Suppose we wanted the louder of two dogs to bark. Fill in the blanks to complete the **barkIfLouder** method. * 4 points

```
public class Dog implements Comparable<Dog> {
  private int barkVolume;

  ...

  public void bark() { ... }

  public int compareTo(Dog other) {
    return this.barkVolume - other.barkVolume;
  }

  // Allow the louder of two dogs to bark (ties can be broken either way)
  public static void barkIfLouder(Dog d1, Dog d2) {
    if (__1__.__2__(d2) __3__ __4__) {
      d1.bark();
    } else {
      d2.bark();
    }
  }
}
```

	d1	d2	compare	compareTo	<	>	0	1
Blank 1	○	○	○	○	○	○	○	○
Blank 2	○	○	○	○	○	○	○	○
Blank 3	○	○	○	○	○	○	○	○
Blank 4	○	○	○	○	○	○	○	○

Maya is writing a Boba class. Select all reasons that she might want to use a Comparator over Comparable. * 2 points

Select all that apply.

- [ ] If she makes Boba implement Comparator, she won't need to create any additional classes and can just override the compareTo method.
- [ ] If she has multiple ways of comparing two Boba objects, she can define a separate Comparator for each way.
- [ ] If she wants to keep her Boba class relatively short, using a Comparator allows her to move the comparison logic to a separate class.
- [ ] The order of inputs to a Comparator's compare function don't affect the output, so she doesn't have to be as precise when calling compare.

Michael wrote the following code to identify the biggest shoe in his shoe collection. However, when he tries to compile his code, it fails. Why couldn't it compile? * 2 points

Select all that apply.

```
public class Shoe {
  public double size;

  public Shoe(double s) { size = s; }

  public int compareTo(Shoe other) {
    return this.size - other.size;
  }

  public static void main(String[] args) {
    List<Shoe> shoes = new ArrayList<>();
    shoes.add(new Shoe(11));
    shoes.add(new Shoe(12.5));
    Shoe largest = Collections.max(shoes);
  }
}
```

- [ ] The compareTo method requires two inputs instead of one, so that it can compare the two objects
- [ ] He forgot to make his Shoe class implement Comparable<Shoe>, so Collections is not aware that his Shoe class has a compareTo method.
- [ ] Collections is part of the standard Java library and can only accept Objects, so passing in a list of Shoes is too specific.
- [ ] The return value of his compareTo should be a boolean, returning true if the instance is greater than the object passed in

A copy of your responses will be emailed to yiyunchen@berkeley.edu.

Submit Clear form