JOBSHEET 6

Sorting (Bubble, Selection, Insertion Sort)

Name
Sherly Lutfi Azkiah Sulistyawati

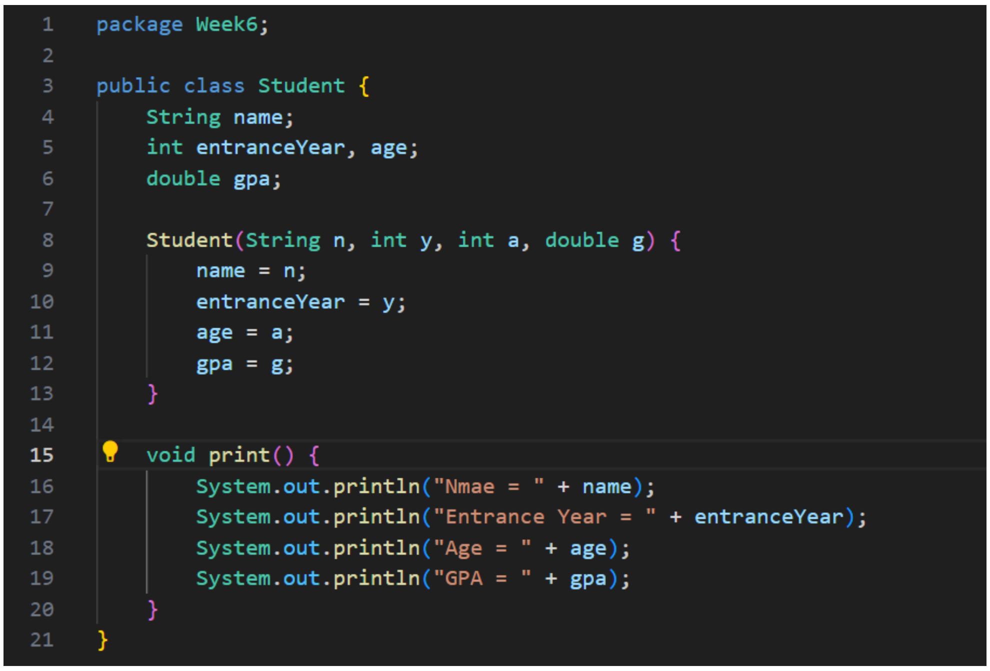
NIM
2341720241

Class
1I

Major
Information Technology

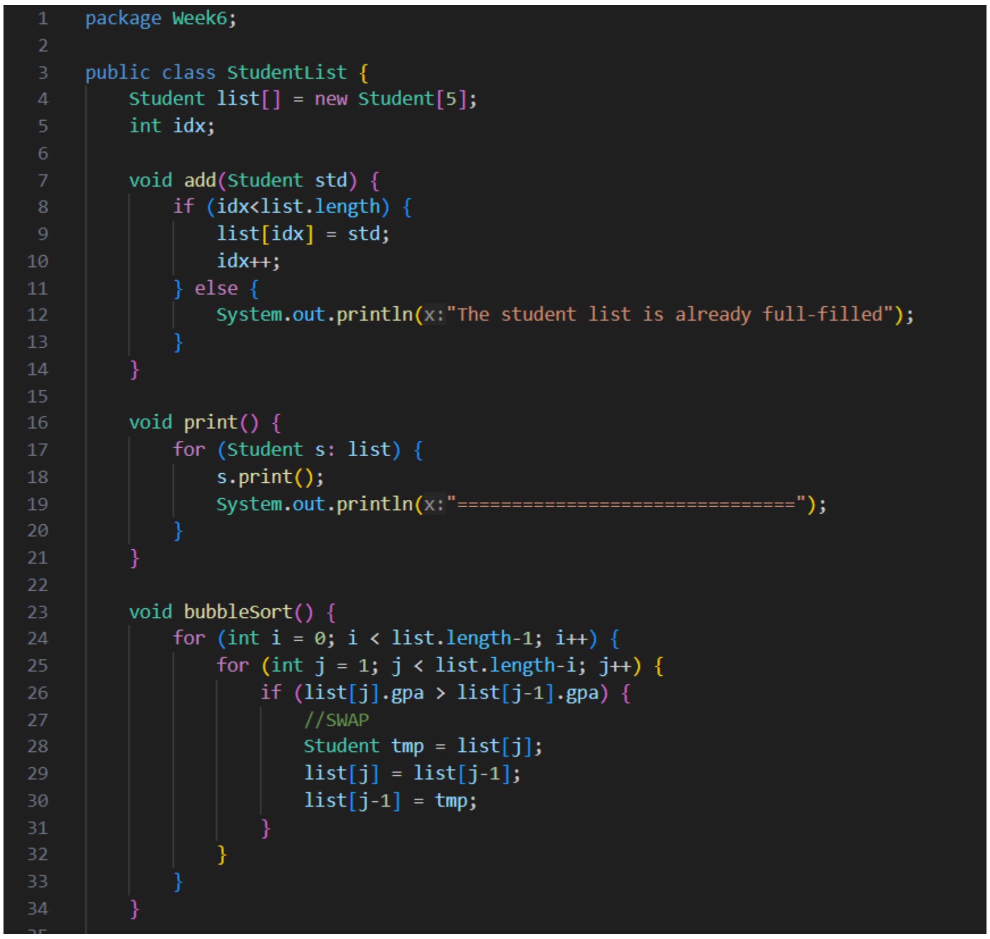
Study Program
D4 Informatics Engineering

Practicum 1: Create Student Class



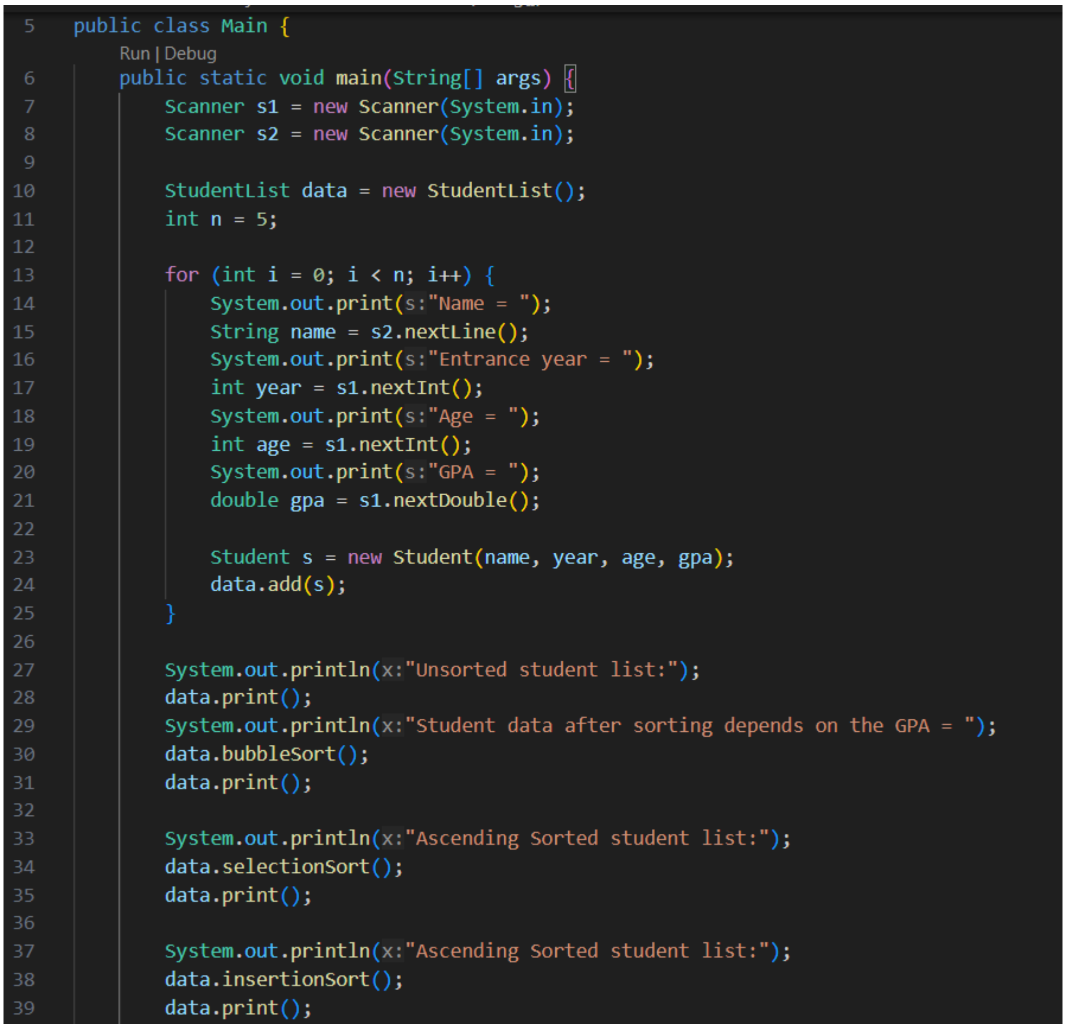
```java
package Week6;

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

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

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

Practicum 2: Create HighAchieverStudent Class



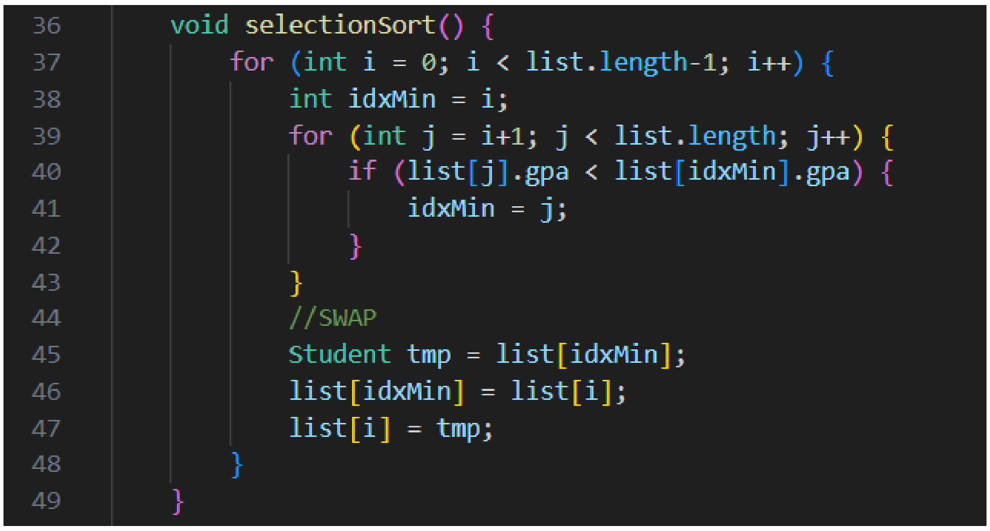
```java
package Week6;

public class StudentList {
    Student list[] = new Student[5];
    int idx;

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

    void print() {
        for (Student s: list) {
            s.print();
            System.out.println(x:"==============================");
        }
    }

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

Practicum 3: Create Main Class



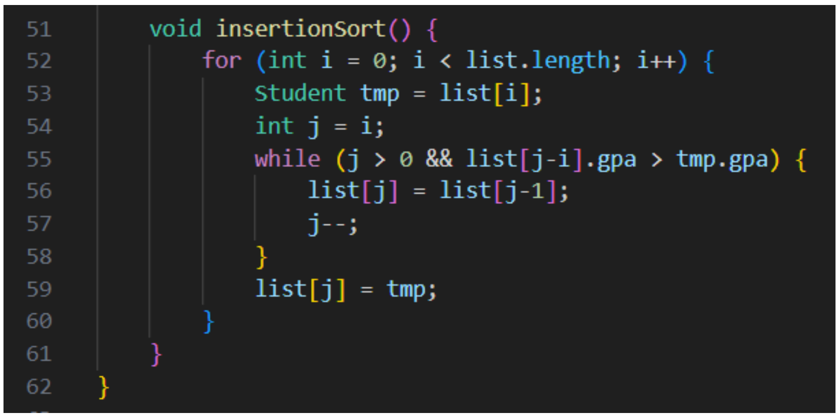
```
public class Main {
    Run | Debug
    public static void main(String[] args) {
        Scanner s1 = new Scanner(System.in);
        Scanner s2 = new Scanner(System.in);

        StudentList data = new StudentList();
        int n = 5;

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

            Student s = new Student(name, year, age, gpa);
            data.add(s);
        }

        System.out.println(x:"Unsorted student list:");
        data.print();
        System.out.println(x:"Student data after sorting depends on the GPA = ");
        data.bubbleSort();
        data.print();

        System.out.println(x:"Ascending Sorted student list:");
        data.selectionSort();
        data.print();

        System.out.println(x:"Ascending Sorted student list:");
        data.insertionSort();
        data.print();
```

Practicum 4: Add Selection Sort Process in HighAchieverStudent Class



```
    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;
                }
            }
            //SWAP
            Student tmp = list[idxMin];
            list[idxMin] = list[i];
            list[i] = tmp;
        }
    }
```

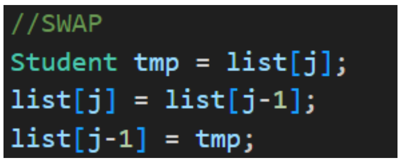
Practicum 4: Add Insertion Sort Process in HighAchieverStudent Class



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

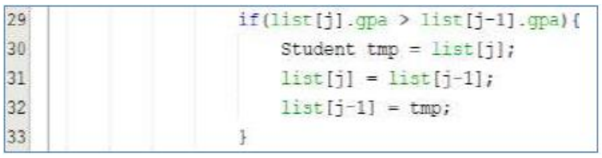
Question

1. In which class we have a function to do sorting with bubble sort approach?
   - The function to perform sorting with the bubble sort approach is located in the **HighAchieverStudent** class. The method is named **bubbleSort**.
2. In which class we have a function to do sorting with insertion sort approach?
   - The function to perform sorting with the insertion sort approach is also located in the **HighAchieverStudent** class. The method is named **insertionSort**.
3. What is the meaning of swapping process? Write the code to do the swapping process in the program above!
   - The swapping process involves exchanging the positions of two elements in an array. In the provided program, the swapping process is used to correctly order the elements during sorting. Here's the code to perform the swapping process:



```
//SWAP
Student tmp = list[j];
list[j] = list[j-1];
list[j-1] = tmp;
```

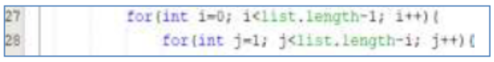
4. In bubbleSort(), there is these lines of code, what's the function of it?



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

   - In the bubbleSort method, the provided lines of code are responsible for comparing adjacent elements in the array and swapping them if they are in the wrong order. If the GPA of the current element (**list[j]**) is greater than the GPA of the previous element (**list[j-1]**), the two elements are swapped. This ensures that the higher GPA elements "bubble up" towards the end of the array.

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



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

   a. What's the difference of loop *i* and loop *j*?

      - The loop **i** iterates over each element of the array, starting from the first element and going up to the second-to-last element. It controls the number of passes through the array.

   b. Why is the criteria of loop *i* is i<listStd.length-1?

      - The condition **i < list.length - 1** ensures that the loop doesn't go beyond the last element of the array. Since each pass puts the largest unsorted element in its correct position, there is no need to compare the last element with any other elements.

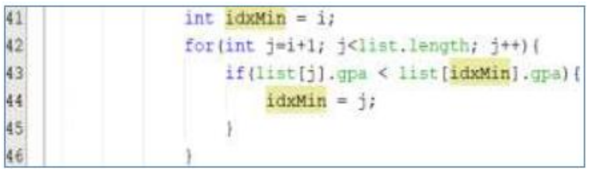
   c. Why is the criteria of loop *j* is j<listStd.length-i?

      - The loop **j** iterates over the unsorted portion of the array. It starts from index **1** (the second element) and goes up to **list.length - i**. This ensures that we only compare the unsorted portion of the array in each pass.

   d. If the data in listStd is 50, how many loop *i* will happen? And how many bubble sort steps will be?

      - If the data in **listStd** is 50, there will be 49 iterations of loop **i** and 49 bubble sort steps. This is because each iteration of loop **i** sorts one element to its correct position from the end.

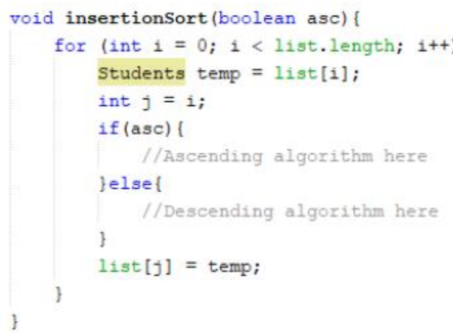
6. In selection sort method, there is these lines of code, what's that for?



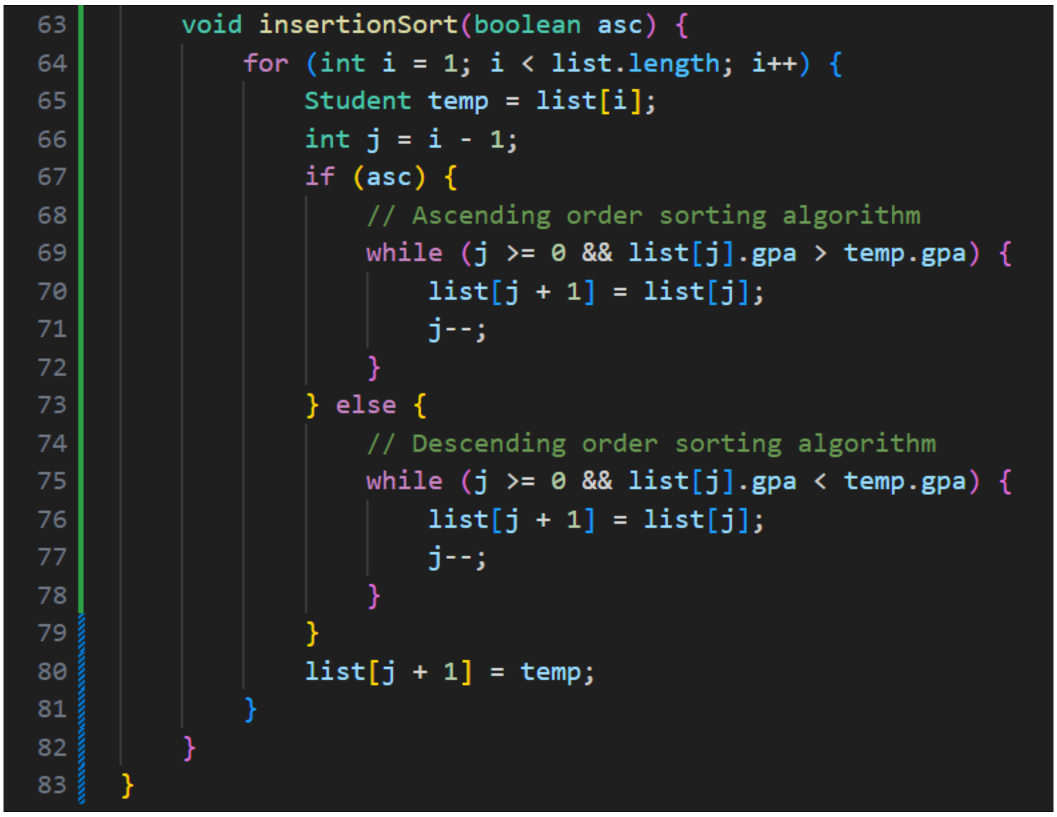
```
            int idxMin = i;
            for(int j=i+1; j<list.length; j++){
                if(list[j].gpa < list[idxMin].gpa){
                    idxMin = j;
                }
            }
```

   - In the selection sort method, the provided lines of code are responsible for finding the index of the minimum element in the unsorted portion of the array. This loop iterates over the unsorted portion of the array (starting from index **i+1**) and compares the GPA of each element with the GPA of the current minimum element (**list[idxMin]**). If a smaller GPA is found, the **idxMin** is updated to the index of the new minimum element.

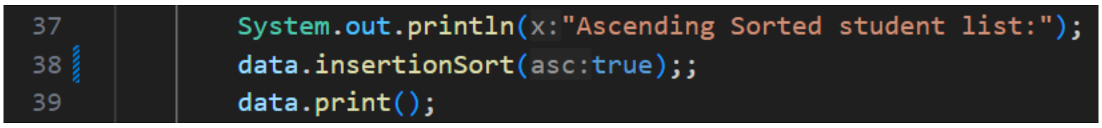
7. Change the insertionSort method so that the user has options to sort in either ascending or descending order. You can do it by adding a parameter, and this parameter's value will be assigned through function calling in main class



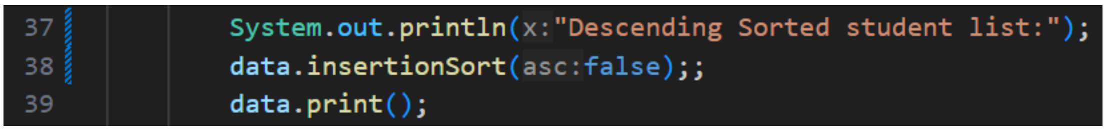
```
void insertionSort(boolean asc){
    for (int i = 0; i < list.length; i++) {
        Students temp = list[i];
        int j = i;
        if(asc){
            //Ascending algorithm here
        }else{
            //Descending algorithm here
        }
        list[j] = temp;
    }
}
```



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



```
        System.out.println(x:"Ascending Sorted student list:");
        data.insertionSort(asc:true);;
        data.print();
```



```
        System.out.println(x:"Descending Sorted student list:");
        data.insertionSort(asc:false);;
        data.print();
```

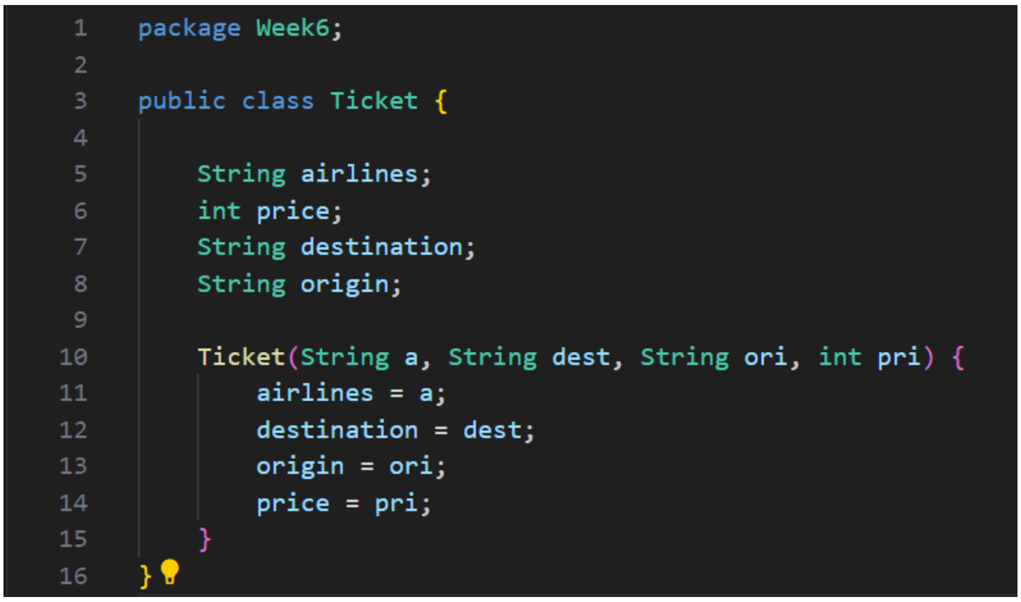
Assignment

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**

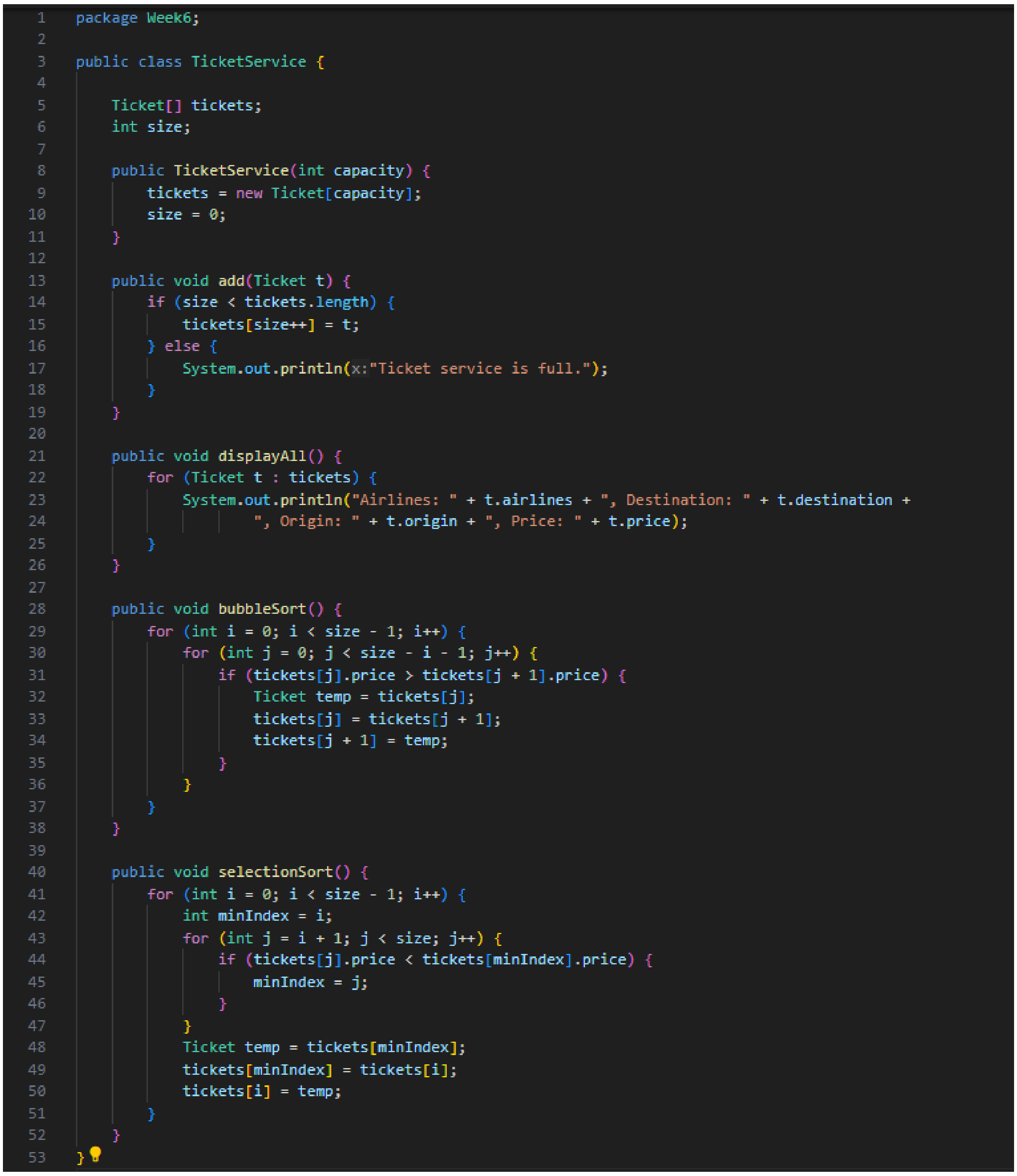
Tickets
airlines: String price: int destination: String origin : String
Ticket(String a, String dest, String origin, int price): void

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

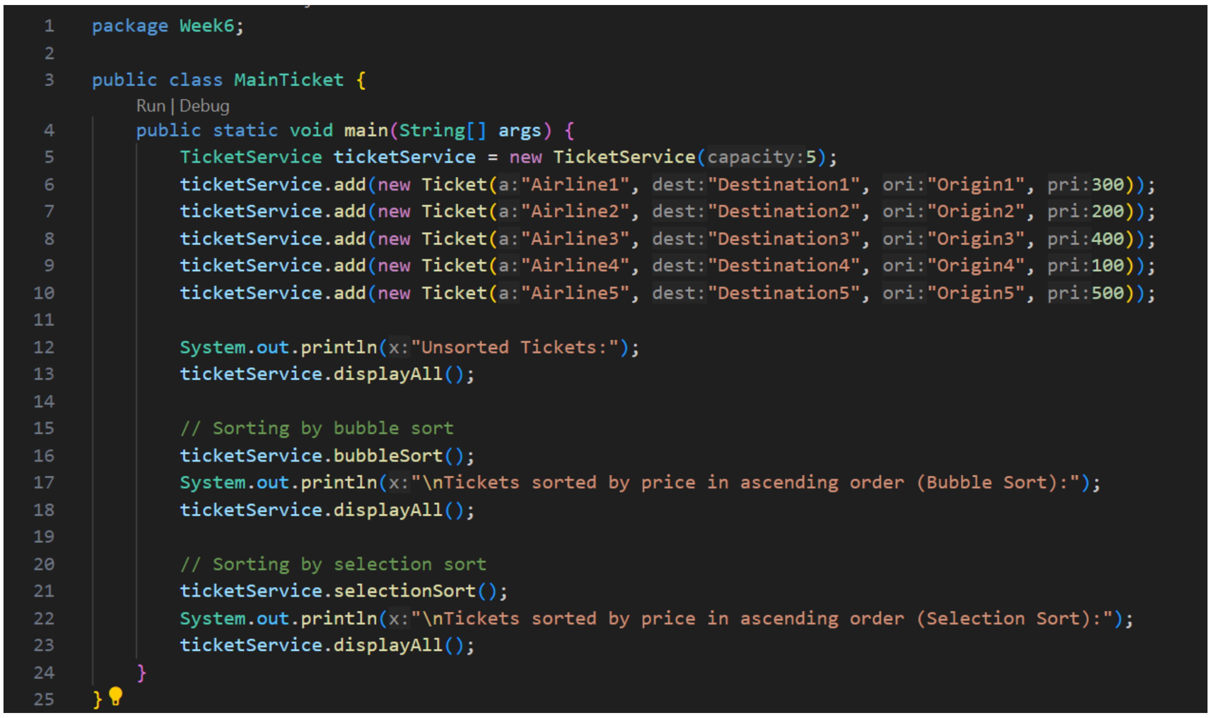
MainTicket
Main(String[] abcd): void



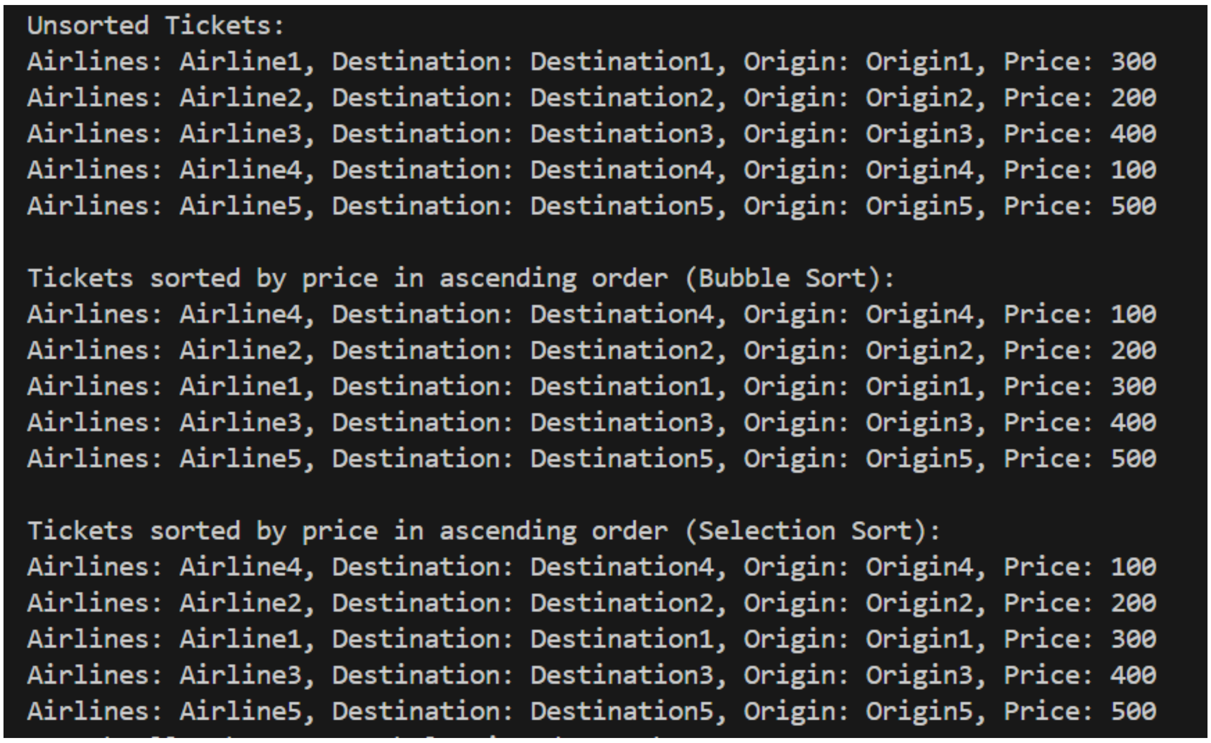
```java
package Week6;

public class Ticket {

    String airlines;
    int price;
    String destination;
    String origin;

    Ticket(String a, String dest, String ori, int pri) {
        airlines = a;
        destination = dest;
        origin = ori;
        price = pri;
    }
}
```



```java
package Week6;

public class TicketService {

    Ticket[] tickets;
    int size;

    public TicketService(int capacity) {
        tickets = new Ticket[capacity];
        size = 0;
    }

    public void add(Ticket t) {
        if (size < tickets.length) {
            tickets[size++] = t;
        } else {
            System.out.println(x:"Ticket service is full.");
        }
    }

    public void displayAll() {
        for (Ticket t : tickets) {
            System.out.println("Airlines: " + t.airlines + ", Destination: " + t.destination +
                    ", Origin: " + t.origin + ", Price: " + t.price);
        }
    }

    public void bubbleSort() {
        for (int i = 0; i < size - 1; i++) {
            for (int j = 0; j < size - i - 1; j++) {
                if (tickets[j].price > tickets[j + 1].price) {
                    Ticket temp = tickets[j];
                    tickets[j] = tickets[j + 1];
                    tickets[j + 1] = temp;
                }
            }
        }
    }

    public void selectionSort() {
        for (int i = 0; i < size - 1; i++) {
            int minIndex = i;
            for (int j = i + 1; j < size; j++) {
                if (tickets[j].price < tickets[minIndex].price) {
                    minIndex = j;
                }
            }
            Ticket temp = tickets[minIndex];
            tickets[minIndex] = tickets[i];
            tickets[i] = temp;
        }
    }
}
```

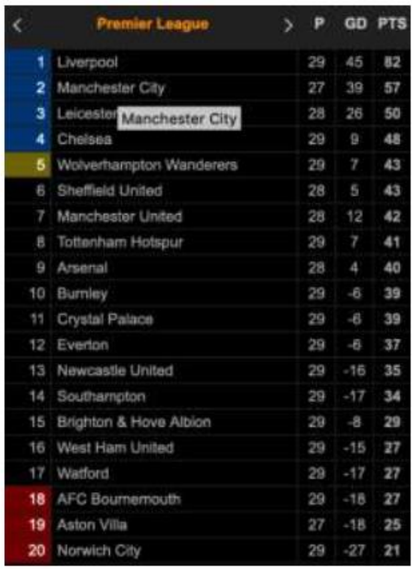


```java
package Week6;

public class MainTicket {
    Run | Debug
    public static void main(String[] args) {
        TicketService ticketService = new TicketService(capacity:5);
        ticketService.add(new Ticket(a:"Airline1", dest:"Destination1", ori:"Origin1", pri:300));
        ticketService.add(new Ticket(a:"Airline2", dest:"Destination2", ori:"Origin2", pri:200));
        ticketService.add(new Ticket(a:"Airline3", dest:"Destination3", ori:"Origin3", pri:400));
        ticketService.add(new Ticket(a:"Airline4", dest:"Destination4", ori:"Origin4", pri:100));
        ticketService.add(new Ticket(a:"Airline5", dest:"Destination5", ori:"Origin5", pri:500));

        System.out.println(x:"Unsorted Tickets:");
        ticketService.displayAll();

        // Sorting by bubble sort
        ticketService.bubbleSort();
        System.out.println(x:"\nTickets sorted by price in ascending order (Bubble Sort):");
        ticketService.displayAll();

        // Sorting by selection sort
        ticketService.selectionSort();
        System.out.println(x:"\nTickets sorted by price in ascending order (Selection Sort):");
        ticketService.displayAll();
    }
}
```



```
Unsorted Tickets:
Airlines: Airline1, Destination: Destination1, Origin: Origin1, Price: 300
Airlines: Airline2, Destination: Destination2, Origin: Origin2, Price: 200
Airlines: Airline3, Destination: Destination3, Origin: Origin3, Price: 400
Airlines: Airline4, Destination: Destination4, Origin: Origin4, Price: 100
Airlines: Airline5, Destination: Destination5, Origin: Origin5, Price: 500

Tickets sorted by price in ascending order (Bubble Sort):
Airlines: Airline4, Destination: Destination4, Origin: Origin4, Price: 100
Airlines: Airline2, Destination: Destination2, Origin: Origin2, Price: 200
Airlines: Airline1, Destination: Destination1, Origin: Origin1, Price: 300
Airlines: Airline3, Destination: Destination3, Origin: Origin3, Price: 400
Airlines: Airline5, Destination: Destination5, Origin: Origin5, Price: 500

Tickets sorted by price in ascending order (Selection Sort):
Airlines: Airline4, Destination: Destination4, Origin: Origin4, Price: 100
Airlines: Airline2, Destination: Destination2, Origin: Origin2, Price: 200
Airlines: Airline1, Destination: Destination1, Origin: Origin1, Price: 300
Airlines: Airline3, Destination: Destination3, Origin: Origin3, Price: 400
Airlines: Airline5, Destination: Destination5, Origin: Origin5, Price: 500
```

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



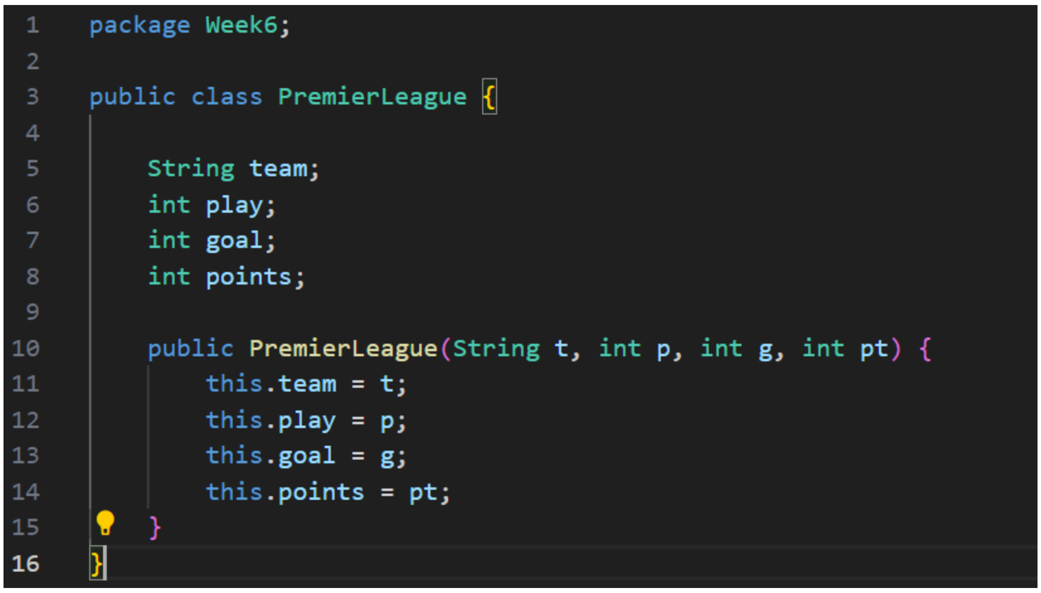
Premier League		P	GD	PTS
1	Liverpool	29	45	82
2	Manchester City	27	39	57
3	Leicester Manchester City	28	26	50
4	Chelsea	29	9	48
5	Wolverhampton Wanderers	29	7	43
6	Sheffield United	28	5	43
7	Manchester United	28	12	42
8	Tottenham Hotspur	29	7	41
9	Arsenal	28	4	40
10	Burnley	29	-6	39
11	Crystal Palace	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 function based on highest to smallest points (in ascending order) with insertion sort algorithm. Take these following class diagrams as your reference:

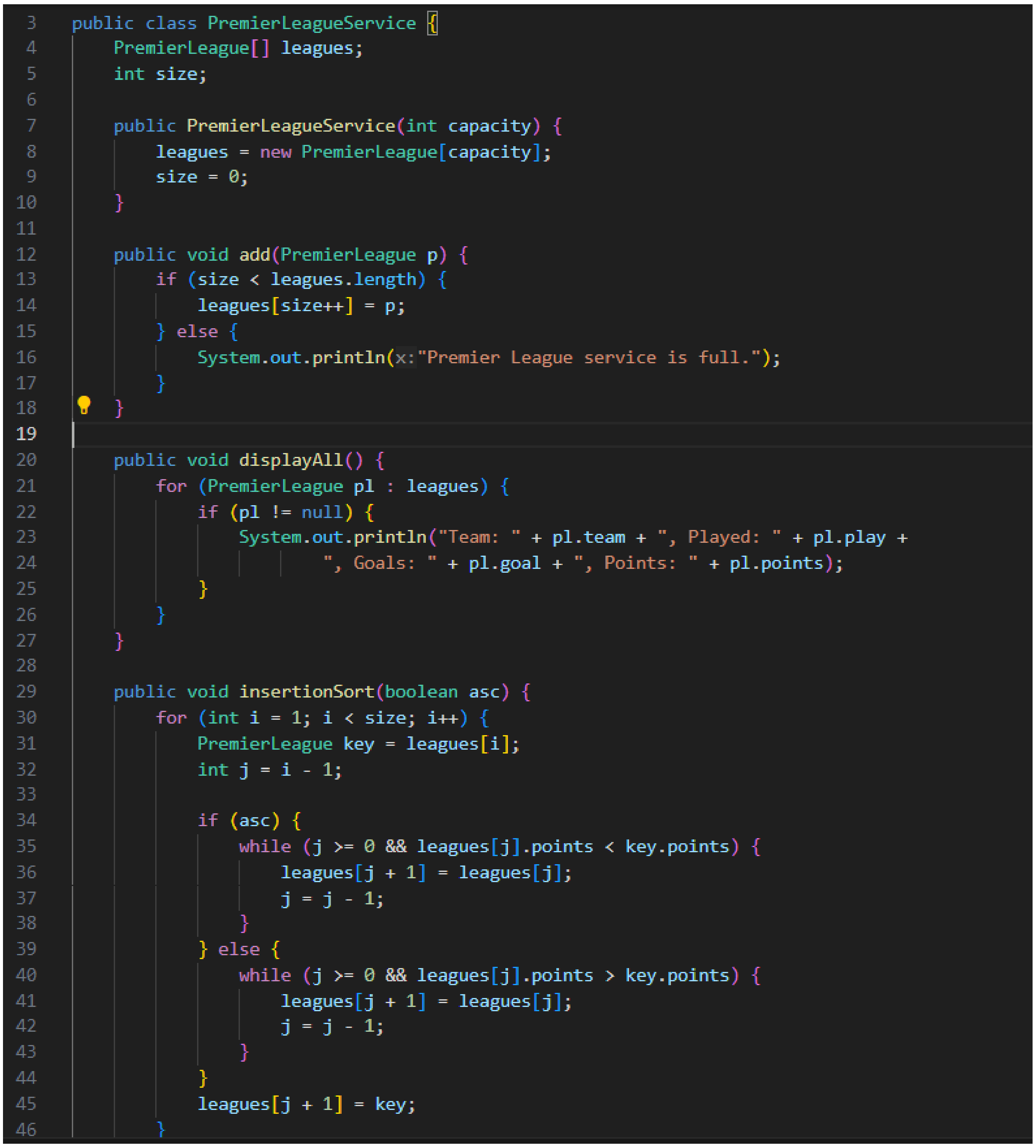
PremierLeague
team: String play: int goal: String points: String
PremierLeague(String t, int p ,int g, int pt): void

PremierLeagueService
leagues: PremierLeague()
add(PremierLeague p) : void displayAll() : void insertionSort(boolean asc) : void

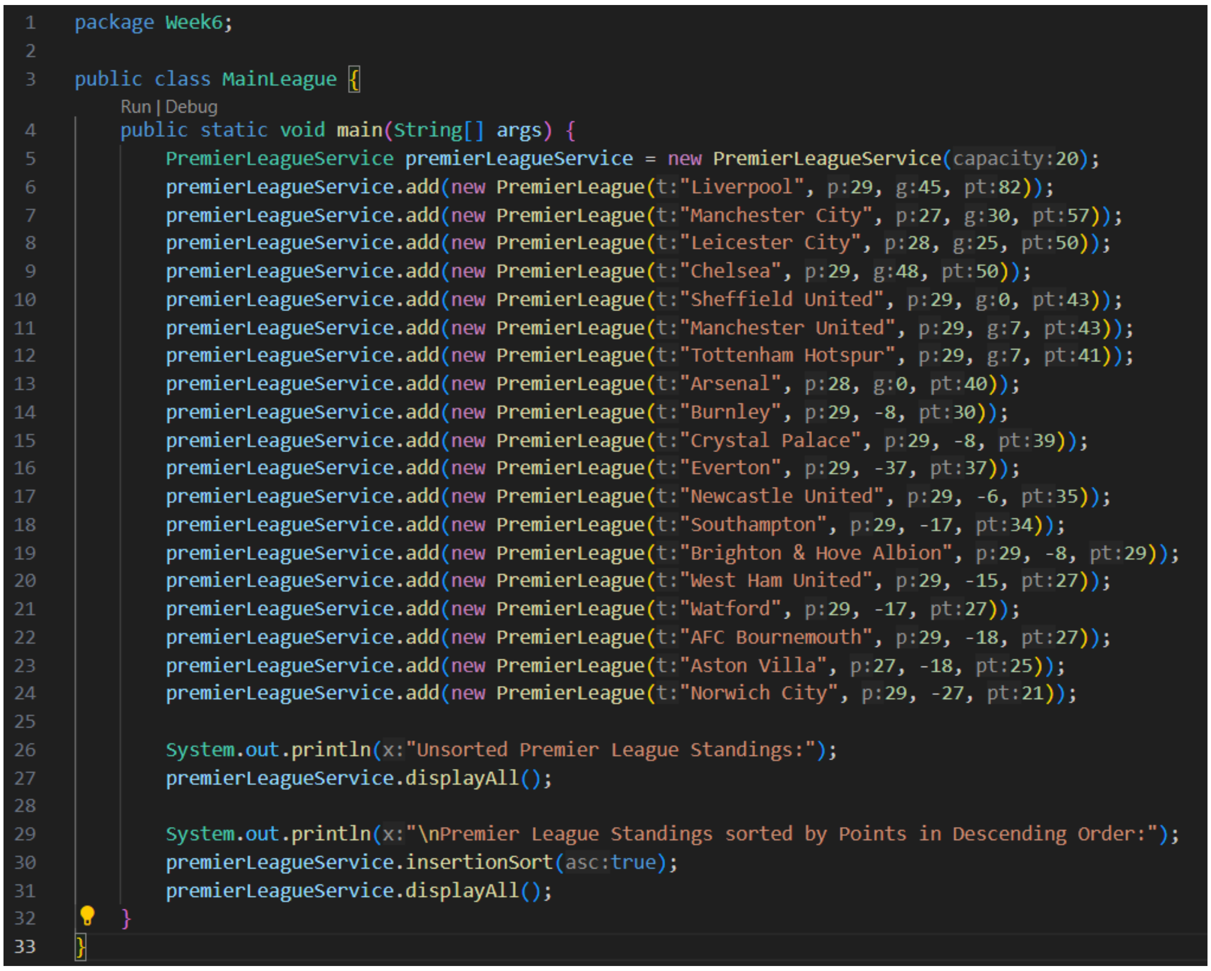
MainLeague
Main(String[] abcd): void



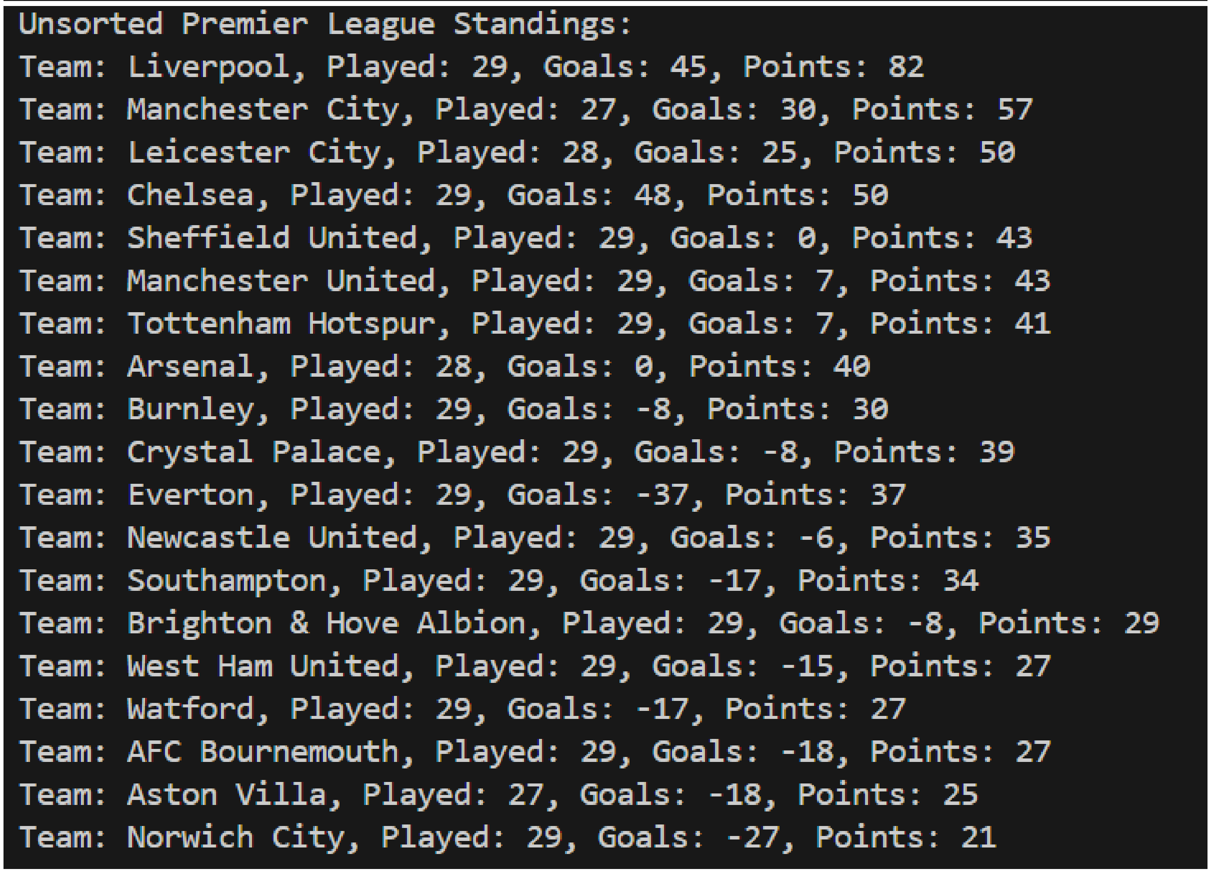
```java
package Week6;

public class PremierLeague {

    String team;
    int play;
    int goal;
    int points;

    public PremierLeague(String t, int p, int g, int pt) {
        this.team = t;
        this.play = p;
        this.goal = g;
        this.points = pt;
    }
}
```



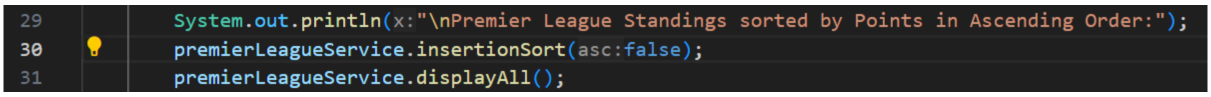
```java
public class PremierLeagueService {
    PremierLeague[] leagues;
    int size;

    public PremierLeagueService(int capacity) {
        leagues = new PremierLeague[capacity];
        size = 0;
    }

    public void add(PremierLeague p) {
        if (size < leagues.length) {
            leagues[size++] = p;
        } else {
            System.out.println(x:"Premier League service is full.");
        }
    }

    public void displayAll() {
        for (PremierLeague pl : leagues) {
            if (pl != null) {
                System.out.println("Team: " + pl.team + ", Played: " + pl.play +
                        ", Goals: " + pl.goal + ", Points: " + pl.points);
            }
        }
    }

    public void insertionSort(boolean asc) {
        for (int i = 1; i < size; i++) {
            PremierLeague key = leagues[i];
            int j = i - 1;

            if (asc) {
                while (j >= 0 && leagues[j].points < key.points) {
                    leagues[j + 1] = leagues[j];
                    j = j - 1;
                }
            } else {
                while (j >= 0 && leagues[j].points > key.points) {
                    leagues[j + 1] = leagues[j];
                    j = j - 1;
                }
            }
            leagues[j + 1] = key;
        }
```



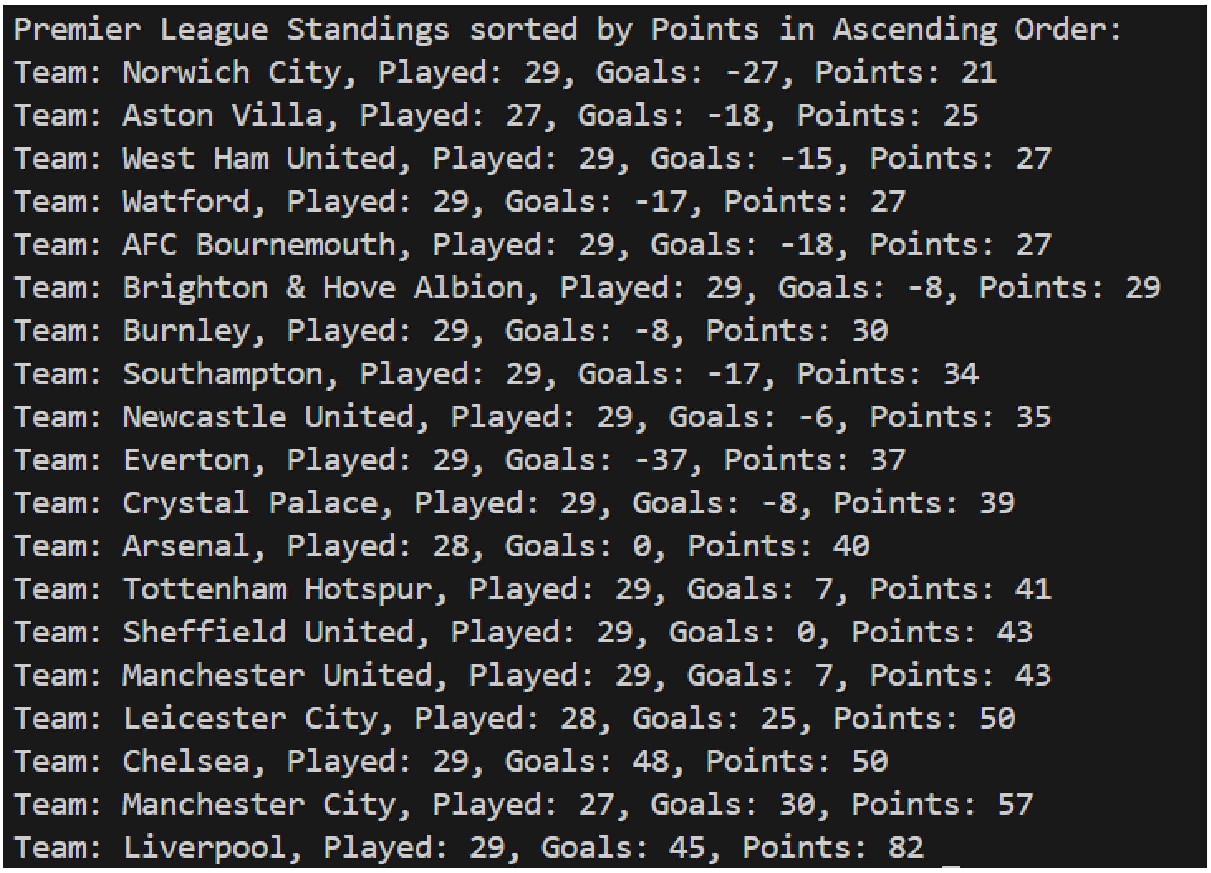
```java
package Week6;

public class MainLeague {
    Run | Debug
    public static void main(String[] args) {
        PremierLeagueService premierLeagueService = new PremierLeagueService(capacity:20);
        premierLeagueService.add(new PremierLeague(t:"Liverpool", p:29, g:45, pt:82));
        premierLeagueService.add(new PremierLeague(t:"Manchester City", p:27, g:30, pt:57));
        premierLeagueService.add(new PremierLeague(t:"Leicester City", p:28, g:25, pt:50));
        premierLeagueService.add(new PremierLeague(t:"Chelsea", p:29, g:48, pt:50));
        premierLeagueService.add(new PremierLeague(t:"Sheffield United", p:29, g:0, pt:43));
        premierLeagueService.add(new PremierLeague(t:"Manchester United", p:29, g:7, pt:43));
        premierLeagueService.add(new PremierLeague(t:"Tottenham Hotspur", p:29, g:7, pt:41));
        premierLeagueService.add(new PremierLeague(t:"Arsenal", p:28, g:0, pt:40));
        premierLeagueService.add(new PremierLeague(t:"Burnley", p:29, -8, pt:30));
        premierLeagueService.add(new PremierLeague(t:"Crystal Palace", p:29, -8, pt:39));
        premierLeagueService.add(new PremierLeague(t:"Everton", p:29, -37, pt:37));
        premierLeagueService.add(new PremierLeague(t:"Newcastle United", p:29, -6, pt:35));
        premierLeagueService.add(new PremierLeague(t:"Southampton", p:29, -17, pt:34));
        premierLeagueService.add(new PremierLeague(t:"Brighton & Hove Albion", p:29, -8, pt:29));
        premierLeagueService.add(new PremierLeague(t:"West Ham United", p:29, -15, pt:27));
        premierLeagueService.add(new PremierLeague(t:"Watford", p:29, -17, pt:27));
        premierLeagueService.add(new PremierLeague(t:"AFC Bournemouth", p:29, -18, pt:27));
        premierLeagueService.add(new PremierLeague(t:"Aston Villa", p:27, -18, pt:25));
        premierLeagueService.add(new PremierLeague(t:"Norwich City", p:29, -27, pt:21));

        System.out.println(x:"Unsorted Premier League Standings:");
        premierLeagueService.displayAll();

        System.out.println(x:"\nPremier League Standings sorted by Points in Descending Order:");
        premierLeagueService.insertionSort(asc:true);
        premierLeagueService.displayAll();
    }
}
```



```
Unsorted Premier League Standings:
Team: Liverpool, Played: 29, Goals: 45, Points: 82
Team: Manchester City, Played: 27, Goals: 30, Points: 57
Team: Leicester City, Played: 28, Goals: 25, Points: 50
Team: Chelsea, Played: 29, Goals: 48, Points: 50
Team: Sheffield United, Played: 29, Goals: 0, Points: 43
Team: Manchester United, Played: 29, Goals: 7, Points: 43
Team: Tottenham Hotspur, Played: 29, Goals: 7, Points: 41
Team: Arsenal, Played: 28, Goals: 0, Points: 40
Team: Burnley, Played: 29, Goals: -8, Points: 30
Team: Crystal Palace, Played: 29, Goals: -8, Points: 39
Team: Everton, Played: 29, Goals: -37, Points: 37
Team: Newcastle United, Played: 29, Goals: -6, Points: 35
Team: Southampton, Played: 29, Goals: -17, Points: 34
Team: Brighton & Hove Albion, Played: 29, Goals: -8, Points: 29
Team: West Ham United, Played: 29, Goals: -15, Points: 27
Team: Watford, Played: 29, Goals: -17, Points: 27
Team: AFC Bournemouth, Played: 29, Goals: -18, Points: 27
Team: Aston Villa, Played: 27, Goals: -18, Points: 25
Team: Norwich City, Played: 29, Goals: -27, Points: 21
```



```java
System.out.println(x:"\nPremier League Standings sorted by Points in Ascending Order:");
premierLeagueService.insertionSort(asc:false);
premierLeagueService.displayAll();
```



```
Premier League Standings sorted by Points in Ascending Order:
Team: Norwich City, Played: 29, Goals: -27, Points: 21
Team: Aston Villa, Played: 27, Goals: -18, Points: 25
Team: West Ham United, Played: 29, Goals: -15, Points: 27
Team: Watford, Played: 29, Goals: -17, Points: 27
Team: AFC Bournemouth, Played: 29, Goals: -18, Points: 27
Team: Brighton & Hove Albion, Played: 29, Goals: -8, Points: 29
Team: Burnley, Played: 29, Goals: -8, Points: 30
Team: Southampton, Played: 29, Goals: -17, Points: 34
Team: Newcastle United, Played: 29, Goals: -6, Points: 35
Team: Everton, Played: 29, Goals: -37, Points: 37
Team: Crystal Palace, Played: 29, Goals: -8, Points: 39
Team: Arsenal, Played: 28, Goals: 0, Points: 40
Team: Tottenham Hotspur, Played: 29, Goals: 7, Points: 41
Team: Sheffield United, Played: 29, Goals: 0, Points: 43
Team: Manchester United, Played: 29, Goals: 7, Points: 43
Team: Leicester City, Played: 28, Goals: 25, Points: 50
Team: Chelsea, Played: 29, Goals: 48, Points: 50
Team: Manchester City, Played: 27, Goals: 30, Points: 57
Team: Liverpool, Played: 29, Goals: 45, Points: 82
```