# Practice 6

Deadline: 2 weeks from now. Should be checked onsite (during labs).

In this practice, we'll be using Java's I/O utilities to explore the source code and byte code of Java core APIs.

## Java Source Code

The `src.zip` in your `$JAVA_HOME` (`$JAVA_HOME` refers to JDK installation directory) contains the Java programming language source files for all classes that make up the Java Core API (that is, sources files for the `java.*`, `javax.*` and some `org.*` packages, ). This source code is provided for informational purposes only, to help developers learn and use the Java programming language.

We provided the `src.zip` in JDK 1.8.0. Please write a program to:

- Read the `src.zip` file.
- Count and print all the `.java` source files in the `java.io` and `java.nio` directories.

Sample output:

```
In .zip: # of .java files in java.io/java.nio packages: 317
java/io/Bits.java
java/io/BufferedInputStream.java
java/io/BufferedOutputStream.java
java/io/BufferedReader.java
java/io/BufferedWriter.java
java/io/ByteArrayInputStream.java
......
```

## Java Byte Code

The `rt.jar` file in your `$JAVA_HOME/jre/lib` contains all of the compiled class files for Java Core API. JRE provides `rt.jar` as bootstrap classes to be loaded when JVM starts, so that you could use core APIs such as `java.lang.String`, `java.util.ArrayList` and `java.io.InputStream`. A `.jar` file is essentially a zip file for `.class` files and you could use zip tools such as WinRAR to explore its content.

We provided the corresponding `rt.jar` in JDK 1.8.0. Please write a program to:

- Read the `rt.jar` file.
- Count and print all the `.class` bytecode files in the `java.io` and `java.nio` packages.

Sample output:

```
In .jar: # of .class files in java.io/java.nio packages: 415
java/nio/Buffer.class
java/nio/ByteBuffer.class
java/nio/HeapByteBuffer.class
```

```
java/nio/Bits.class
java/nio/ByteOrder.class
java/nio/Bits$1.class
java/nio/CharBuffer.class
......
```

## Compare the count

The `src.zip` and `rt.jar` we provided are from the same JDK installation. Suppose that one `.java` is compiled to one corresponding `.class`, then the count of `.java` source files in `java.io` and `java.nio` packages should be the same as the count of `.class` files in these two packages. But the two counts are different.

One of the reason is the existence of *inner (annoymous) classes*. If a class contains an inner class, then the compiled `.class` file will be `ClassName.class` and `ClassName$InnerClassName.class`. In case of anonymous inner classes, the compiled `.class` file will be `ClassName.class` and `ClassName$1.class`.

Now, let's count `.class` related to inner classes and remove them from the total count. Then, let's identify the number of `.java` and `.class` with the same fully-qualified names.

```
# of .class files for inner classes: 101
# of .java files with corresponding .class: 313
```

Finally, let's identify `.java` files without any corresponding `.class` files, and `.class` files without any corresponding `.java` files. Please inspect the code by yourself to find out the reasons.

```
# of .java without its .class: 4
java/nio/file/attribute/package-info
java/nio/file/spi/package-info
java/nio/file/package-info
java/nio/channels/package-info

# of .class without its .java: 1
java/io/FilePermissionCollection
```

## Evaluation

The practice will be checked by teachers or SAs. What will be tested:

1. That you understand every line of your own code, not just copy from somewhere
2. That your program compiles correctly (javac)
3. Correctness of the program logic
4. That the result is obtained in a reasonable time

Late submissions after the deadline will incur a 20% penalty, meaning that you can only get 80% of this practice's score.