1 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 2 //  
 3 // Name: Derek Gallardo  
 4 // Date: July 17th, 2022  
 5 // Version: 1.0.0  
 6 // Programming Language: Java  
 7 // Java Version: 17  
 8 // Description: We have been asked to create a program to keep track of the dinosaurs in the park. Our park has both herbivores and carnosaurs.  
 9 //  
 10 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 11   
 12 import java.util.\*;  
 13   
 14   
 15 public class Test {  
 16 public static void main(String[] args) {  
 17   
 18 // Create Data Structure for List of Dinosaurs  
 19 List<Dinosaur> dinosaurs = new ArrayList<>();  
 20 dinosaurs.add(new Thyrannosaurus());  
 21 dinosaurs.add(new Dilophosaurus());  
 22 dinosaurs.add(new Chindesaurus());  
 23 dinosaurs.add(new Staurikosaurus());  
 24 dinosaurs.add(new Iguanodon());  
 25 dinosaurs.add(new Parasaurolophus());  
 26 dinosaurs.add(new Stenopelix());  
 27 dinosaurs.add(new Triceratops());  
 28   
 29 // Print List of Dinosaurs  
 30 System.out.println("List of dinosaurs: ");  
 31 for (Dinosaur d : dinosaurs) {  
 32 System.out.println(d);  
 33 }  
 34   
 35   
 36 // Sort list of Dinosaurs by Height  
 37 System.out.println();  
 38 System.out.println("Sorted by Height: ");  
 39 Collections.sort(dinosaurs);  
 40 for (Dinosaur d : dinosaurs) {  
 41 System.out.println(d);  
 42 }  
 43 System.out.println();  
 44 System.out.println();  
 45   
 46 // Begin Menu  
 47   
 48 try (Scanner scanner = new Scanner(System.in)) {  
 49 boolean isOver = false;  
 50 while (!isOver) {  
 51 int choice = getMenuChoice(scanner);  
 52 switch (choice) {  
 53 case 0:  
 54 isOver = true;  
 55 break;  
 56 case 1:  
 57 System.out.println("Choose dinosaur:");  
 58 Dinosaur d = chooseDinosaur(scanner, dinosaurs);  
 59 System.out.println(d);  
 60 break;  
 61 case 2:  
 62 System.out.println("Choose 1st dinosaur to compare:");  
 63 Dinosaur d1 = chooseDinosaur(scanner, dinosaurs);  
 64 System.out.println("Choose 2nd dinosaur to compare:");  
 65 Dinosaur d2 = chooseDinosaur(scanner, dinosaurs);  
 66 int result = d1.compareTo(d2);  
 67 if (result < 0) {  
 68 System.out.println(d1.getName() + " < " + d2.getName());  
 69 } else if (result > 0) {  
 70 System.out.println(d1.getName() + " > " + d2.getName());  
 71 } else {  
 72 System.out.println(d1.getName() + " = " + d2.getName());  
 73 }  
 74 break;  
 75 default:  
 76 throw new IllegalStateException();  
 77 }  
 78 }  
 79 }  
 80   
 81 }  
 82   
 83   
 84 // Menu Choice  
 85 private static int getMenuChoice(Scanner scanner) {  
 86 while (true) {  
 87 StringBuilder builder = new StringBuilder("Menu:" + System.lineSeparator());  
 88 builder.append("1. View dinosaur info").append(System.lineSeparator());  
 89 builder.append("2. Compare to dinosaurs").append(System.lineSeparator());  
 90 builder.append("0. Exit").append(System.lineSeparator());  
 91 builder.append("Your choice: ");  
 92 System.out.print(builder);  
 93 try {  
 94 int choice = Integer.parseInt(scanner.nextLine());  
 95 if (choice < 0 || choice > 2) {  
 96 throw new IllegalArgumentException();  
 97 }  
 98 return choice;  
 99 } catch (Exception e) {  
100 System.out.println("Invalid input");  
101 }  
102 System.out.println();  
103 }  
104 }  
105   
106 // Dinosaur Choice  
107   
108 private static Dinosaur chooseDinosaur(Scanner scanner, List<Dinosaur> dinosaurs) {  
109 while (true) {  
110 StringBuilder builder = new StringBuilder();  
111 for (int i = 0; i < dinosaurs.size(); i++) {  
112 builder.append((i + 1)).append(". ").append(dinosaurs.get(i).getName()).append(System.lineSeparator());  
113 }  
114 builder.append("Your choice: ");  
115 System.out.print(builder);  
116 try {  
117 int choice = Integer.parseInt(scanner.nextLine());  
118 if (choice <= 0 || choice > dinosaurs.size()) {  
119 throw new IllegalArgumentException();  
120 }  
121 return dinosaurs.get(choice-1);  
122 } catch (Exception e) {  
123 System.out.println("Invalid input");  
124 }  
125 System.out.println();  
126 }  
127 }  
128 }  
129  
501 public interface Dinosaur extends Comparable<Dinosaur> {  
2 String getPeriod();  
3 String getWhereFound();  
4 double getHeight();  
5 }  
6

1 public abstract class Herbivore implements Dinosaur {  
 2 private final String name;  
 3 private final double height;  
 4 private final String period;  
 5 private final String whereFound;  
 6   
 7 public Herbivore(String name, double height, String period, String whereFound) {  
 8 this.name = name;  
 9 this.height = height;  
10 this.period = period;  
11 this.whereFound = whereFound;  
12 }  
13   
14 @Override  
15 public double getHeight() {  
16 return height;  
17 }  
18   
19 @Override  
20 public String getPeriod() {  
21 return period;  
22 }  
23   
24 @Override  
25 public String getWhereFound() {  
26 return whereFound;  
27 }  
28   
29 @Override  
30 public int compareTo(Dinosaur o) {  
31 return Double.compare(height, o.getHeight());  
32 }  
33   
34 @Override  
35 public String toString() {  
36 return name + ". Height: " + height + "m. Found in " + whereFound + ". Lived in " + period + "." ;  
37 }  
38 }  
39 1 public abstract class Carnivore implements Dinosaur {  
 2 private final String name;  
 3 private final double height;  
 4 private final String period;  
 5 private final String whereFound;  
 6   
 7 public Carnivore(String name, double height, String period, String whereFound) {  
 8 this.name = name;  
 9 this.height = height;  
10 this.period = period;  
11 this.whereFound = whereFound;  
12 }  
13   
14 @Override  
15 public double getHeight() {  
16 return height;  
17 }  
18   
19 @Override  
20 public String getPeriod() {  
21 return period;  
22 }  
23   
24 @Override  
25 public String getWhereFound() {  
26 return whereFound;  
27 }  
28   
29 @Override  
30 public int compareTo(Dinosaur o) {  
31 return Double.compare(height, o.getHeight());  
32 }  
33   
34 @Override  
35 public String toString() {  
36 return name + ". Height: " + height + "m. Found in " + whereFound + ". Lived in " + period + "." ;  
37 }  
38 }  
391 public class EarlyDinosaurs extends Carnivore {  
2 public EarlyDinosaurs(String name, double height, String period, String whereFound) {  
3 super(name, height, period, whereFound);  
4 }  
5 }  
61 public class Theropods extends Carnivore {  
2 private double weight;  
3   
4 public Theropods(String name, double height, String period, String whereFound, double weight) {  
5 super(name, height, period, whereFound);  
6 this.weight = weight;  
7 }  
8 }  
91 public class Ceratopsians extends Herbivore {  
2 public Ceratopsians(String name, double height, String period, String whereFound) {  
3 super(name, height, period, whereFound);  
4 }  
5 }  
61 public class Chindesaurus extends EarlyDinosaurs {  
2 public Chindesaurus() {  
3 super("Chindesaurus", 4.0, "Late Triassic", "USA");  
4 }  
5 }  
61 public class Dilophosaurus extends Theropods {  
2 public Dilophosaurus() {  
3 super("Dilophosaurus", 6.0, "Early Jurassic", "USA", 300);  
4 }  
5 }  
6

1 public class Iguanodon extends Ornithopods {  
2 public Iguanodon() {  
3 super("Iguanodon", 10.0, "Early Cretaceous", "Belgium, England");  
4 }  
5 }  
6

1 public class Ornithopods extends Herbivore {  
2 public Ornithopods(String name, double height, String period, String whereFound) {  
3 super(name, height, period, whereFound);  
4 }  
5 }  
61 public class Parasaurolophus extends Ornithopods {  
2 public Parasaurolophus() {  
3 super("Parasaurolophus", 11.0, "Late Cretaceous", "Canada, USA");  
4 }  
5 }

1 public class Staurikosaurus extends EarlyDinosaurs {  
2 public Staurikosaurus() {  
3 super("Staurikosaurus", 2.0, "Late Triassuc", "Brazil");  
4 }  
5 }  
61 public class Stenopelix extends Ceratopsians {  
2 public Stenopelix() {  
3 super("Stenopelix", 1.5, "Early Cretaceous", "Germany");  
4 }  
5 }  
6  
61 public class Thyrannosaurus extends Theropods {  
2 public Thyrannosaurus() {  
3 super("Thyrannosaurus", 12.0, "Late Cretaceous", "Canada, USA", 7000);  
4 }  
5 }  
61 public class Triceratops extends Ceratopsians {  
2 public Triceratops() {  
3 super("Triceratops", 9.0, "Late Cretaeous", "USA");  
4 }  
5 }  
6