Java OOP 10128

Java Interface

Pini shlomi

Array Sort

- Arrays.sort get array of elements and sorts its elements
- it's working fine for basic element's types because the compare operator defined

```
public static void main(String[] args) {
        int[] arr = {32,5,2,8,4,6,9,7};
        System.out.println("Before sorting: " + Arrays.toString(arr));
        Arrays.sort(arr);
        System.out.println("After  sorting: " + Arrays.toString(arr));
}
```

```
                console
Before sorting: [32, 5, 2, 8, 4, 6, 9, 7]
After  sorting: [2, 4, 5, 6, 7, 8, 9, 32]
```

Array Sort - Objects

```java
public class Person {
		private String name;
		private int age;
		public Person(String name, int age) {
			this.name = name;
			this.age = age;
		}

		@Override
		public String toString() {
			return "{" + name + ", " + age + "}";
		}
}
```

```java
Person[] arr =  {new Person("Yosi", 27),new Person("Yael", 22),new Person("Mor", 25)};
System.out.println("Before sorting: " +    Arrays.toString(arr));
Arrays.sort(arr);
System.out.println("After  sorting: " + Arrays.toString(arr));
```

console

```
Before sorting: [{Yosi, 27}, {Yael, 22}, {Mor, 25}]
Exception in thread "main" java.lang.ClassCastException: class Person cannot be cast to
class java.lang.Comparable …
```

Comparable Interface

```
public class Person implements Comparable<Person>   {
        private String name;
        private int age;
        public Person(String name, int age)  {…}

        @Override
        public String toString()  {…}

        @Override
        public int compareTo(Person other)  {
                return this.age - other.age;
        }
}

Person[] arr =  {new Person("Yosi", 27),new Person("Yael", 22),new Person("Mor", 25)};
System.out.println("Before sorting: " + Arrays.toString(arr));
Arrays.sort(arr);
System.out.println("After  sorting: " + Arrays.toString(arr));
```

console

```
Before sorting: [{Yosi, 27}, {Yael, 22}, {Mor, 25}]
After  sorting: [{Yael, 22}, {Mor, 25}, {Yosi, 27}]
```

תרגיל 1: מה יקרה בהרצת הקוד הבא?

```java
abstract class Person implements Comparable<Person> {
    protected final String name;
    protected Person(String name) {
        this.name = name;
    }
    @Override
    public int compareTo(Person other) {
        return name.compareTo(other.name);
    }
}
class Student extends Person {
    public Student(String name) {
        super(name);
    }

    @Override
    public int compareTo(Person other) {
        Student s = (Student) other;
        return name.compareTo(s.name);
    }
}
```

```java
class Lecturer extends Person {
    public Lecturer(String name) {
        super(name);
    }
}

public class Main {
    public static void main(String[] args) {
        Person p1 = new Student("Noa");
        Person p2 = new Lecturer("Adam");
        System.out.println(p1.compareTo(p2));
    }
}
```

Comparable and equals

```java
class Person implements Comparable<Person> {
    protected final String name;
    protected final int age;
    protected Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    @Override
    public int compareTo(Person other) {
        return Integer.compare(age, other.age);
    }
    @Override
    public boolean equals(Object obj) {
        if (this == obj){
            return true;
        }
        if (obj instanceof Person person){
            return name.equals(person.name) && age == person.age;
        }
        return false;
    }
    @Override
    public String toString() {
        return "(" + name + ", " + age + ")";
    }
}
```

```java
public class Main {
    public static void main(String[] args) {
        Person p1 = new Person("Alice", 30);
        Person p2 = new Person("Bob", 30);
        System.out.println(p1.compareTo(p2));
        System.out.println(p1.equals(p2));
        TreeSet<Person> treeSet = new TreeSet<>();
        treeSet.add(p1);
        treeSet.add(p2);
        System.out.println(treeSet);
        HashMap<Person, Integer> hashMap = new HashMap<>();
        hashMap.put(p1, 10);
        hashMap.put(p2, 14);
        System.out.println(hashMap);
    }
}
```

console

```
Comparing by compare: 0
Comparing by equals: false
[(Alice, 30)]
{(Bob, 30)=14, (Alice, 30)=10}
```

Java Practice Exercises

אפקה המכללה האקדמית להנדסה בתל אביב

פיני שלומי מהנדס תוכנה 054-4636992

תרגיל 2 : מה יקרה בהרצת הקוד הבא?

```java
class Employee implements Comparable<Employee> {
    protected final String name;
    public Employee(String name) {
        this.name = name;
    }
    @Override
    public int compareTo(Employee other) {
            return name.compareTo(other.name);
    }
    @Override
    public String toString() {
        return name ;
    }
}
class Manager extends Employee {
    private final int level;

    public Manager(String name, int level) {
        super(name);
        this.level = level;
    }
@Override
    public int compareTo(Employee other) {
        if (other instanceof Manager m) {
            int res = Integer.compare(this.level, m.level);
            if (res != 0) return res;
        }
        return super.compareTo(other);
    }
}
public class Main {
    public static void main(String[] args) {
        Employee e1 = new Manager("Kobi", 3);
        Employee e2 = new Employee("Dor");
        Employee e3 = new Manager("Keren", 2);
        Employee e4 = new Manager("Yahara", 3);
        Employee e5 = new Employee("Tom");
        System.out.println(e1 + " vs "  + e2 + ": " + e1.compareTo(e2));
        System.out.println(e2 + " vs "  + e3 + ": " + e2.compareTo(e3));
        System.out.println(e1 + " vs "  + e4 + ": " + e1.compareTo(e4));
        System.out.println(e3 + " vs "  + e4 + ": " + e3.compareTo(e4));
        System.out.println(e2 + " vs "  + e5 + ": " + e2.compareTo(e5));
    }
}
```

Comparator - multi comparing

```java
public class Person{
          private String name;
          private int age;
          public Person(String name, int age) {…}
          @Override
          public String toString() {…}

          public String getName() {..}

          public int getAge() {…}
}
public class ComparePersonByAge implements Comparator<Person> {

          @Override
          public int compare(Person p1, Person p2) {
                    return p1.getAge() - p2.getAge();
          }
}
 public class ComparePersonByName implements Comparator<Person> {

          @Override
          public int compare(Person p1, Person p2) {
                    return p1.getName().compareTo(p2.getName());
          }
 }
```

Removing Comparable Interface

Add getter's

Using Comparator interface

Using String method

main – using Comparator

```
Person[] arr =  {
new Person("Yosi", 27),
new Person("Yael", 22),
new Person("Mor", 25)
};

System.out.println("Before sorting: " + Arrays.toString(arr));
Arrays.sort(arr, new ComparePersonByName());
System.out.println("After  sorting by name: " + Arrays.toString(arr));
Arrays.sort(arr, new ComparePersonByAge());
System.out.println("After  sorting by age: " + Arrays.toString(arr));
```

console

```
Before sorting: [{Yosi, 27}, {Yael, 22}, {Mor, 25}]
After  sorting by name: [{Mor, 25}, {Yael, 22}, {Yosi, 27}]
After  sorting by age: [{Yael, 22}, {Mor, 25}, {Yosi, 27}]
```

Java Practice Exercises

אפקה המכללה האקדמית להנדסה בתל אביב

פיני שלומי מהנדס תוכנה 054-4636992

תרגיל 3 : מה יקרה בהרצת הקוד הבא?

```
final class Product {
    private final String name;
    private final double price;

    public Product(String name, double price) {
        this.name = name;
        this.price = price;
    }
    public String getName() { return name; }
    public double getPrice() { return price; }
    @Override
    public String toString() {
        return name + " ($" + price + ")";
    }
}

class PriceAscComparator implements Comparator<Product> {
    @Override
    public int compare(Product p1, Product p2) {
        return Double.compare(p1.getPrice(), p2.getPrice());
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Product laptop = new Product("Laptop", 1200.00);
        Product mouse = new Product("Mouse", 25.00);
        Product keyboard = new Product("Keyboard", 75.00);
        PriceAscComparator priceComp = new PriceAscComparator();
        System.out.println(priceComp.compare(laptop, mouse));
        System.out.println(priceComp.compare(mouse, keyboard));
        System.out.println(priceComp.compare(keyboard, laptop));
    }
}
```

תרגיל 4: תוכנית למימוש ממשקי java

קישור לקובץ התרגיל

קישור ל-starter

אתר להורדת ספריות מ-github.

Animals

```java
public abstract class Animal{
  private String name;
  private String color;

  public Animal(String name, String color) {…}

  public Animal(Animal other) {…}

  public String getColor() {…}

  public void setColor(String color) {…}

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

```java
public class Horse extends Animal{
  private int height;

  public Horse(String name, String color, int height) {…}

  public Horse(Horse other) {…}

  public void ride() {…}

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

```java
public class Cat extends Animal{
  private double whiskerLen;

  public Cat(String name, String color, int wiskersLen) {…}

  public Cat(Cat other) {…}

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

```java
public class Fish extends Animal {
  public Cat(String name, String color) {…}

  public Fish(Fish other) {…}

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

main - animals

אפקה
המכללה האקדמית להנדסה בתל אביב

פיני שלומי מהנדס תוכנה 054-4636992

```
public static void main(String[] args) {
          Animal[] animals = new Animal[3];
          animals[0] = new Cat("Pitzi", "Brown", 5.7);
          animals[1] = new Fish("Dagi", "gold");
          animals[2] = new Horse("Davi", "Black", 184);

          Animal[] newAnimals = new Animal[3];
          for (int i = 0; i < animals.length; i++) {
                    newAnimals[i] = animals[i];
          }
          animals[0].setColor("blue");
          System.out.println("Animals:");
          System.out.println(Arrays.toString(animals));
          System.out.println("New animals:");
          System.out.println(Arrays.toString(newAnimals));
}
```

console

```
Animals:
[Cat: Pitzi, blue, 5.7, Fish: Dagi, gold, Horse: Davi, Black, 184]
New animals:
[Cat: Pitzi, blue, 5.7, Fish: Dagi, gold, Horse: Davi, Black, 184]
```

main – animals (deep copy)

אפקה
המכללה האקדמית להנדסה בתל אביב

פיני שלומי מהנדס תוכנה 054-4636992

```java
public static void main(String[] args) {
          Animal[] animals = new Animal[3];
          animals[0] = new Cat("Pitzi", "Brown", 5.7);
          animals[1] = new Fish("Dagi", "gold");
          animals[2] = new Horse("Davi", "Black", 184);

          Animal[] newAnimals = new Animal[3];
          for (int i = 0; i < animals.length; i++) {
                    if (animals[i] instanceof Cat) {
                              newAnimals[i] = new Cat((Cat)animals[i]);
                    } else if (animals[i] instanceof Fish) {
                              newAnimals[i] = new Fish((Fish)animals[i]);
                    } else if (animals[i] instanceof Horse) {
                              newAnimals[i] = new Horse((Horse)animals[i]);
                    }
          }
          animals[0].setColor("blue");
          System.out.println("Animals:");
          System.out.println(Arrays.toString(animals));
          System.out.println("New animals:");
          System.out.println(Arrays.toString(newAnimals));
}
```

Using copy constructor for each class

console

```
Animals:
[Cat: Pitzi, blue, 5.7, Fish: Dagi, gold, Horse: Davi, Black, 184]
New animals:
[Cat: Pitzi, Brown, 5.7, Fish: Dagi, gold, Horse: Davi, Black, 184]
```

main – animals - clone

```
public static void main(String[] args) {
        Animal[] animals = new Animal[3];
        animals[0] = new Cat("Pitzi", "Brown", 5.7);
        animals[1] = new Fish("Dagi", "gold");
        animals[2] = new Horse("Davi", "Black", 184);

        Animal[] newAnimals = new Animal[3];
        for (int i = 0; i < animals.length; i++) {
                newAnimals[i] = animals[i].clone();
        }
        animals[0].setColor("blue");
        System.out.println("Animals:");
        System.out.println(Arrays.toString(animals));
        System.out.println("New animals:");
        System.out.println(Arrays.toString(newAnimals));
}
```

Using clone method

console

```
Animals:
[Cat: Pitzi, blue, 5.7, Fish: Dagi, gold, Horse: Davi, Black, 184]
New animals:
[Cat: Pitzi, Brown, 5.7, Fish: Dagi, gold, Horse: Davi, Black, 184]
```

Cloneable

```
public abstract class Animal implements Cloneable  {
  private String name;
  private String color;

  public Animal(String name, String color) {…}

  public Animal(Animal other) {…}

  public String getColor() {…}

  public void setColor(String color) {…}

  @Override
  protected Animal clone() throws CloneNotSupportedException {
   return (Animal) super.clone();
  }

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

Java Practice Exercises

תרגיל 5: מה יקרה בהרצת הקוד הבא?

אפקה המכללה האקדמית להנדסה בתל אביב

פיני שלומי מהנדס תוכנה 054-4636992

```java
class Gadget implements Cloneable {
   private final String model;
   private final int serialNumber;
   public Gadget(String model, int serialNumber) {
      this.model = model;
      this.serialNumber = serialNumber;
   }
   @Override
   protected Gadget clone() throws CloneNotSupportedException {
      return (Gadget) super.clone();
   }
   @Override
   public String toString() {
      return "Gadget [Model: " + model + ", S/N: " + serialNumber + "]";
   }
}
class Tool implements Cloneable {
   private final String type;
   public Tool(String type) {
      this.type = type;
   }
   @Override
   public String toString() {
      return "Tool [Type: " + type + "]";
   }
}
```

```java
public class Main {
   public static void main(String[] args) {
      Gadget g1 = new Gadget("SmartPhone X", 1001);
      Tool t1 = new Tool("Hammer");
      System.out.println("Original Gadget: " + g1);
      System.out.println("Original Tool: " + t1);
      try {
         Gadget g2 = g1.clone();
         System.out.println("Cloned Gadget: " + g2);
         Tool t2 = t1.clone();
         System.out.println("Cloned Tool: " + t2);
      } catch (CloneNotSupportedException e) {
         System.out.println("Caught exception: “
                + e.getMessage());
         System.out.println("Type of exception: "
               + e.getClass().getName());
      }
   }
}
```

Java Practice Exercises

אפקה המכללה האקדמית להנדסה בתל אביב

פיני שלומי מהנדס תוכנה 054-4636992

תרגיל 6: תוכנית למימוש קלט פלט

קישור לקובץ התרגיל

קישור ל-starter

אתר להורדת ספריות מ-github.