PRAKTIKUM 10
ALGORITMA PENGURUTAN (SELECTION SORT)

A. TUJUAN PEMBELAJARAN

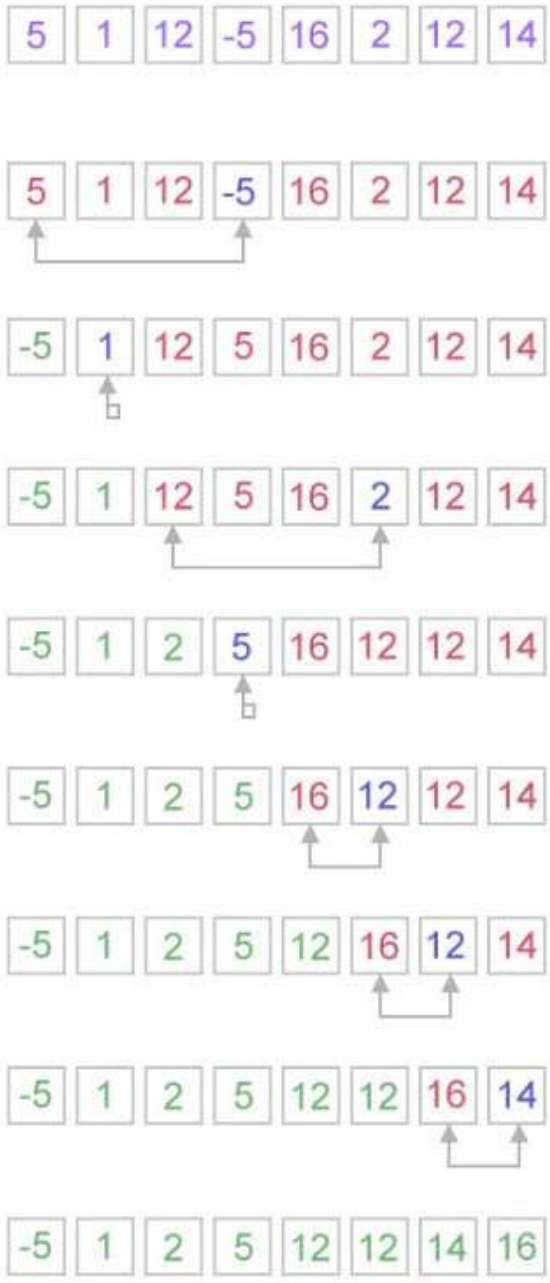
1. Memahami step by step algoritma pengurutan *selection sort.*
2. Mampu mengimplementasikan algoritma pengurutan *selection sort* dengan berbagai macam parameter berupa tipe data primitif atau tipe Generic.
3. Mampu mengimplementasikan algoritma pengurutan *selection sort* secara *ascending* dan *descending.*

B. DASAR TEORI

Algoritma *Selection Sort*

Ide utama adalah pada data indeks ke-0, dibandingkan dengan data sesudahnya untuk mencari elemen yang paling kecil, selanjutnya elemen terkecil tersebut ditukar dengan elemen pada indeks ke-0. Selanjutnya data indeks ke-1, dibandingkan dengan data sesudahnya untuk mencari elemen yang paling kecil, selanjutnya elemen terkecil tersebut ditukar dengan elemen pada indeks ke-1, dan seterusnya sampai data terurut secara ascending.

Contoh terdapat data 5, 1, 12, -5, 16, 2, 12, 14. Data acuan pada indek ke-0 yaitu 5 dibandingkan dengan data sesudahnya untuk mencari elemen terkecil. Elemen terkecil setelah 5 adalah -5, sehingga 5 ditukar dengan -5, sehingga data menjadi -5, 1, 12, 5, 16, 2, 12, 14. Data acuan berikutnya adalah indek ke-1 yaitu 1 dibandingkan dengan data sesudahnya untuk mencari elemen terkecil. Elemen terkecil setelah 1 ternyata adalah 1, sehingga 1 ditukar dengan 1, sehingga data menjadi -5, 1, 12, 5, 16, 2, 12, 14 dan seterusnya.


5 1 12 -5 16 2 12 14
5 1 12 -5 16 2 12 14
-5 1 12 5 16 2 12 14
-5 1 12 5 16 2 12 14
-5 1 2 5 16 12 12 14
-5 1 2 5 16 12 12 14
-5 1 2 5 12 16 12 14
-5 1 2 5 12 12 16 14
-5 1 2 5 12 12 14 16

21

C. TUGAS PENDAHULUAN

Jelaskan algoritma pengurutan *selection sort* secara *ascending* dengan data 5 6 3 1 2

D. PERCOBAAN

Percobaan 1 : *Selection sort* secara *ascending* dengan data int

```
public class SelectionDemo {
    public static void selectionSort(int[] arr) {
        // index of smallest element in the sublist
        int smallIndex;
        int pass, j, n = arr.length;
        int temp;

        for (pass = 0; pass < n - 1; pass++) {
            smallIndex = pass;
            for (j = pass + 1; j < n; j++)
            {
                if (arr[j]< arr[smallIndex]) {
                    smallIndex = j;
                }
            }
            // tukar nilai terkecil dengan arr[pass]
            temp = arr[pass];
            arr[pass] = arr[smallIndex];
            arr[smallIndex] = temp;
        }
    }

    public static void tampil(int data[]) {
        for (int i = 0; i < data.length; i++) {
            System.out.print(data[i] + " ");
        }
        System.out.println();
    }
}

public class MainSelection {
    public static void main(String[] args) {
        int A[] = {10,6,8,3,1};
        SelectionDemo.tampil(A);
        SelectionDemo.selectionSort(A);
        SelectionDemo.tampil(A);
    }
}
```

Percobaan 2 : *Selection sort* secara *descending* dengan data int

```
public class SelectionDemo {
    public static void selectionSort(int[] arr) {
```

```
        // index of smallest element in the sublist
        int smallIndex;
        int pass, j, n = arr.length;
        int temp;

        for (pass = 0; pass < n - 1; pass++) {
            smallIndex = pass;
            for (j = pass + 1; j < n; j++)
            {
                if (arr[j]> arr[smallIndex]) {
                    smallIndex = j;
                }
            }
            // tukar nilai terkecil dengan arr[pass]
            temp = arr[pass];
            arr[pass] = arr[smallIndex];
            arr[smallIndex] = temp;
        }
    }

    public static void tampil(int data[]) {
        for (int i = 0; i < data.length; i++) {
            System.out.print(data[i] + " ");
        }
        System.out.println();
    }
}

public class MainSelection {
    public static void main(String[] args) {
        int A[] = {10,6,8,3,1};
        SelectionDemo.tampil(A);
        SelectionDemo.selectionSort(A);
        SelectionDemo.tampil(A);
    }
}
```

Percobaan 3 : *Selection sort* secara *ascending* dengan data double

```
public static void selectionSort(double[] arr) {
        // index of smallest element in the sublist
        int smallIndex;
        int pass, j, n = arr.length;
        double temp;

        for (pass = 0; pass < n - 1; pass++) {
            smallIndex = pass;
            for (j = pass + 1; j < n; j++)
            {
                if (arr[j]> arr[smallIndex]) {
                    smallIndex = j;
                }
            }
```

```
            // tukar nilai terkecil dengan arr[pass]
            temp = arr[pass];
            arr[pass] = arr[smallIndex];
            arr[smallIndex] = temp;
        }
    }

    public static void tampil(double data[]) {
        for (int i = 0; i < data.length; i++) {
            System.out.print(data[i] + " ");
        }
        System.out.println();
    }

    public class MainSelection2 {
    public static void main(String[] args) {
        double A[] = {10.3,6.2,8.4,3.6,1.1};
        SelectionDemo.tampil(A);
        SelectionDemo.selectionSort(A);
        SelectionDemo.tampil(A);
    }
}
```

E. LATIHAN

1. Buatlah program sorting Selection dengan parameter array Integer (class Wrapper) !

```
public static void selectionSort(Integer[] A){…}
```

2. Buatlah program sorting Selection dengan parameter array Double (class Wrapper) !

```
public static void selectionSort(Double[] A){…}
```

3. Buatlah fungsi tampil() untuk menampilkan data.

```
public static<T> void tampil(T data[]){ }
```

4. Lakukan pengujian fungsi selectionSort(), dengan membuat fungsi main() sebagai berikut :

```
public class Demo1 {
    public static void main(String[] args) {
        //Data Integer
        Integer arr3[] = {1,5,6,2,8,9};
        SelectionDemo.selectionSort(arr3);
        SelectionDemo.tampil(arr3);

        //data Double
        Double arr4[] = {1.3,5.2,6.6,2.7,8.8,9.1};
        SelectionDemo.selectionSort(arr4);
        SelectionDemo.tampil(arr4);
```

```
    }
}
```

5. Buatlah program sorting Selection dengan parameter array Number !

```
public static<T extends Number> void selectionSort(T[] A){…}
```

6. Lakukan pengujian fungsi selectionSort(), dengan membuat fungsi main() sebagai berikut :

```
public class Demo2 {
    public static void main(String[] args) {
        Float arr5[] = {1.3f,5.2f,6.6f,2.7f,8.8f,9.1f};
        SelectionDemo.selectionSort (arr5);
        SelectionDemo.tampil(arr5);

        Byte arr6[] = {6,7,11,1,3,2};
        SelectionDemo.selectionSort (arr6);
        SelectionDemo.tampil(arr6);
    }
}
```

7. Buatlah program sorting Selection dengan parameter array yang memenuhi T extends Comparable !

```
public static <T extends Comparable> void selectionSort2(T[]
arr) { }
```

8. Lakukan pengujian fungsi selectionSort(), dengan membuat fungsi main() sebagai berikut :

```
public class Demo3 {
    public static void main(String[] args) {
        //data String
        String arr7[]=
{"jeruk","anggur","belimbing","jambu","kelengkeng"};
        SelectionDemo.selectionSort2(arr7);
        SelectionDemo.tampil(arr7);
    }
}
```

9. 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.

```
public class Mahasiswa implements Comparable <Mahasiswa> {
    private String nrp ;
    private String nama ;
```

```
    @Override
    public int compareTo(Mahasiswa o) {…}

    @Override
    public String toString() {…}
}
```

10. Lakukan pengujian fungsi selectionSort() lagi, sebelumnya tambahkan pada fungsi main() seperti di bawah ini !

```
public class Demo4 {
    public static void main(String[] args) {
        Mahasiswa arr8[] = {new Mahasiswa("02", "Budi"), new
Mahasiswa("01", "Andi"), new Mahasiswa("04", "Udin"), new
Mahasiswa("03", "Candra")};
        SelectionDemo.selectionSort2(arr8);
        SelectionDemo.tampil(arr8);
    }
}
```

F. LAPORAN RESMI

Kerjakan hasil percobaan(D) dan latihan(E) di atas dan tambahkan analisa.