# EXERCISES TO COMPLETE

## Find the best phones

You are looking for the 'best' smartphone out of a list of thousands of phones.

To begin this mini project, you must download the find-best-phones skeleton code.

This download contains four Java classes:

- **Phone** — a model class representing a smartphone. A phone has a model name, a screen size, and a battery capacity.
- **PhoneList** — a model class representing a list of phones. There is an **addPhone** method to add a phone to the list, a **getAllPhones** to get a list of all phones, and a **getBestPhones** method to get a list of just the 'best' phones.
- **PhoneParser** — a helper class which can parse phone data from a string, and load all the phone data from a text file.
- **FindBestPhones** — the main program class, which will print the model names of all the 'best' phones.

Additionally, there are two text files named **phone-data.txt** and **phone-data-short.txt**, containing data for many different phones.

To determine the 'best' phones, we will compare phones by their screen size and battery capacity. We say that phone X "dominates" phone Y if it is better in one criterion and at least as good in the other criterion; for example, it has a bigger screen and the same battery capacity (or more).

A phone is 'best' if it is not dominated by any other phone. The 'best' phones in the phone-data-short.txt file are:

```
Samsung Galaxy S8 Plus
Huawei Mate 10 Pro
```

Once you have finished this mini project, your program should print these two model names when you test it with the "short" data file.

## Completing the model classes

The model classes **Phone** and **PhoneList** are almost completely given, but there are some details you should complete.

**Task 1:** ensure the screen size and battery capacity are positive values by throwing an **IllegalArgumentException** when the phone object is created.

**Task 2:** implement the **dominates** method in the **Phone** class. You should use the rule above to determine if a phone "dominates" another phone.

**Task 3:** change the **getAllPhones** and **getBestPhones** methods in the **PhoneList** class so they return **unmodifiable views** of the collections they return.

### Parsing the phone data

The phone data in the text files is in the following format:

```
model screenSize batteryCapacity
```

In particular, each line consists of a model name, then a space, then its screen size, then a space, then its battery capacity. The model names are encoded with underscores _ instead of spaces.

**Task 4:** implement the **parse** method in the **PhoneParser** class. This method takes the phone data as a string; it should extract the relevant data from the string, and return a Phone object with this data.

You may use any parsing technique for this task; string manipulation, a regex, or a **Scanner**. Don't forget to replace the underscores with spaces in the model name!

**Task 5:** implement the **parseFile** method in the **PhoneParser** class. This method takes a filename as a string; it should:

- Create an empty **PhoneList**.
- Create a **BufferedReader** to read the lines of the text file.
- Parse each line using the **parse** method to create a **Phone**, and add it to the list.

### The main program

The main program class **FindBestPhones** will make use of the other classes, to print the model names of the 'best' phones.

**Task 6:** implement the **main** method. To do this, you will need to:

- Use the **parseFile** method to get the phone data from the file.
- Print the model names of the 'best' phones, using the **getBestPhones** method.
- Handle any I/O failures which might occur in the **parseFile** method.