yang dicari :
Nama : Dea
Menggunakan binary Search
Data dengan nama 'Dea' ditemukan pada indeks 1
Nim = 219
Nama = Dea
Umur = 19
IPK = 4.0
Catatan: Ada lebih dari satu hasil dengan nama yang sama.
```