JUnit test Report for Assignment 2

1. **Class Tested:**
   DropboxFileEventHandler.java
   FileEvent.java
   FileState.java
   FileStates.java
   FileReactor.java
   DefaultFileManager.java
   DropboxCmd.java
   DropboxCmdProcessor.java
   DropboxProtocol.java

2. **Path for Test cases:**
   Org.cs27x.test.dropbox, org.cs27x.test.filewatcher and org.cs27x.test.FileReactor

3. **Class refactored**
   a. DropboxFileEventHandler.java
   As in the method handle (FileEvent evt), many methods in other class have been used. Whenever there is an error, we cannot see which step makes it. Thus in order to be easy to test, the method handle (FileEvent evt) has been divided into three methods: newFileEventAfterResolve (FileEvent evt), newFileEventAfterFilter (FileEvent evt) and finalHandle (FileEvent evt), and are tested separately.

   b. FileReactor.java
   There are many private methods and inner class in this class. A stub class has been made – FileReactorStub.java to help test. In the FileReactorStub.java, a private variable of WatchEvent and a public method of getWatchEvent() have been added, and also a singleWatch() method which can only detect the first behavior in the file system. Then by using the getWatchEvent () method, we can audit and verify it.
   When running the test case of this class. Two classes should be run at the same time. GoWithTestFileReactor1.java and GoWithTestFileReactor2.java are responsible to make some file behaviors, such as create, modify and delete file and directory in the root directory.

4. **Flaw tested:**
   a. Original method: Write (Path p, byte[] data, boolean overwrite) method in DefaultFileManager.java
   Test case: testWrite () in TestDefaultFileManager.java
   Flaw: Error caused by null data.
   Add if (data != null) in the original method

b. Original method: updateFileState (DropboxCmd cmd, Path resolved) method and cmdReceived (DropboxCmd cmd) in DropboxCmdProcessor.java
   Test case: testAddUpdateMock() and testCmdReceived() method in TestDropboxCmdProcessor.java
   Flaw: Cannot handle if the file is not existed.

c. Original method: addFile(Path p) and updateFile(Path p) in DropboxProtocol.java
   Test case: testAddFile() and testUpdateFile() in TestDropboxProtocol.java
   Flaw: error when file is not existed or it is a directory
   Modify: add if(!Files.isDirectory(p) && Files.exists(p))

d. Original method: watchLoop() in FileReactor.java
   Test case: testWatchServiceCreateF() in createFile() in testFileReactor2.java and GoWithTestFileReactor2.java
   Flaw: error when create a file in a directory of the root directory
   Modify: The reason is that the WatchService only watch the change of the root directory. Whenever a new directory is created, it should be resisted in the current WatchService as well.

e. Original method: handle() and newFileEventAfterResolve() in DropboxFileEventHandler.java
   Test case: testNewFileEventAfterResolve() in TestDropboxFileEventHandler.java
   Flaw: when the file is in a directory of the root directory, the resolved file is not the same as the absolute path.
   Modify: the reason is that in the DropboxFileEventHandler.java, the file path is modified to the file name and then use a file manager to resolve the path, and the file manager only use the root directory.

f. Original method: getState(Path p), getOrCreateState(Path p) and insert(Path p) in FileStates.java
   Test case: testGetState(), testGetOrCreateState() and testInsert() in testFileStates.java
   Flaw: error when path is null in three methods, and error when insert a directory or a non-exist path
   Modify: add if(p==null) return null, add if(!Files.exists(p)) return null in insert method, and check if it is a directory and if it is, iterate the directory to compute the size of the file.