JOBSHEET 7

Searching

Name
Sherly Lutfi Azkiah Sulistyawati

NIM
2341720241

Class
1I

Major
Information Technology

Study Program
D4 Informatics Engineering

Practicum 1: Sequential Search Method

Practice > Week7 > Students.java > ...

```java
package Week7;

public class Students {
    int nim, age;
    String name;
    double gpa;

    public Students(int nim, String name, int age, double gpa) {
        this.nim = nim;
        this.name = name;
        this.age = age;
        this.gpa = gpa;
    }

    public void display() {
        System.out.println("NIM : " + nim);
        System.out.println("Name: " + name);
        System.out.println("Age : " + age);
        System.out.println("GPA : " + gpa);
    }
}
```

Practice > Week7 > SearchStudent.java > SearchStudent > add(Students)

```java
package Week7;

public class SearchStudent {
    Students[] listStd = new Students[5];
    int idx;

    public void add(Students std) {
        if (idx < listStd.length) {
            listStd[idx] = std;
            idx++;
        } else {
            System.out.println(x:"Data is already full");
        }
    }

    public void display() {
        for (Students students : listStd) {
            students.display();
            System.out.println(x:"----------------------------");
        }
    }

    public int findSeqSearch(int search) {
        int position = -1;
        for (int i = 0; i < listStd.length; i++) {
            if (listStd[i].nim == search) {
                position = i;
                break;
            }
        }
        return position;
    }

    public void showPosition(int x, int pos) {
        if (pos != -1) {
            System.out.println("Data : " + x + " is found in index-" + pos);
        } else {
            System.out.println("Data : " + x + " is not found");
        }
    }

    public void showData(int x, int pos) {
        if (pos != -1) {
            System.out.println("NIM \t : " + x);
            System.out.println("Name \t : " + listStd[pos].name);
            System.out.println("Age \t : " + listStd[pos].age);
            System.out.println("GPA \t : " + listStd[pos].gpa);
        } else {
            System.out.println("Data " + x + " is not found");
        }
    }
}
```

Practice > Week7 > MainStudent.java > MainStudent > main(String[])

```java
package Week7;

import java.util.Scanner;

public class MainStudent {
    Run | Debug
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        Scanner sl = new Scanner(System.in);

        SearchStudent data = new SearchStudent();
        int amountStudent = 5;

        System.out.println("-----------------------------------------------");
        System.out.println("Input student data accordingly from smallest NIM");
        for (int i = 0; i < amountStudent; i++) {
            System.out.println("----------------------------");
            System.out.print("NIM\t: ");
            int nim = s.nextInt();
            System.out.print("Name\t: ");
            String name = sl.nextLine();
            System.out.print("Age\t: ");
            int age = s.nextInt();
            System.out.print("GPA\t: ");
            double gpa = s.nextDouble();

            Students std = new Students(nim, name, age, gpa);
            data.add(std);
        }

        System.out.println("-----------------------------------------------");
        System.out.println("Entire Student Data");
        data.display();

        System.out.println("_____________________________");
        System.out.println("_____________________________");
        System.out.print("Search student by NIM: ");
        int search = s.nextInt();
        System.out.println("Using Sequential Search");
        int position = data.findSeqSearch(search);
        data.showPosition(search, position);
        data.showData(search, position);

        s.close();
```

```
-----------------------------------------------
Input student data accordingly from smallest NIM
----------------------------
NIM     : 2017
Name    : Dewi Lestari
Age     : 23
GPA     : 3.5
----------------------------
NIM     : 2018
Name    : Sinta Sanjaya
Age     : 22
GPA     : 4
----------------------------
NIM     : 2019
Name    : Danang Adi
Age     : 22
GPA     : 3.7
----------------------------
NIM     : 2020
Name    : Budi Prakarsa
Age     : 20
GPA     : 2.9
----------------------------
NIM     : 2021
Name    : Vania Siti
Age     : 20
GPA     : 3.0
```

```
-----------------------------------------------
Entire Student Data
NIM : 2017
Name : Dewi Lestari
Age : 23
GPA : 3.5
----------------------------
NIM : 2018
Name : Sinta Sanjaya
Age : 22
GPA : 4.0
----------------------------
NIM : 2019
Name : Danang Adi
Age : 22
GPA : 3.7
----------------------------
NIM : 2020
Name : Budi Prakarsa
Age : 20
GPA : 2.9
----------------------------
NIM : 2021
Name : Vania Siti
Age : 20
GPA : 3.0
----------------------------

_____________________________
_____________________________
Search student by NIM: 2018
Using Sequential Search
Data : 2018 is found in index-1
NIM     : 2018
Name    : Sinta Sanjaya
Age     : 22
GPA     : 4.0
```

Question

1. What is the difference of method **displayData** and **displayPosition** in **StudentSearch** class?

   - displayData: This method is responsible for displaying all the student data stored in the listStd array. It iterates through the array and prints the details of each student, including NIM, name, age, and GPA.

   - displayPosition: This method is used to display the position of a student by NIM. It takes the NIM of a student as input and displays the index where that student is located in the listStd array.

2. What is the function of break in this following program code?

```
if(listStd[i].nim == search){
    position = i;
    break;
}
```

   - break statement is used to exit the loop as soon as the desired student with the specified NIM is found. Once the student is found, there is no need to continue searching, so the loop terminates immediately using break.

3. If inserted NIM data is not sorted from smallest to biggest value, will the program encounter an error? Is the result still correct? Why is that?

   - If the inserted NIM data is not sorted from smallest to biggest value and the program uses binary search to find a specific NIM, the program will not encounter an error, but the results may not be correct. Binary search requires the data to be sorted in ascending order for it to work correctly.

Practicum 2: Binary Search Method

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

```
System.out.println(x:"_____");
System.out.println(x:"_____");
System.out.print(s:"Search student by NIM: ");
int search1 = s.nextInt();
System.out.println(x:"Using Binary Search");
int position1 = data.FindBinarySearch(search, left:0, amountStudent-1);
data.showPosition(search1, position1);
data.showData(search1, position1);
```

```
_____
_____
Search student by NIM: 2018
Using Binary Search
Data : 2018 is found in index-1
NIM      : 2018
Name     : Sinta Sanjaya
Age      : 22
GPA      : 4.0
```

Question

1. Show the program code in which runs the divide process

```
mid = (left + right) / 2;
```

2. Show the program code in which runs the conquer process

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

3. If inserted NIM data is not sorted, will the program crash? Why?

   If inserted NIM data is sorted from largest to smallest value (e.g 20215, 20214 20212, 20211,20210) and element being searched is 20210. How is the result of binary search? Does it return the correct one? if not, then change the code so that the binary search executed properly

   - If the inserted NIM data is not sorted, the program will not crash, but the binary search algorithm will not produce correct results. This is because binary search relies on the data being sorted in ascending order for its efficiency and correctness.

   If the NIM data is sorted from largest to smallest value, the binary search algorithm may not return the correct result. This is because binary search expects the data to be sorted in ascending order, but in this case, the data is sorted in descending order.

4. Modify program above so that the students amount inserted is matched with user inpu

```java
public class SearchStudent {
    Students[] listStd;
    int idx;

    public SearchStudent(int amountStudent) {
        this.listStd = new Students[amountStudent];
        this.idx = 0;
    }
```

```java
public class MainStudent {
    Run | Debug
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        Scanner sl = new Scanner(System.in);

        System.out.print(s:"Enter the number of students: ");
        int amountStudent = s.nextInt();

        SearchStudent data = new SearchStudent(amountStudent);
```

```
Enter the number of students: 3
-----------------------------------------------
Input student data accordingly from smallest NIM
----------------------------
NIM     : 2017
Name    : Dewi
Age     : 23
GPA     : 3.5
----------------------------
NIM     : 2018
Name    : Sinta
Age     : 22
GPA     : 4
----------------------------
NIM     : 2019
Name    : Danang
Age     : 22
GPA     : 3.7
-----------------------------------------------
Entire Student Data
NIM : 2017
Name: Dewi
Age : 23
GPA : 3.5
----------------------------
NIM : 2018
Name: Sinta
Age : 22
GPA : 4.0
----------------------------
NIM : 2019
Name: Danang
Age : 22
GPA : 3.7
----------------------------

----------------------------
----------------------------
Search student by NIM:
```

Practicum 3: Review Divide and Conquer

Practice > Week7 > MergeSort.java > MergeSort > merge(int[], int, int, int)

```java
package Week7;

public class MergeSort {
    public void mergeSort(int[] data) {

    }

    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;

        //compare every single part
        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];
        }
    }

    //Divide into 2 parts and divide it again until no more thing to be divided
    public void sort(int data[], int left, int right) {
        if (left < right) {
            int mid = (left + right) / 2;
            sort(data, left, mid);
            sort(data, mid+1, right);
            merge(data, left, mid, right);
        }
    }

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

```

Practice > Week7 > SortMain.java > SortMain > main(String[])

```java
package Week7;

public class SortMain {
    Run | Debug
    public static void main(String[] args) {
        //Create an object
        MergeSort mergeSort = new MergeSort();
        int[] data = {10, 40, 30, 50, 70, 20, 100, 90};

        System.out.println(x:"Sorting with merge sort");
        System.out.println(x:"Initial Data");
        mergeSort.printArray(data);

        //Call mergeSort method to sort the array
        mergeSort.sort(data, left:0, data.length - 1);
        System.out.println(x:"Sorted Data");
        mergeSort.printArray(data);
    }
}
```

Assignment

1. Modify the searching program above with these requirements:

   a. Before we search using binary search, we have to sort the data first. You can use whichever sorting algorithm that you are comfortable with

```java
public void bubbleSort() {
    int n = listStd.length;
    for (int i = 0; i < n-1; i++) {
        for (int j = 0; j < n-i-1; j++) {
            if (listStd[j].nim > listStd[j+1].nim) {
                // Swap listStd[j] and listStd[j+1]
                Students temp = listStd[j];
                listStd[j] = listStd[j+1];
                listStd[j+1] = temp;
            }
        }
    }
}
```

```java
// Sorting data before performing binary search
System.out.println(x:"_____");
System.out.println(x:"Sorting Data");
data.bubbleSort();
data.display();
```

```
----------------------------
----------------------------
Search student by NIM: 2018
Using Sequential Search
Data : 2018 is found in index-1
NIM      : 2018
Name     : Sinta Sanjaya
Age      : 22
GPA      : 4.0

----------------------------
Sorting Data
NIM : 2017
Name: Dewi Lestari
Age : 23
GPA : 3.5
----------------------------
NIM : 2018
Name: Sinta Sanjaya
Age : 22
GPA : 4.0
----------------------------
NIM : 2019
Name: Danang Adi
Age : 22
GPA : 3.7
----------------------------
NIM : 2020
Name: Budi Prakarsa
Age : 20
GPA : 2.9
----------------------------
NIM : 2021
Name: Vania Siti
Age : 20
GPA : 3.0
----------------------------

----------------------------
----------------------------
Search student by NIM: 2018
Using Binary Search
Data : 2018 is found in index-1
NIM      : 2018
Name     : Sinta Sanjaya
Age      : 22
GPA      : 4.0
PS D:\College\Semester 2\AlgoritmadanStrukturData>
```

2. Modify the searching above with these requirements:

   - Search by student's name with Sequential Search algorithm

```java
public int findByNameSeqSearch(String name) {
    for (int i = 0; i < listStd.length; i++) {
        if (listStd[i].name.equalsIgnoreCase(name)) {
            return i;
        }
    }
    return -1;
}

// In SearchStudent class
public void showPositionByName(String name, int pos) {
    if (pos != -1) {
        System.out.println("Data for student with name '" + name + "' is found in index-" + pos);
    } else {
        System.out.println("Student with name '" + name + "' is not found");
    }
}

public void showData(String name, int pos) {
    if (pos != -1) {
        System.out.println("Name \t : " + name);
        System.out.println("NIM \t : " + listStd[pos].nim);
        System.out.println("Age \t : " + listStd[pos].age);
        System.out.println("GPA \t : " + listStd[pos].gpa);
    } else {
        System.out.println("Student with name '" + name + "' is not found");
    }
}
```

```java
System.out.println(x:"_____");
System.out.println(x:"_____");
System.out.print(s:"Search student by Name: ");
String searchName = sl.nextLine();
System.out.println(x:"Using Sequential Search by Name");
int positionByName = data.findByNameSeqSearch(searchName);
data.showPositionByName(searchName, positionByName);
data.showData(searchName, positionByName);
```

```
_____
_____
Search student by Name: Sinta Sanjaya
Using Sequential Search by Name
Data for student with name 'Sinta Sanjaya' is found in index-1
Name     : Sinta Sanjaya
NIM      : 2018
Age      : 22
GPA      : 4.0
```

   - How is the output of the program if there is any duplicate name?

     If there are duplicate names in the list of students, the program will display information for the first occurrence of the name found in the list.

3. There is 2d array as follows:

Index	0	1	2	3	4
0	45	78	7	200	80
1	90	1	17	100	50
2	21	2	40	18	65

Based on data above, create a program to search data in 2d array, which the data to be searched is defined by user input (using sequential search)

Practice > Week7 > J Main.java > ...

```java
package Week7;

import java.util.Scanner;

public class Main {
    Run | Debug
    public static void main(String[] args) {
        int[][] data = {
            {45, 78, 7, 200, 80},
            {90, 1, 17, 100, 50},
            {21, 2, 40, 18, 65}
        };

        Scanner scanner = new Scanner(System.in);

        displayData(data);

        System.out.print(s:"Enter the value to search: ");
        int searchValue = scanner.nextInt();

        int[] result = sequentialSearch2DArray(data, searchValue);

        if (result != null) {
            System.out.println("Value " + searchValue + " found at index (" + result[0] + ", " + result[1] + ")");
        } else {
            System.out.println("Value " + searchValue + " not found in the 2D array.");
        }
    }

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

    public static int[] sequentialSearch2DArray(int[][] data, int target) {
        int[] result = new int[2];

        for (int i = 0; i < data.length; i++) {
            for (int j = 0; j < data[i].length; j++) {
                if (data[i][j] == target) {
                    result[0] = i; // Row index
                    result[1] = j; // Column index
                    return result;
                }
            }
        }

        return null; // If target not found
    }
```

```
45      78      7       200     80
90      1       17      100     50
21      2       40      18      65
Enter the value to search: 100
Value 100 found at index (1, 3)
```

4. There is a 1D array as follows:

5. 0	1	2	3	4	5	6	7	8	9
12	17	2	1	70	50	90	17	2	90

Create a program to sort the array, search & display the biggest value, and print the amount of biggest value available alongside with its position.

Practice > Week7 > Array1D.java > ...

```java
package Week7;

import java.util.Arrays;

public class Array1D {
    Run | Debug
    public static void main(String[] args) {
        int[] array = {12, 17, 2, 1, 70, 50, 90, 17, 2, 90};

        System.out.println(x:"Unsorted Array:");
        for (int num : array) {
            System.out.print(num + " ");
        }
        System.out.println();

        Arrays.sort(array);

        System.out.println(x:"Sorted Array:");
        for (int num : array) {
            System.out.print(num + " ");
        }
        System.out.println();

        // Find and display the biggest value and its position
        int biggestValue = array[array.length - 1];
        System.out.println("Biggest Value: " + biggestValue);
        System.out.println(x:"Position(s):");
        for (int i = 0; i < array.length; i++) {
            if (array[i] == biggestValue) {
                System.out.println("Index " + i);
            }
        }

        // Count the amount of occurrences of the biggest value
        int count = 0;
        for (int num : array) {
            if (num == biggestValue) {
                count++;
            }
        }
        System.out.println("Amount of biggest value: " + count);
    }
}
```

```
Unsorted Array:
12 17 2 1 70 50 90 17 2 90
Sorted Array:
1 2 2 12 17 17 50 70 90 90
Biggest Value: 90
Position(s):
Index 8
Index 9
Amount of biggest value: 2
```