

## 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

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

Buku
kodeBuku: int judul: String tahunTerbit: int pengarang: String stock : int
Buku(kodeBuku:int, judul: String, tahunTerbit: int, pengarang: String, stock:int) tampil(): void TampilDataBuku():void

Berdasarkan class diagram di atas, akan dibuat class Buku yang berfungsi untuk membuat objek buku yang akan dimasukan ke dalam sebuah array. Terdapat sebuah konstruktor berparameter dan juga Method tampil() untuk menampilkan semua attribute yang ada.

PencarianBuku
listBuku: Buku[5] idx: int
Tambah(bk: Buku): void tampil(): void FindSeqSearch(int cari): int Tampilposisi(int x,int pos): void TampilData(int x,int pos) :void

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

#### 6.2.1. Langkah-langkah Percobaan Sequential Search

1. Buatlah Project baru pada dengan nama **TestSearching**
2. Kemudian buat packages baru dengan nama **P7**
3. Buat class **BukuNoAbsen**, kemudian deklarasikan atribut berikut ini:

```
public class Buku {
    int kodeBuku;
    String judulBuku;
    int tahunTerbit;
    String pengarang;
    int stock;
```

4. Buatlah konstruktor dengan nama **Buku** dengan parameter (**String kodeBuku, String judulBuku, int tahunTerbit, String pengarang, int stock**) kemudian Isi konstruktor tersebut dengan kode berikut!

```
public Buku(int kodeBuku, String judulBuku, int tahunTerbit, String pengarang, int stock) {
    kodeBuku = kodeBuku;
    judulBuku = judulBuku;
    tahunTerbit = tahunTerbit;
    pengarang = pengarang;
    stock = stock;
}
```

Catatan : Perhatikan konstruktor diatas! Apakah sudah benar? Jika belum, maka perbaiki konstruktor diatas

5. Buatlah method **tampilDataBuku** bertipe void.

```
public void tampilDataBuku(){
    System.out.println("=====");
    System.out.println("Kode buku :"+kodeBuku);
    System.out.println("Judul buku : "+judulBuku);
    System.out.println("Tahun Terbit : "+tahunTerbit);
    System.out.println("Pengarang : "+pengarang);
    System.out.println("Stock : "+stock);
}
```

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

```
public class PencarianBuku {
    Buku listBk[] = new Buku[5];
    int idx;
}
```

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

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

8. Tambahkan method **tampil()** di dalam class **PencarianBukuNoAbsen** Method **tampil()** digunakan untuk menampilkan semua data buku yang ada di dalam class tersebut! Perhatikan penggunaan sintaks for yang menggunakan konsep for-each. Syntax tersebut akan memberikan tanda error berupa garis merah, Perbaiki agar tidak ada error!

```
void tampil() {
    for (Buku m : listBk[]) {
        m.tampilDataBuku();
    }
}
```

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

```

public int FindSeqSearch(int cari) {
    int posisi = 2;
    for (int j = 0; j < listBk.length; j++) {
        if (listBk[j].kodeBuku==cari){
            j = posisi;
            break;
        }
    }
    return posisi;
}

```

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

```

public void Tampilposisi(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 class baru dengan nama **BukuMainNoAbsen** tambahkan method **main**. buatlah sebuah objek PencarianMhs dan buatlah 5 objek buku kemudian tambahkan semua objek buku tersebut dengan memanggil fungsi tambah pada objek PencarianBuku.

```

Scanner s = new Scanner(System.in);
Scanner s1 = new Scanner(System.in);

PencarianBuku data = new PencarianBuku();
int jumBuku = 5;

System.out.println("-----");
System.out.println("Masukkan data Buku secara Urut dari KodeBuku Terkecil : ");
for (int i = 0; i < jumBuku ; i++) {
    System.out.println("-----");
    System.out.print("Kode Buku \t: ");
    int kodeBuku = s.nextInt();
    System.out.print("Judul buku \t : ");
    String judulBuku = s1.nextLine();
    System.out.print("Tahun Terbit \t : ");
    int tahunTerbit= s.nextInt();
    System.out.print("Pengarang \t : ");
    String pengarang= s1.nextLine();
    System.out.print("Stock \t : ");
    int stock= s.nextInt();

    Buku m = new Buku(kodeBuku, judulBuku, tahunTerbit, stock, pengarang);
    data.tambah(m);
}
System.out.println("-----");
System.out.println("Data keseluruhan Mahasiswa : ");

```

Note : perbaiki kode jika terdapat kesalahan.

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

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

13. Untuk melakukan pencarian berdasarkan kode buku. Buatlah variable **cari** yang dapat menampung masukan dari keyboard lalu panggil method **FindSeqSearch** dengan isi parameternya adalah variable cari. Untuk menampilkan index data yang dicari panggil method **TampilPosisi**

```
System.out.println("_____");
System.out.println("_____");
System.out.println("Pencarian Data : ");
System.out.println("Masukkan Kode Buku yang dicari: ");
System.out.print("Kode Buku : ");
int cari = s.nextInt();
System.out.println("menggunakan sequential Search");
int posisi = data.FindSeqSearch(cari);
data.Tampilposisi(cari, posisi);
```

Verifikasi hasil pencarian :

```
Masukkan data Buku secara Urut dari KodeBuku Terkecil :
-----
Kode Buku      : 111
Judul buku     : Algoritma
Tahun Terbit   : 2019
Pengarang      : Wahyuni
Stock          : 5
-----
Kode Buku      : 123
Judul buku     : Big Data
Tahun Terbit   : 2020
Pengarang      : Susilo
Stock          : 3
-----
Kode Buku      : 125
Judul buku     : Desain UI
Tahun Terbit   : 2021
Pengarang      : Supriadi
Stock          : 3
-----
Kode Buku      : 126
Judul buku     : Web Programming
Tahun Terbit   : 2022
Pengarang      : Pustaka Adi
Stock          : 2
-----
Kode Buku      : 127
Judul buku     : Etika Mahasiswa
Tahun Terbit   : 2023
Pengarang      : Darmawan Adi
Stock          : 2
```

```
Data keseluruhan Buku :
=====
Kode buku :111
Judul buku : Algoritma
Tahun Terbit : 2019
Pengarang :Wahyuni
Stock : 5
=====
Kode buku :123
Judul buku : Big Data
Tahun Terbit : 2020
Pengarang :Susilo
Stock : 3
=====
Kode buku :125
Judul buku : Desain UI
Tahun Terbit : 2021
Pengarang :Supriadi
Stock : 3
=====
Kode buku :126
Judul buku : Web Programming
Tahun Terbit : 2022
Pengarang :Pustaka Adi
Stock : 2
=====
Kode buku :127
Judul buku : Etika Mahasiswa
Tahun Terbit : 2023
Pengarang :Darmawan Adi
Stock : 2
```

```
Pencarian Data :
Masukkan Kode Buku yang dicari:
Kode Buku : 111
menggunakan sequential Search
data : 111 ditemukan pada indeks 0
```

Apakah pencarian pada program anda sudah sesuai? Jika belum perbaiki kode sehingga pencarian sesuai.

14. Buatlah method **TampilData** bertipe void pada class **PencarianBukuNoAbsen** dan tambahkan isi dari method **TampilData**.

```
public void TampilData(int x,int pos)
{
    if (pos!= -1) {
        System.out.println("Kode Buku\t : " + x );
        System.out.println("Judul\t : "+listBk[pos].judulBuku);
        System.out.println("Tahun Terbit\t : "+listBk[pos].tahunTerbit);
        System.out.println("Pengarang\t : "+listBk[pos].pengarang);
        System.out.println("Stock\t : "+listBk[pos].stock);
    } else {
        System.out.println("data " + x + "tidak ditemukan");
    }
}
```

15. Di dalam method **main()**, Lakukan pemanggilan method **TampilData** dari class **PencarianMhs**.

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

16. Jalankan dan amati hasilnya.

### 6.2.2. Verifikasi Hasil Percobaan

Cocokkan hasil kode program anda dengan gambar berikut ini. Jika hasil belum cocok, perbaiki kode program Anda!

Jika data ditemukan:

```
Pencarian Data :
Masukkan Kode Buku yang dicari:
Kode Buku : 111
menggunakan sequential Search
data : 111 ditemukan pada indeks 0
Kode Buku      : 111
Judul          : Algoritma
Tahun Terbit   : 2019
Pengarang      : Wahyuni
Stock          : 5
```

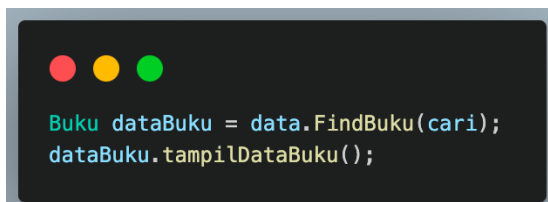
Jika data tidak ditemukan :

```
Pencarian Data :
Masukkan Kode Buku yang dicari:
Kode Buku : 124
menggunakan sequential Search
data 124 tidak ditemukan
data 124 tidak ditemukan
Mamlatuls-MacBook-Air:Praktikum mamlatulhaniah$
```

Note : hasil kode yang ditampilkan diatas adalah hasil pencarian dengan menampilkan index dan data yang dicari

### 6.2.3. Pertanyaan

1. Jelaskan fungsi **break** yang ada pada method **FindSeqSearch**!
2. Jika Data Kode Buku yang dimasukkan tidak terurut dari kecil ke besar. Apakah program masih dapat berjalan? Apakah hasil yang dikeluarkan benar? Tunjukkan hasil screenshoot untuk bukti dengan kode Buku yang acak. Jelaskan Mengapa hal tersebut bisa terjadi?
3. Buat method baru dengan nama **FindBuku** menggunakan konsep sequential search dengan tipe method dari **FindBuku** adalah **BukuNoAbsen**. Sehingga Anda bisa memanggil method tersebut pada class **BukuMain** seperti gambar berikut :



```
Buku dataBuku = data.FindBuku(cari);
dataBuku.tampilDataBuku();
```

### 6.3. Searching / Pencarian Menggunakan Binary Search

#### 6.3.1. Langkah-langkah Percobaan Binary Search

1. Pada percobaan 6.2.1 (sequential search) tambahkan method **FindBinarySearch** bertipe integer pada class **PencarianBukuNoAbsen**. 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 = (right) / 2;
        if (cari == listBk[mid].kodeBuku) {
            return (mid);
        } else if (listBk[mid].kodeBuku > cari) {
            return FindBinarySearch(cari, left, mid);
        } else {
            return FindBinarySearch(cari, mid, right);
        }
    }
    return -1;
}
    
```

2. Panggil method **FindBinarySearch** di kelas **BukuMainNoAbsen**. Kemudian panggil method **tampilposisi** dan **tampilData**

```

System.out.println("=====");
System.out.println("menggunakan binary Search");
posisi = data.FindBinarySearch(cari, 0, jumBuku - 1);
data.Tampilposisi(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. Jika hasil belum cocok, **perbaiki kode program Anda!**

hasil kode yang ditampilkan berikut ini adalah hasil pencarian dengan menampilkan index dan data yang dicari. Hasil running penambahan data dan menampilkan data dapat anda lakukan seperti pada percobaan 6.2.1 langkah Nomor 13.



```
=====
Pencarian Data :
Masukkan Kode Buku yang dicari:
Kode Buku : 126
menggunakan sequential Search
data : 126 ditemukan pada indeks 3
Kode Buku      : 126
Judul          : Web Programming
Tahun Terbit   : 2022
Pengarang      : Pustaka Adi
Stock          : 2
=====
menggunakan binary Search
data : 126 ditemukan pada indeks 3
Kode Buku      : 126
Judul          : Web Programming
Tahun Terbit   : 2022
Pengarang      : Pustaka Adi
Stock          : 2
MamluatulS-MacBook-Air:Praktikum_mamluatulhaniah$
```


### 6.3.3. Pertanyaan

1. Tunjukkan pada kode program yang mana proses divide dijalankan!
2. Tunjukkan pada kode program yang mana proses conquer dijalankan!
4. Jika data Kode Buku yang dimasukkan tidak urut. Apakah program masih dapat berjalan? Mengapa demikian! Tunjukkan hasil screenshot untuk bukti dengan kode Buku yang acak. Jelaskan Mengapa hal tersebut bisa terjadi?
3. Jika Kode Buku yang dimasukkan dari Kode Buku 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 search agar hasilnya sesuai!

## 6.4. Percobaan Pengayaan Divide and Conquer

### 6.4.1. Langkah-langkah Percobaan Merge Sort

1. Buatlah Package baru didalam package P5 dengan nama **MergeSortTest**
2. Tambahkan class **MergeSortingNoAbsen** pada package tersebut
3. Pada class **MergeSortingNoAbsen** buatlah method **mergeSort** yang menerima parameter data array yang akan diurutkan



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

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



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

- 5 Implementasikan proses **merge** sebagai berikut.



```
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;

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];
}
```

- 6 Buatlah method **sort**



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

- 7 Implementasikan kode berikut pada method **sort**



```
if (left < right) {
    int middle = (left + right) / 2;
    sort(data, left, middle);
    sort(data, middle + 1, right);
    merge(data, left, middle, right);
}
```

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

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

- 10 Sebagai langkah terakhir, deklarasikan data yang akan diurutkan kemudian panggil proses sorting pada class **MergeSortMainNoAbsen**. Tambahkan fungsi main pada kelas tersebut, kemudian tuliskan kode berikut didalam fungsi main.

```
int data[] = {10,40,30,50,70,20,100,90};
System.out.println("sorting dengan merge sort");
MergeSort mSort= new MergeSort();
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
```

#### 6.5. Latihan Praktikum

- Modifikasi percobaan searching diatas dengan ketentuan berikut ini
  - Ubah tipe data dari kode Buku yang awalnya int menjadi String
  - Tambahkan method untuk pencarian kode Buku (bertipe data String) dengan menggunakan sequential search dan binary search.
- Modifikasi percobaan searching diatas dengan ketentuan berikut ini



- Tambahkan method pencarian judul buku menggunakan sequential search dan binary search. Sebelum dilakukan searching dengan binary search data harus dilakukan pengurutan dengan menggunakan algoritma Sorting (bebas pilih algoritma sorting apapun)! Sehingga ketika input data acak, maka algoritma searching akan tetap berjalan
- Buat aturan untuk mendeteksi hasil pencarian judul buku yang lebih dari 1 hasil dalam bentuk kalimat peringatan! Pastikan algoritma yang diterapkan sesuai dengan kasus yang diberikan!