抽象类与接口的异同

相同点： 抽象化，两者都可以包含抽象方法，即没有具体实现的方法，子类或实现类必须提供这些方法的具体实现。

多态性支持：通过抽象类和接口，可以实现多态性，允许使用父类型引用指向子类型对象，从而在运行时根据实际对象类型调用相应的方法。

不同点：

特性	抽象类	接口
继承方式	一个类只能继承一个抽象类	一个类可以实现多个接口
构造器	可以有构造器	不能有构造器
成员变量	可以拥有实例变量、静态变量	默认情况下，所有变量都是`public static final`
方法实现	可以包含非抽象方法（带有实现）	Java 8之前，只能包含抽象方法；Java 8之后，可以包含默认方法和静态方法
访问修饰符	成员和方法可以有不同的访问级别（private, protected, public）	方法默认为`public`，Java 8后默认方法也可以是`public`或`private`
设计意图	更适合“is-a”的关系，比如Car是一个Vehicle	更适合定义能力或行为，比如Runnable表示可运行

阅读以下代码并补全Rectangle类，包括： 1、矩形的长宽变量 2、构造函数 3、重写求面积的方法

```
import java.util.ArrayList;
import java.util.List;
// Shape.java
abstract class Shape {
    // 抽象方法
    public abstract double getArea();
}

// Circle.java
class Circle extends Shape {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

    @Override
    public double getArea() {
        return Math.PI * radius * radius;
    }
}
```

```java
// Rectangle.java
class Rectangle extends Shape {

}

// TestShapes.java

class TestShapes {
    public static void main(String[] args) {
        List<Shape> shapes = new ArrayList<>();
        shapes.add(new Circle(5));
        shapes.add(new Rectangle(4, 6));

        for (Shape shape : shapes) {
            System.out.println("The area of the shape is: " + shape.getArea());
        }
    }
}
```

运行以下代码，将书本类根据年份进行排序。 代码运行成功以后，请完成： 1、使用课上学过的SortAll类进行排序，替换Arrays.sort(books); 2、排序逻辑根据ID进行排序，而非年份

```java
// Book.java
import java.util.Arrays;

class Book implements Comparable<Book> {
    private int ID;
    private String title;
    private String author;
    private int year;

    public Book(int ID,String title, String author, int year) {
        this.ID = ID;
        this.title = title;
        this.author = author;
        this.year = year;
    }

    @Override
    public int compareTo(Book other) {
        int yearComparison = Integer.compare(this.year, other.year);

        return yearComparison;

    }

    @Override
    public String toString() {
        return "Book [title=" + title + ", author=" + author + ", year=" + year +
"]";
```

```
    }
}



class SortBooks {
    public static void main(String[] args) {
        Book[] books = {
                new Book(1,"The Great Gatsby", "F. Scott Fitzgerald", 1925),
                new Book(2,"1984", "George Orwell", 1949),
                new Book(3,"To Kill a Mockingbird", "Harper Lee", 1960),
                new Book(4,"Animal Farm", "George Orwell", 1945)
        };

        Arrays.sort(books);

        for (Book book : books) {
            System.out.println(book);
        }
    }
}
```