This exercise gives you practice with data type expressions, including tuples, options, and recursive types, and how they are represented in Java. For example, here is a type expression and its corresponding Java class:

Rat = BigInt x BigInt

```java
class Rat {
    BigInt numerator;
    BigInt denominator;
}
```

**Write Java classes and/or interfaces that implement each of the following type expressions.** (Note that Java provides the primitive data types like int and double, as well as the Date, String, and List data types, so you don’t need to define those. Also, some field name suggestions are given in italics, to help understand the types. Finally, for simplicity, show only fields, not methods, and omit access modifiers like public and private.)

1. URL = Http(String host x int port x String path) U Https(String x int x String) U File(String path)

2. FileEntry = Date x String name x Contents
   Contents = Folder(List<FileEntry>) U PlainFile(byte*)

3. Account = Checking(double balance) U Savings(double) U Joint(Account, Account)

**Write type expressions for each of the following Java implementations.** (Note that Runnable is another built-in Java data type.)

4. class Person {
   String name;
   int age;
   Date birth;
}

5. interface Jobs {}
   class SomeJobs implements Jobs {
       Runnable job;
       Date deadline;
       Jobs moreJobs;
   }
   class NoJobs implements Jobs {
   }

6. class LoginAccount {
   String username;  
   Authenticator auth;
}
   interface Authenticator {}  
   class Password implements Authenticator {
       String password;
   }
   class Certificate implements Authenticator {
       byte[] certificate;
   }
   class SameAs implements Authenticator {
       LoginAccount account;
   }