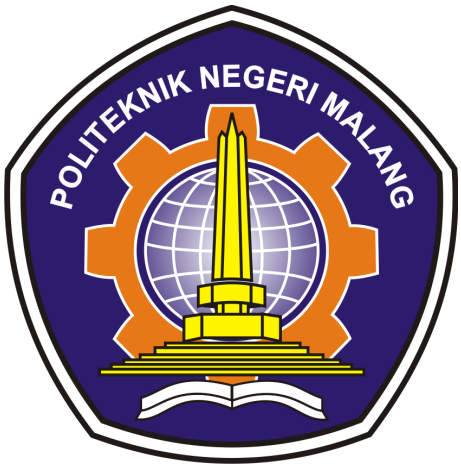
Data Structure and Algorithm Practicum
Sorting
Bubble, Selection, and Insertion Sort



Name
Muhammad Baihaqi Aulia Asy'ari

NIM
2241720145

Class
1I

Department
Information Technology

Study Program
D4 Informatics Engineering

1. Practicum Steps

a. Practicum 1 – Create Student Class

1. Pay attention in following class diagram. This will be used as our reference when creating Students Class.

Students
name: String entryYear: int age: int gpa: double
Students(n: String, t: int, u: int, i: double) displayInfo(): void

2. Create **Students** class as follows!

```
package Sorting;

public class Student {
    String name;
    int entranceYear, age;
    double gpa;

    public Student(String n, int y, int a, double g) {
        name = n;
        entranceYear = y;
        age = a;
        gpa = g;
    }

    void print() {
        System.out.println("Name          : "+name);
        System.out.println("Entrance Year:
        ↪  "+entranceYear);
        System.out.println("Age           : "+age);
        System.out.println("GPA           : "+gpa);
    }
}
```

b. Practicum 2 – Create HighAchieverStudent Class

1. In HighAchieverStudent Class, create a list and a method to sort the student's data based on their GPA. In addition, create a method to

display all those data and other method to insert the data to the list. Take a look at class diagram below!

HighAchieverStudent
list: Students[5] idx: int
add(std: Students): void display(): void bubbleSort(): void

2. Create **HighAchieverStudent** class as follows!

```
package Sorting;

public class HighAchieverStudent {
    Student[] list = new Student[5];
    int idx;
}
```

3. Add method add() in that class. This method will be used to add an object from **Students** class to listStd attribute.

```
void add(Student std) {
    if (idx<list.length) {
        list[idx] = std;
        idx++;
    } else {
        System.out.println("The student list is already
        ↪  full-filled");
    }
}
```

4. Add method display() in that class. This method will be used to display all the data that is exist in listStd. Take a look on how we use for loops, even though it is quite different than usual, the concept is still similar. Instead of accessing each element by its index, we just loop by each element available in the list.

```
void display() {
    for (Student student : list) {
        student.print();
        System.out.println("------------------------
        ↪  ------------------------");
    }
}
```

5. Add method bubbleSort() in that class.

```
void bubbleSort() {
    for (int i = 0; i < list.length-1; i++) {
        for (int j = 0; j < list.length-i-1; j++) {
            if (list[j].gpa > list[j-1].gpa) {
                Student tmp = list[j-1];
                list[j] = list[j-1];
                list[j-1] = tmp;
            }
        }
    }
}
```

6. Up to this point, the HighAchieverStudent class is finished

c. Practicum 3 – Create Main Class

1. Create main class and its main method

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

    }
}
```

2. In main method, create 2 objects from **Scanner** class, and an object from **HighAchieverStudent** class. After that, declare the variable of amountStd to 5!

```
Scanner s1 = new Scanner(System.in);
Scanner s2 = new Scanner(System.in);
HighAchieverStudent data = new HighAchieverStudent();
int n = 5;
```

3. Make loops 5 times with for, to insert name, age, entryYear, and GPA foreach students. Once it is done, instantiate the object from **Students** class and insert it to the list in **HighAchieverStudent** class.

```
for (int i = 0; i < n; i++) {
    System.out.print("Name         : ");
    String name = s2.nextLine();
    System.out.print("Entrance year: ");
    int year = s1.nextInt();
    System.out.print("Age          : ");
    int age = s1.nextInt();
    System.out.print("GPA          : ");
    int gpa = s1.nextInt();

    Student s = new Student(name, year, age, gpa);
```

```
        data.add(s);
    }
```

4. Display the student's data that has been inserted in the list!

```
System.out.println("Unsorted student list:");
data.display();
```

5. Call method bubbleSort() and show the result!

```
System.out.println("Student data after sorted in
↪  decending order in gpa: ");
data.bubbleSort();
data.display();
```

Try to execute the program and understand the result. Are the data in the list is sorted based on GPA?

d. Practicum 4 – Add Selection Sort process in HighAchieverStudent Class

1. Go to **HighAchieverStudent** class, add method selectionSort() there. This method will do the sorting process in ascending order, but with selection sort approach.

```
void selectionSort() {
    for (int i = 0; i < list.length-1; i++) {
        int idxMin = i;
        for (int j = i+1; j < list.length; j++) {
            if (list[j].gpa < list[idxMin].gpa) {
                idxMin = j;
            }
        }
        Student tmp = list[idxMin];
        list[idxMin] = list[i];
        list[i] = tmp;
    }
}
```

2. After that, open main class. In main method, add these line of code to execute selectionSort() method that we've just created. Try to execute the program and understand the result. Are the data in the list is

```
System.out.println("Ascending Sorted student list");
data.selectionSort();
data.display();
```

sorted based on GPA?

e. Practicum 5 – Add Insertion Sort process in HighAchieverStudent Class

1. Go to **HighAchieverStudent** class, add method insertionSort() there. This method will do the sorting process in ascending order, but with selection sort approach.

```
void insertionSort() {
    for (int i = 0; i < list.length-1; i++) {
        Student temp = list[i];
        int j = i;
        while (j > 0 && list[j-1].gpa > temp.gpa) {
            list[j] = list[j-1];
            j--;
        }
        list[j] = temp;
    }
}
```

2. After that, open main class. In main method, add these lines of code to execute insertionSort () method that we've just created.

```
System.out.println("Ascending Sorted student list");
data.insertionSort();
data.display();
```

Try to execute the program and understand the result. Are the data in the list is sorted based on GPA?

2. **Questions**

1. In which class we have a function to do sorting with bubble sort approach?

   Answer:
   HighAchieverStudent

2. In which class we have a function to do sorting with insertion sort approach?
   Answer:
   HighAchieverStudent

3. What is the meaning of swapping process? Write the code to do the swapping process in the program above!
   Answer:
   its a process where the program switch an element of an array with another element in the same array. usually use for sorting.

4. In bubbleSort(), there is these lines of code, what's the function of it?

```
if (list[j].gpa > list[j-1].gpa) {
    Student tmp = list[j-1];
    list[j] = list[j-1];
```

```
        list[j-1] = tmp;
    }
```

Answer:
if the previous element of the array is smaller than the current array, the program will swap the element.

5. Look at the loops inside the bubbleSort() method:

```
for (int i = 0; i < list.length-1; i++) {
    for (int j = 0; j < list.length-i-1; j++) {

    }
}
```

   a. What's the difference of loop i and loop j?
   Answer:
   the j loop iterate through the array to swap and carry. and the i loop iterate the starting index for the swap and carry process.
   b. Why is the criteria of loop i is i¡listStd.length-1?
   c.
   Answer:
   because the i doesnt need to iterate through the last element.
   d. Why is the criteria of loop j is j¡listStd.length-i?
   Answer:
   because the program doesnt need to swap through the whole array when the end part of the array is alredy sorted.
   e. If the data in listStd is 50, how many loop i will happen? And how many bubble sort steps will be?
   Answer:
   the i loop would need to loop through the same amount of data in the array, which is 50. and the swap and carry process would need to be processed in amount of the sum of series of number from 1 to the amount of data in the array.

6. In selection sort method, there is these lines of code, what's that for?

```
int idxMin = i;
for (int j = i+1; j < list.length; j++) {
    if (list[j].gpa < list[idxMin].gpa) {
        idxMin = j;
    }
}
```

Answer:

it loops through the unsorted part of the array and then saving the index of the minimum value in the array.

7. Change the insertionSort method so that the user has options to sort in either ascending or descending order. You can do it by adding a parameter, and this parameter's value will be assigned through function calling in main class

```
void insertionSort(boolean asc) {
    for (int i = 0; i < list.length-1; i++) {
        Student temp = list[i];
        int j = i;
        if(asc) {
            while (j > 0 && list[j-1].gpa > temp.gpa) {
                list[j] = list[j-1];
                j--;
            }
        } else {
            while (j > 0 && list[j-1].gpa < temp.gpa) {
                list[j] = list[j-1];
                j--;
            }
        }
        list[j] = temp;
    }
}
```

3. **Assignment**

1. There is a company that provide services in airplane ticket sales, they are developing a backend system for ticket reservation. One of its features is to display all available tickets based on filter from user. The ticket list must able to be sorted by the price in ascending and descending order. Implement these class diagrams in java program and create the sorting algorithm with **bubble sort** and **selection sort**

Tickets
airlines: String price: int destination: String origin : String
Ticket(String a, String dest, String origin, int price): void

TicketService
tickets : Ticket()
add(Ticket t) : void displayAll() : void bubbleSort() : void selectionSort() : void

MainTicket
Main(String[] abcd): void

```java
package Assignment1;

public class Tickets {
    String airlines, destination, origin;
    int price;

    public Tickets(String a, String dest, String origin, int
    ↪  price) {
        airlines = a;
        destination = dest;
        this.origin = origin;
        this.price = price;
    }
}
```

```java
package Assignment1;

public class TicketService {
    Tickets[] tickets = new Tickets[1];

    public void add(Tickets t) {
        if (tickets[0] == null) {
            tickets[0] = t;
        } else {
            Tickets[] temp = tickets;
            int newTicketsLen = tickets.length + 1;
            tickets = new Tickets[newTicketsLen];

            for (int i = 0; i < temp.length; i++) {
                tickets[i] = temp[i];
            }

            tickets[newTicketsLen-1] = t;
        }
    }

    public void displayAll() {
        for (int i = 0; i < tickets.length; i++) {
            System.out.println("------------------------
            ↪  ------------------------");
            System.out.printf("%s
            ↪  Airlines\n",tickets[i].airlines);
```

```
            System.out.printf("From %s to
            ↪  %s\n",tickets[i].origin,
            ↪  tickets[i].destination);
            System.out.printf("Cost: Rp
            ↪  %,d\n",tickets[i].price);
        }
    }

    void bubbleSort(boolean asc) {
        for (int i = 0; i < tickets.length-1; i++) {
            for (int j = 0; j < tickets.length-i-1; j++) {
                if (asc) {
                    if (tickets[j].price >
                    ↪  tickets[j-1].price) {
                        Tickets tmp = tickets[j-1];
                        tickets[j] = tickets[j-1];
                        tickets[j-1] = tmp;
                    }
                } else {
                    if (tickets[j].price <
                    ↪  tickets[j-1].price) {
                        Tickets tmp = tickets[j-1];
                        tickets[j] = tickets[j-1];
                        tickets[j-1] = tmp;
                    }
                }
            }
        }
    }

    void selectionSort(boolean asc) {
        for (int i = 0; i < tickets.length-1; i++) {
            int idxSelected = i;
            for (int j = i+1; j < tickets.length; j++) {
                if (asc) {
                    if (tickets[j].price <
                    ↪  tickets[idxSelected].price) {
                        idxSelected = j;
                    }
                } else {
                    if (tickets[j].price >
                    ↪  tickets[idxSelected].price) {
```

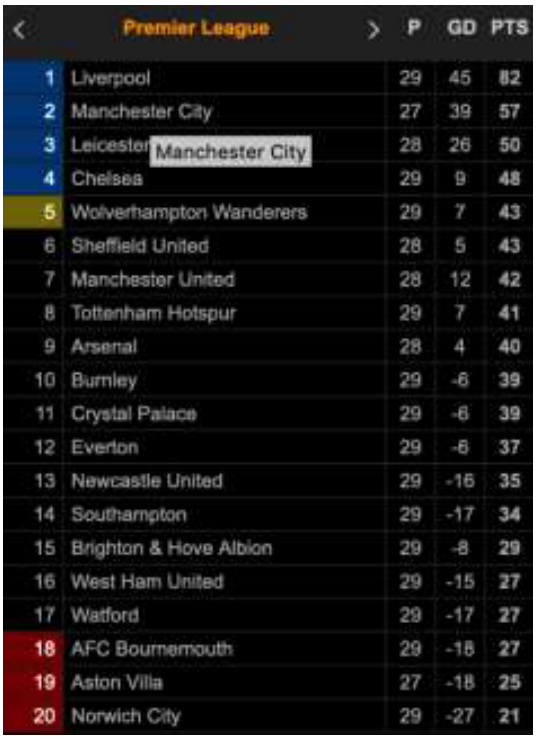
```
                        idxSelected = j;
                    }
                }
            }
            Tickets tmp = tickets[idxSelected];
            tickets[idxSelected] = tickets[i];
            tickets[i] = tmp;
        }
    }
}
```

```
package Assignment1;

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

    }
}
```

2. Premiere League in 2020 is already in half-season. In this season, Liverpool is the top of the list, the full list is displayed below



	Premier League	P	GD	PTS
1	Liverpool	29	45	82
2	Manchester City	27	39	57
3	Leicester	28	26	50
4	Chelsea	29	9	48
5	Wolverhampton Wanderers	29	7	43
6	Sheffield United	28	5	43
7	Manchester United	28	12	42
8	Tottenham Hotspur	29	7	41
9	Arsenal	28	4	40
10	Burnley	29	-6	39
11	Crystal Palace	29	-6	39
12	Everton	29	-6	37
13	Newcastle United	29	-16	35
14	Southampton	29	-17	34
15	Brighton & Hove Albion	29	-8	29
16	West Ham United	29	-15	27
17	Watford	29	-17	27
18	AFC Bournemouth	29	-18	27
19	Aston Villa	27	-18	25
20	Norwich City	29	-27	21

Change the standings list above to class diagram that has sorting club func-

tion based on highest to smallest points (in ascending order) with insertion sort algorithm. Take these following class diagrams as your reference:

`Answer:`