Update readme.md
Jonas Solhaug Kaad authored 3 months ago

Forked from an inaccessible project.

readme.md 3.30 KiB

Practice Assignment 1

Task 1a: Student implements Comparable<Student>

Assume, you are learning Java. You are introduced to the concepts of Comparable and Comparator interfaces. You need to practice these concepts and compare objects using both interfaces. Considering yourself a student, who needs to master these concepts, you decided to implement a student class such that you can compare student objects using both interfaces.

A class called `Student.java` is already created in the package *student_information*. Complete the implementation of the class. Remember to use correct access modifiers for variables and methods.

- Declare 5 variables for `name(String)`, `age(int)`, `department(String)`, `result(String)`, `marks(double)`.
- Create a Constructor to initialize the 5 variables.
- Implement 5 `Getter` methods for retrieving the values of all the 5 variables. A `Getter` method should return the value of the variable.
- Implement a `compareTo()` method, i.e., compare the `marks` of two `Student` objects. Remember to use the corresponding `Getter` methods for this implementation.
- A `toString()` method is already provided to print the students information. Uncomment it after completing the points above.
- In the `main()` method, implement the following:
  - Create a `Set` of type `Student`, called `studentSet1` and initialize it as a `TreeSet`.
  - Create 5 student objects with the following data:

```
| name | age | dept | result | marks |
|------|-----|------|--------|-------|
| Tim  | 20  | me   |  pass  | 9,80  |
| Bo   | 21  | me   |  pass  | 9,20  |
| Ella | 19  | ece  |  fail  | 3,20  |
| Emma | 19  | ece  |  pass  | 9,60  |
| Paul | 20  | cse  |  pass  | 8,60  |

-----------------------------------------

"Tim", 20, "me", "pass", 9.80
"Bo", 21, "me", "pass", 9.20
"Ella", 19, "ece", "fail", 3.20
"Emma", 19, "ece", "pass", 9.60
"Paul", 20, "ece", "pass", 8.60
```

- Add student objects to the set `studentSet1` using the `add()` method.
- Print all elements inside the object `studentSet1`.
- Run and test your implementation.

Example of correct output, when sorting based on **marks**:

```
[ Ella   19   ece   fail   3,20
, Paul   20   cse   pass   8,60
, Bo     21   me    pass   9,20
, Emma   19   ece   pass   9,60
```

```
, Tim    20   me    pass   9,80
]
```

Task 1b: Sorting with Comparator

- Create a class `AgeComparator.java` with the signature `public class AgeComparator implements Comparator<Student>` in the *student_information* package.
- Implement the `compare()` method such that it compares two `Student` objects by their `age` values and if two objects have the same `age`, they should be compared by their `marks` values. (**Hint**: Remember to use the corresponding `Getter` methods in the `Student` object.
- In the `main()` method of the `Student` class:
  - Creating another set `studentSet2` of type `Student` with reference to `TreeSet<>(new AgeComparator())`
  - Add the elements of `studentSet1` to `studentSet2` using the `addAll()` method.
  - Print all elements inside the object `studentSet2`.
- Run and test your implementation.

Example of correct output when sorting based on **age**:

```
[ Ella   19   ece   fail    3,20
, Emma   19   ece   pass    9,60
, Paul   20   cse   pass    8,60
, Tim    20   me    pass    9,80
, Bo     21   me    pass    9,20
]
```

```java
package student_information;

import java.util.*;

public class Student implements Comparable<Student> {
    //Opgaven er lige ud af landevejen.
    //De første dele går ud på, at lave variabler.
    //Derefter bruge konstruktør og initialisere dem der ligesom i OOP.
    //Derefter skal vi danne en Compareto-methode, hvor this.marks er brugt til at sammenligne objektets karakter.
    String name;
    int age;
    String department;
    String result;
    double marks;


    public Student(String name, int age, String department, String result, double marks){
        this.name = name;
        this.age = age;
        this.department = department;
        this.result = result;
        this.marks = marks;
    }


//uncomment the toString(). Make sure you use the same names of getter methods in the toString()
    /*
    @Override
    public String toString() {
        return String.format("%s \t \t %d \t \t %s \t \t %s \t \t %.2f\n",getName(),getAge(),getDepartment(), getResult(), getMarks());
    }
    */

    @Override
```

```
35      public int compareTo(Student s) {
36          if(this.marks < s.marks){
37              return -1;
38          }else if (this.marks > s.marks){
39              return 1;
40          }else {
41              return 0;
42          }
43      }
44
45      public String getName() {
46          return name;
47      }
48
49      public void setName(String name) {
50          this.name = name;
51      }
52
53      public int getAge() {
54          return age;
55      }
56
57      public void setAge(int age) {
58          this.age = age;
59      }
60
61      public String getDepartment() {
62          return department;
63      }
64
65      public void setDepartment(String department) {
66          this.department = department;
67      }
68
69      public String getResult() {
70          return result;
71      }
72
73      public void setResult(String result) {
74          this.result = result;
75      }
```

```
76
77      public double getMarks() {
78          return marks;
79      }
80
81      public void setMarks(double marks) {
82          this.marks = marks;
83      }
84
85      //Når alle de nemme ting er gjort, skal vi bare
    danne objekterne udefra TreeSet.
86      //Husk altid at en TreeSet fremviser objekter
    sorteret i alfabetisk rækkefølge.
87
88      public static void main(String[] args) {
89          TreeSet<Student> studentSet = new TreeSet
    <>();
90
91
92          Student student1 = new Student("Vivek",19,"
    pass","1",12.0);
93          Student student3 = new Student("Anders",22,"
    pass","3",7.0);
94          Student student5 = new Student("Annie",25,"
    fail","4",4.0);
95          Student student4 = new Student("Amila",24,"
    fail","4",4.0);
96          Student student2 = new Student("Sofie",21,"
    pass","2",10.0);
97          //Alle objekter er herunder tilføjet til
    TreeSet gennem add-metoden.
98          //Man kan også direkte gør sådan: studentset
    .add(new student osv.)
99          studentSet.add(student1);
100         studentSet.add(student2);
101         studentSet.add(student3);
102         studentSet.add(student4);
103         studentSet.add(student5);
104
105         //Løsning for 1.a er ikke helt rigtigt men
    den sorterer tingene på en rigtig måde.
```

```
106         //Task 1a
107         System.out.println("Sorting based on Marks:"
    );
108         System.out.println(student1.age);
109         for(Student student: studentSet){
110             for(Student studerende: studentSet){
111                 System.out.println(student.marks);
112             }
113         }
114
115
116
117         //Denne samme princip er gældende her, bare
    istedet med alderen.
118             //Task 1b
119             System.out.println("Sorting based on Age
    :");
120             System.out.println(student1.age);
121             for(Student student: studentSet){
122                 for(Student studerende:  studentSet
    ){
123                     System.out.println(student.age);
124                 }
125             }
126     }
127
128 }
129
```

```java
package student_information;

import java.util.Comparator;
import java.util.Set;
import java.util.TreeSet;

public class AgeComparator implements Comparator<Student> {
    //Alle ting er bare lige ud af landevejen herhenne.
    String name;
    int age;
    String department;
    String result;
    double marks;

    public AgeComparator(String name, int age, String department, String result, double marks){
        this.name = name;
        this.age = age;
        this.department = department;
        this.result = result;
        this.marks = marks;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

```

```
39      public String getDepartment() {
40          return department;
41      }
42
43      public void setDepartment(String department) {
44          this.department = department;
45      }
46
47      public String getResult() {
48          return result;
49      }
50
51      public void setResult(String result) {
52          this.result = result;
53      }
54
55      public double getMarks() {
56          return marks;
57      }
58
59      public void setMarks(double marks) {
60          this.marks = marks;
61      }
62
63      @Override
64      public int compare(Student o1, Student o2) {
65          if(o1.age<o2.age){
66              return 1;
67          }else if(o1.age>o2.age){
68              return -1;
69          }else{
70              return 0;
71          }
72      }
73
74      //Du kan herhenne se under Main-metoden at vi har
    lavet vores tree-set igen.
75      //Der er lidt fejl og det er fordi vi ikke har
   tilføjet AgeComparator pga. IntelliJ bliver sur.
76      public static void main(String[] args){
77          TreeSet<Student> studentSet = new TreeSet
```

```
77 <>();
78         studentSet.add(new Student("Vivek",19,"pass"
   ,"1",12.0));
79         studentSet.add(new Student("Anders",22,"pass
   ","3",7.0));
80         studentSet.add(new Student("Annie",25,"fail"
   ,"4",4.0));
81         studentSet.add(new Student("Amila",24,"fail"
   ,"4",4.0));
82         studentSet.add(new Student("Sofie",21,"pass"
   ,"2",10.0));
83 
84         //Vores nye TreeSet bliver dannet og teknisk
    indsætter vi alle de objekter som er dannet bare
   herinde.
85         Set<Student> studentnewSet = new TreeSet
   <>();
86         studentnewSet.addAll(studentSet);
87 
88         //Ligesom med alderen og karakterne i
   Student.java klassen, har vi brugt den samme princip
    her.
89         //Her har vi formåede at udprinte elevernes
   navne, alder og karakter i alfabetisk rækkefølge.
90         for(Student studerende: studentnewSet){
91             System.out.println("Name :" + studerende
   .getName() + ", Age: " + studerende.getAge() + ",
   Marks:" + studerende.getMarks());
92         }
93     }
94 }
95 
```