Laporan Hasil Praktikum

Algoritma & Struktur Data

Jobsheet 6

MUHAMMAD PEARL OCSHADA

244107020064

TI 1-E

Program Studi Teknik Informatika

Jurusan Teknologi Informasi

POLINEMA

2024

Searching

Percobaan1 : Searching menggunakan Sequential Search

1. Pada pertemuan Jobsheet 7 ini akan menggunakan class Mahasiswa, MahasiswaBerprestasi, dan MahasiswaDemo pada pertemuan Jobsheet 6
2. Buat folder baru bernama Jobsheet7 di dalam repository Praktikum ASD, kemudian buka ketiga class dari Jobsheet 6 tersebut dan copy ke folder Jobsheet 7
3. Tambahkan method sequentialSearching bertipe integer dengan parameter cari bertipe double pada class MahasiswaBerprestasi. Kemudian Deklarasikan isi method sequentialSearching dengan algoritma pencarian data menggunakan teknik sequential searching.
4. Buatlah method tampilPoisisi bertipe void dan Deklarasikan isi dari method tampilPoisisi pada class MahasiswaBerprestasi.
5. Pada class MahasiswaBerprestasi, buatlah method tampilDataSearch bertipe void dan Deklarasikan isi dari method tampilDataSearch .
6. Pada class MahasiswaDemo , tambahkan kode program berikut ini untuk melakukan pencarian data dengan algoritma sequential searching.
7. Jalankan dan amati hasilnya.
    a. Source Code class mahasiswa20

```
package Praktikum_ASD.ALSD_Jobsheet6;


 public class mahasiswa20 {
     String nim;
     String nama;
     String kelas;
     double ipk;


     mahasiswa20(){


     }
     public mahasiswa20(String nm,String name,String kls, double ip){
         nim = nm;
         nama = name;
         ipk = ip;
         kelas = kls;
     }
     void tampilInformasi(){
         System.out.println("Nama :" + nama);
         System.out.println("NIM :" + nim);
         System.out.println("Kelas :"+ kelas);
         System.out.println("IPK : " + ipk);
         System.out.println();
     }
 }
```

b. Source Code class mahasiswaBerprestasi

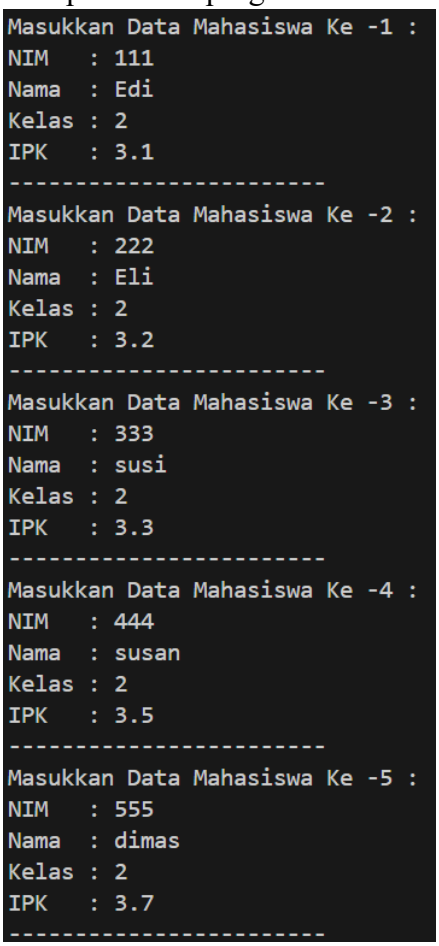
```
package Praktikum_ASD.ALSD_Jobsheet6;

 public class mahasiswaBerprestasi {
     mahasiswa20 [] listMhs = new mahasiswa20[5];
     int idx;
     void tambah(mahasiswa20 m){
         if (idx < listMhs.length) {
             listMhs[idx]=m;
             idx++;
         }else {
             System.out.println("data sudah penuh");
         }
     }
     void tampil(){
         for(mahasiswa20 m :listMhs){
             if (m != null) {
                 m.tampilInformasi();
             }
         }
     }
     void bubbleSort(){
         for (int i = 0; i < listMhs.length-1; i++) {
             for (int j = 1; j < listMhs.length; j++) {
                 if (listMhs[j].ipk > listMhs[j-1].ipk) {
                     mahasiswa20 temp = listMhs[j];
                     listMhs[j] = listMhs [j-1];
                     listMhs[j-1]=temp;
                 }
             }
         }
     }
     void selectionSort(){
         for (int i = 0; i < listMhs.length; i++) {
             int idxMin = i;
             for (int j = 0; j < listMhs.length; j++) {
                 if (listMhs[j].ipk < listMhs[idxMin].ipk) {
                 idxMin = j;
                 }
             }
             mahasiswa20 tmp = listMhs[idxMin];
             listMhs[idxMin] = listMhs[i];
             listMhs[i] = tmp;
```

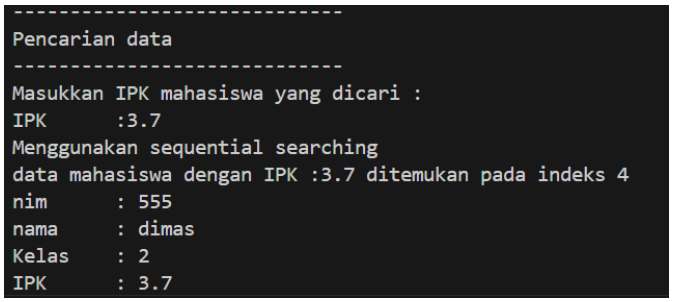
```
}
    }
    void insertionSort(){
        for (int i = 0; i < listMhs.length; i++) {
            mahasiswa20 temp = listMhs[i];
            int j =1;
            while (j>0 && listMhs[j-1].ipk > temp.ipk){
                listMhs[j] = listMhs[j-1];
                j--;
            }
            listMhs[j] = temp;
        }
    }
    int sequentialSearching(double cari){
        int posisi = -1;
        for (int j=0; j< listMhs.length;j++){
            if(listMhs[j].ipk == cari){
                posisi = j;
                break;
            }
        }
        return posisi;
    }

    void tampilPosisi(double x, int pos){
        if(pos!=-1){
            System.out.println("data mahasiswa dengan IPK :"+x+" ditemukan pada
indeks "+pos);
        } else{
            System.out.println("data "+x+" tidak ditemukan");
        }
    }
    void tampilDataSearch(double x, int pos){
        if (pos != -1) {
            System.out.println("nim\t : "+listMhs[pos].nim);
            System.out.println("nama\t : "+listMhs[pos].nama);
            System.out.println("Kelas\t : "+listMhs[pos].kelas);
            System.out.println("IPK\t : "+x);
        }else{
            System.out.println("Data mahasiswa dengan IPK "+x+" tidak ditemukan");
        }
    }
 }
```

c. Source code mahasiswaDemo20

```
package Praktikum_ASD.ALSD_Jobsheet6;
 import java.util.Scanner;
 public class mahasiswaDemo20 {
     public static void main(String[] args) {
         Scanner sc = new Scanner(System.in);
         mahasiswaBerprestasi list = new mahasiswaBerprestasi();
         int jumMhs=5;

         for (int i = 0; i < jumMhs; i++) {
             System.out.println("Masukkan Data Mahasiswa Ke -"+(i+1)+" : ");
             System.out.print("NIM   : ");
             String nim = sc.nextLine();
             System.out.print("Nama  : ");
             String nama = sc.nextLine();
             System.out.print("Kelas : ");
             String kelas = sc.nextLine();
             System.out.print("IPK   : ");
             String ip = sc.nextLine().replace(",", ".");
             double ipk = Double.parseDouble(ip);
             System.out.println("------------------------");
             list.tambah(new mahasiswa20(nim, nama, kelas, ipk));
         }
         list.tampil();
         // melakukan pencarian data sequential
         System.out.println("----------------------------");
         System.out.println("Pencarian data");
         System.out.println("----------------------------");
         System.out.println("Masukkan IPK mahasiswa yang dicari :");
         System.out.print("IPK\t :");
         String cariStr = sc.nextLine().replace(",", ".");
         double cari = Double.parseDouble(cariStr);

         System.out.println("Menggunakan sequential searching");
         double posisi = list.sequentialSearching(cari);
         int pss = (int)posisi;
         list.tampilPosisi(cari, pss);
         list.tampilDataSearch(cari, pss);

     }
 }
```

d. Compile & run program





Pertanyaan

1. Perbedaan metode tampilDataSearch dan tampilPosisi pada class MahasiswaBerprestasi:
   tampilPosisi(double x, int pos):
   -Metode ini hanya menampilkan indeks posisi dari mahasiswa dengan IPK yang dicari.
   -Jika data ditemukan, akan ditampilkan dalam format:
   -"Data Mahasiswa Dengan IPK : ditemukan pada indeks " tampilDataSearch(double x, int pos):
   - Selain menampilkan posisi, metode ini juga menampilkan informasi lengkap mahasiswa, seperti NIM, Nama, Kelas, dan IPK.
   - Jika mahasiswa ditemukan, metode akan mencetak detail mahasiswa yang bersangkutan.
2. Pada kode program tersebut break digunakan untuk menghentikan loop setelah menemukan mahasiswa dengan IPK yang sesuai dengan nilai cari. Jika ditemukan, variabel posisi akan menyimpan indeks mahasiswa tersebut, lalu loop akan berhenti, sehingga tidak perlu mencari lebih lanjut.

Percobaan2 : Searching menggunakan binarySearch

1. Pada percobaan 6.2.1 (sequential search) tambahkan method findBinarySearch bertipe integer pada class MahasiswaBerprestasi. Kemudian Deklarasikan isi method findBinarySearch dengan

algoritma pencarian data menggunakan teknik binary searching

```
int findBinarySearch(double cari, int left, int right){
         int mid;
         if (right>=left) {
             mid =(left+right)/2;
             if (cari == listMhs[mid].ipk) {
                 return (mid);
             }else if (listMhs[mid].ipk > cari) {
                 return findBinarySearch(cari, left, mid-1);
             }else {
                 return findBinarySearch(cari, mid+1,right);
             }
         }
         return  -1;
     }
```

2. Panggil method findBinarySearch terdapat pada class MahasiswaBerprestasi di kelas MahasiswaDemo. Kemudia panggil method tampilPosisi dan tampilDataSearch

```
System.out.println("-----------------------------");
         System.out.println("Pencarian data");
         System.out.println("-----------------------------");
         System.out.println("Masukkan IPK mahasiswa yang dicari :");
         System.out.print("IPK\t :");
         String cariStr2 = sc.nextLine().replace(",", ".");
         double cari2 = Double.parseDouble(cariStr2);
         System.out.println("-----------------------------");
         System.out.println(" menggunakan binary search ");
         System.out.println("-----------------------------");
         double posisi2 = list.findBinarySearch(cari2, 0,jumMhs-1);
         int pss2 = (int) posisi2;
         list.tampilPosisi(cari2, pss2);
         list.tampilDataSearch(cari2, pss2);
```

3. Jalankan dan amati hasilnya (inputkan data IPK secara terurut -ASC seperti verifikasi hasil percobaan dibawah ini).

```
Masukkan Data Mahasiswa Ke -1 :
NIM   : 111
Nama  : Edi
Kelas : 2
IPK   : 3.1
-----------------------
Masukkan Data Mahasiswa Ke -2 :
NIM   : 222
Nama  : Eli
Kelas : 2
IPK   : 3.2
-----------------------
Masukkan Data Mahasiswa Ke -3 :
NIM   : 333
Nama  : susi
Kelas : 2
IPK   : 3.3
-----------------------
Masukkan Data Mahasiswa Ke -4 :
NIM   : 444
Nama  : susan
Kelas : 2
IPK   : 3.5
-----------------------
Masukkan Data Mahasiswa Ke -5 :
NIM   : 555
Nama  : dimas
Kelas : 2
IPK   : 3.7
-----------------------
```

```
----------------------------
 menggunakan binary search
----------------------------
data mahasiswa dengan IPK :3.2 ditemukan pada indeks 1
nim      : 222
nama     : Eli
Kelas    : 2
IPK      : 3.2
```

Pertanyaan

1. Proses divide terjadi pada methode findBinarySearch dalam class mahasiswaBerprestasi :
   ```
   mid = (left+right)/2;
   ```
   pencarian dilakukan dengan membagi array menjadi dua bagian berdasarkan nilai tengah (mid).
2. Proses conquer terjadi dalam methode findBinarySearch, pada kode program bagian:
   ```
   if (cari ==listMhs[mid].ipk) {
       return (mid);
   }
   else if (listMhs[mid].ipk>cari) {
       return findBinarySearch(cari, left, mid-1);
   }
   else{
       return findBinarySearch(cari, mid+1, right);
   }
   ```
   Setelah pembagian, algoritma menentukan ke bagian mana harus mencari lebih lanjut, merupakan bagian conquer, di mana program menyelesaikan pencarian secara rekursif.
3. Jika data IPK yang dimasukkan tidak urut :
   a. Sequential search (sequentialSearching) tetap bisa berjalan karena tidak memerlukan data yang terurut.
   b. Binary search (findBinarySearch) tidak akan berfungsi dengan benar jika data tidak diurutkan karena algoritma ini mengandalkan pembagian array yang terurut.
   c. Agar dapat berjalan maka harus dilakukan pengurutan data terlebih dahulu sebelum melakukan binary search menggunakan shorting.
4. Jika kita mencari IPK 3.2 dalam daftar {3.8, 3.7, 3.5, 3.4, 3.2}, maka Binary Search tidak akan berfungsi dengan benar karena algoritma default mengasumsikan

bahwa data diurutkan secara ascending.

Solusi mengubah kondisi :

```
else if (listMhs[mid].ipk<cari) {
```

5. Code yang dimodifikasi
    a. Pada class mahasiswaBerprestasi

```
mahasiswa20 [] listMhs;

    int idx;

    public mahasiswaBerprestasi(int banyakMhs){

        listMhs = new mahasiswa20[banyakMhs];

        idx=0;

    }
```

    b. Pada class mahasiswaDemo20

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

        Scanner sc = new Scanner(System.in);

        System.out.print("Masukkan Jumlah data yang ingin dimasukkan :");

        int jumMhs = sc.nextInt();

        sc.nextLine();

        mahasiswaBerprestasi list = new mahasiswaBerprestasi(jumMhs);
```

LATIHAN PRAKTIKUM

1. Pada Latihan praktikum pertemuan sebelumnya pada Jobsheet 6 yang terdapat 3 class yaitu Dosen, DataDosen , dan DosenDemo, tambahkan method:
    a. PencarianDataSequential : digunakan untuk mencari data dosen berdasarkan nama dengan algoritma sequential search.
    b. PencarianDataBinary : digunakan untuk mencari data dosen berdasarkan usia dengan algoritma Binary Search.
    c. Buat aturan untuk mendeteksi hasil pencarian lebih dari 1 hasil dalam bentuk kalimat peringatan! Pastikan algoritma yang diterapkan sesuai dengan kasus yang diberikan!

JAWABAN

1. Source Code
    a. Class Dosen

```
package Praktikum_ASD.ALSD_Jobsheet7;
public class dosen {
    String kode;
    String nama;
    boolean jenisKelamin;
    int usia;
    dosen(String kd, String name, boolean jk, int age){
        kode = kd;
        nama = name;
        jenisKelamin = jk;
        usia = age;
    }
    void tampil(){
        System.out.println("Kode            :"+ kode);
        System.out.println("Nama            :"+ nama);
        System.out.println("Jenis Kelamin :"+(jenisKelamin ? "Laki- Laki" :
"Perempuan"));
        System.out.println("Usia            :"+usia);
    }
}
```

b. Class dataDosen

```
package Praktikum_ASD.ALSD_Jobsheet7;
public class dataDosen {
    dosen[] dataDosen = new dosen[10];
    int idx = 0;
    void tambah(dosen dsn){
        if (idx < dataDosen.length) {
            dataDosen[idx]=dsn;
            idx++;
        }else {
            System.out.println("data sudah penuh");
        }
    }
    void tampil(){
        if (idx == 0) {
            System.out.println("Belum ada data");
        } else {
            for (int i = 0; i < idx; i++) {
                dataDosen[i].tampil();
            }
        }
    }
```

```
void sortingASC(){
        for (int i = 0; i < idx - 1; i++) {
            for (int j = 0; j < idx - i - 1; j++) {
                if (dataDosen[j].usia > dataDosen[j + 1].usia) {
                    dosen temp = dataDosen[j];
                    dataDosen[j] = dataDosen[j + 1];
                    dataDosen[j + 1] = temp;
                }
            }
        }
    }
    void sortingDSC(){
        for (int i = 0; i < idx - 1; i++) {
            int idxMax = i;
            for (int j = i + 1; j < idx; j++) {
                if (dataDosen[j].usia > dataDosen[idxMax].usia) {
                    idxMax = j;
                }
            }
            dosen tmp = dataDosen[idxMax];
            dataDosen[idxMax] = dataDosen[i];
            dataDosen[i] = tmp;
        }
}
    void insertionSort(){
        for (int i = 1; i < idx; i++) {
            dosen temp = dataDosen[i];
            int j = i;
            while (j > 0 && dataDosen[j - 1].usia < temp.usia) {
                dataDosen[j] = dataDosen[j - 1];
                j--;
            }
            dataDosen[j] = temp;
        }
    }
```

```
void sequentialSearching(String cari){
        int posisi = 0;
        for (int i = 0; i < idx; i++) {
            if (dataDosen[i].nama.equalsIgnoreCase(cari)) {
                dataDosen[i].tampil();
                posisi++;
            }
        }
        if (posisi > 1) {
            System.out.println("Hasil pencarian ditemukan lebih dari satu,
Silahkan masukkan data dengan benar");
        }else if (posisi == 0) {
            System.out.println("Data tidak ditemukan");
        }
    }
    void cariBinarySearch(int usia) {
        sortingASC();
        int left = 0, right = idx - 1;
        int count = 0;
        while (left <= right) {
            int mid = left + (right - left) / 2;
            if (dataDosen[mid].usia == usia) {
                int tempMid = mid;
                while (tempMid >= 0 && dataDosen[tempMid].usia == usia) {
                    dataDosen[tempMid].tampil();
                    count++;
                    tempMid--;
                }
                tempMid = mid + 1;
                while (tempMid < idx && dataDosen[tempMid].usia == usia) {
                    dataDosen[tempMid].tampil();
                    count++;
                    tempMid++;
                }
                break;
            } else if (dataDosen[mid].usia < usia) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
```

```
        if (count > 1) {
            System.out.println("Hasil pencarian ditemukan lebih dari satu
kecocokan. Silakan masukkan data dengan tepat!");
        } else if (count == 0) {
            System.out.println("Data tidak ditemukan.");
        }
    }
}
```

c. Class dosenMain

```
package Praktikum_ASD.ALSD_Jobsheet6;
import java.util.Scanner;
public class dosenMain {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        dataDosen listDosen = new dataDosen();

        while (true) {
            System.out.println("-- Sorting Menu --");
            System.out.println("1. Tambahkan Data Dosen");
            System.out.println("2. Tampilkan Data Dosen");
            System.out.println("3. Sorting menggunakan Bubble Sort(ASC)");
            System.out.println("4. Sorting menggunakan Selection Sort(DSC)");
            System.out.println("5. Sorting menggunakan Insertion Sort(DSC)");
            System.out.println("6. Keluar");
            System.out.print("Pilih Salah Satu :");
            int pilihan = sc.nextInt();

            switch (pilihan) {
                case 1:
                   System.out.print("Masukkan total data yang mau ditambahkan :");
                   int data = sc.nextInt();
                   sc.nextLine();
                   for (int i = 0; i < data; i++) {
                        System.out.print("\nKode              : ");
                        String kode = sc.nextLine();

                        System.out.print("Nama              : ");
                        String nama = sc.nextLine();
```

```
 System.out.print("Jenis Kelamin (L/P): ");
                        char gender = sc.next().charAt(0);
                        boolean jk = (gender == 'L' || gender == 'l');

                        System.out.print("Usia              : ");
                        int usia = sc.nextInt();
                        sc.nextLine();

                        dosen dsn = new dosen(kode, nama, jk, usia);
                        listDosen.tambah(dsn);
                    }
                    break;
                case 2 :
                    System.out.println("\n--- Data Keseluruhan ---");
                    listDosen.tampil();
                    break;
                case 3:
                  System.out.println("\n Data telah diSorting (ASC - Bubble Sort)");
                    listDosen.sortingASC();
                    listDosen.tampil();
                    break;
                case 4:
             System.out.println("\n Data telah diSorting (DSC - Selection Sort)");
                    listDosen.sortingDSC();
                    listDosen.tampil();
                    break;
                case 5:
            System.out.println("\n Data telah diSorting (DSC - Insertion Sort)");
                    listDosen.insertionSort();
                    listDosen.tampil();
                    break;
                case 6:
                System.out.println("Terima Kasih");
                    break;
                default:
                    System.out.println("Masukkan data dengan benar");
                    break;
            }
            if (pilihan == 6) {
                break;
            }
        }
    }
}
```

2. Compile & Run program

```
-- Sorting Menu --
1. Tambahkan Data Dosen
2. Tampilkan Data Dosen
3. Sorting menggunakan Bubble Sort(ASC)
4. Sorting menggunakan Selection Sort(DSC)
5. Sorting menggunakan Insertion Sort(DSC)
6. Searching menggunakan Nama
7. Searching menggunakan Usia
8. Keluar
Pilih Salah Satu :1
Masukkan total data yang mau ditambahkan :2

Kode              : 111
Nama              : Edi
Jenis Kelamin (L/P): L
Usia              : 29

Kode              : 222
Nama              : Eli
Jenis Kelamin (L/P): P
Usia              : 28
-- Sorting Menu --
1. Tambahkan Data Dosen
2. Tampilkan Data Dosen
3. Sorting menggunakan Bubble Sort(ASC)
4. Sorting menggunakan Selection Sort(DSC)
5. Sorting menggunakan Insertion Sort(DSC)
6. Searching menggunakan Nama
7. Searching menggunakan Usia
8. Keluar
Pilih Salah Satu :6
Masukkan nama dosen yang mau dicari :Edi
Kode            :111
Nama            :Edi
Jenis Kelamin :Laki- Laki
Usia            :29
-- Sorting Menu --
1. Tambahkan Data Dosen
2. Tampilkan Data Dosen
3. Sorting menggunakan Bubble Sort(ASC)
4. Sorting menggunakan Selection Sort(DSC)
5. Sorting menggunakan Insertion Sort(DSC)
6. Searching menggunakan Nama
7. Searching menggunakan Usia
8. Keluar
Pilih Salah Satu :7

Masukkan Usia yang mau dicari :27
Data tidak ditemukan.
```

```
-- Sorting Menu --
1. Tambahkan Data Dosen
2. Tampilkan Data Dosen
3. Sorting menggunakan Bubble Sort(ASC)
4. Sorting menggunakan Selection Sort(DSC)
5. Sorting menggunakan Insertion Sort(DSC)
6. Searching menggunakan Nama
7. Searching menggunakan Usia
8. Keluar
Pilih Salah Satu :7

Masukkan Usia yang mau dicari :28
Kode            :222
Nama            :Eli
Jenis Kelamin :Perempuan
Usia            :28
```