Assignment 2

The goal of this assignment is to create, install, and test the advisor plugin.

Background

The advisor is a plugin that runs on the client side. Charlie consults the advisor, if one is installed and enabled, prior to each play on behalf of the user.

The advisor implements *IAdvisor* in the *charlie.plugin* package. The interface has exactly one specified behavior:

```
public Play advise(Hand myHand, Card upCard)
```

The parameter, *myHand*, is the human player's hand and *upCard* is the dealer's up-card. The return value, Play, is an enumerated type in the *charlie.util* package. (Note: there's class named *Play* in the *charlie.message.view.to* package. This class is used to tell *IPlayer* instances it's their turn and so don't confuse this with the enumerate type, *Play*.)

The valid plays are NONE, HIT, STAY, DOUBLE_DOWN, and SPLIT. NONE means "no advice." SPLIT is not supported by the Charlie view but is still a valid play which forces the user to choose hit, stay, or double-down.

Charlie guarantees that when *advise* is invoked:

A) *Hand* will have at least two cards
B) the hand's value is < 21

The advisor should not alter the hand. It should instead only invoke the following methods

Method	Semantics
size	To get the number of cards in Hand
getCard	To get cards from the hand where the index is zero-based.
getValue	To get the hand's value

Implement the full 468 cell Basic Strategy in Attachment 1. Implement the rules so that they can be reused.

For instance, create a class, *BasicStrategy*, that has a method, *getPlay*, which takes two parameters, *Hand* and *Card*, and returns a *Play* instance--modeled on *advise*. However, *getPlay* exploits the observation that the Basic Strategy is composed of four distinct processing sections which could be written as four distinct methods.

Tasks

Part I - Get the IAdvisor plugin working.

1. Start NetBeans and open the Charlie project and the MyCharliePlugins project.

2. In MyCharliePlugins, right-click on `charlie.client` package > New > Java Class… and give it the name, `Advisor`.

3. Write the advisor by implementing `IAdvisor` which is in the `charlie.plugin` package of the Charlie project.

   Initially, for the `advise` method, return `Play.NONE`. You'll extend this later.

4. Open the Charlie properties file, `charlie.props`, and add the property

   ```
   charlie.advisor charlie.client.Advisor
   ```

5. Right click on MyCharliePlugins | Clean and Build

6. Right click on Charlie | Properties | Libraries | Add Jar/Folder and browse to the dist folder of MyCharliePlugins and add its jar. (Note: you did this in Assignment 1 and should not need to be done here.)

7. Right click on Charlie | Clean and Build.

8. Start the Charlie server and client.

9. After login, if everything goes well, you'll see the advisor checkbox enabled.

10. Check the "Advisor" box and playtest the advisor.

    Complete the advise method to implement the complete Basic Strategy in Appendix 1.

Part II - Write the unit test cases.

1. Create 16 unit test cases in the MyCharliePlugins project.

   To create a test case, right click on MyCharliePlugins | New | Unit Tests | JUnit Test… and if prompted, choose JUnit 4.x. This creates a test case in the **Test Packages** folder in the default package.

   Create a package in **Test Packages** with the name `charlie.bs.section`*Y* where *Y* is the Basic Strategy section number. Move your test case into this package.

   Name the test files according to your hand and the up card. For instance, `charlie.bs.section1` you'll have two JUnit programs, `Test00_12_2.java` and `Test01_12_2.java` that test player 7+5 and 8+4 vs. dealer 2, respectively.

See `Test00_12_2.java` which is attached as an example of a working test case.

See the testing grid in Appendix 2.

2. Create a test suite and add all 16 test cases to the suite.

3. Run the test suite and verify that all test cases pass.

4. Compress Charlie and MyCharliePlugins projects, redeploy them on a different host, and run the test suite on that machine.

Deliverables

1. Compress ONLY the MyCharliePlugins folder into a .zip file. (Don't use .rar, .7zip, or other such compressed formats.) Rename it to <your-name>`-MyCharliePlugins.zip` where <your-name> is your name or the team member names.

2. Upload the *.zip* into the assignment shell by the due date. Please only upload one file per team. Do not send the Charlie project.

Evaluation

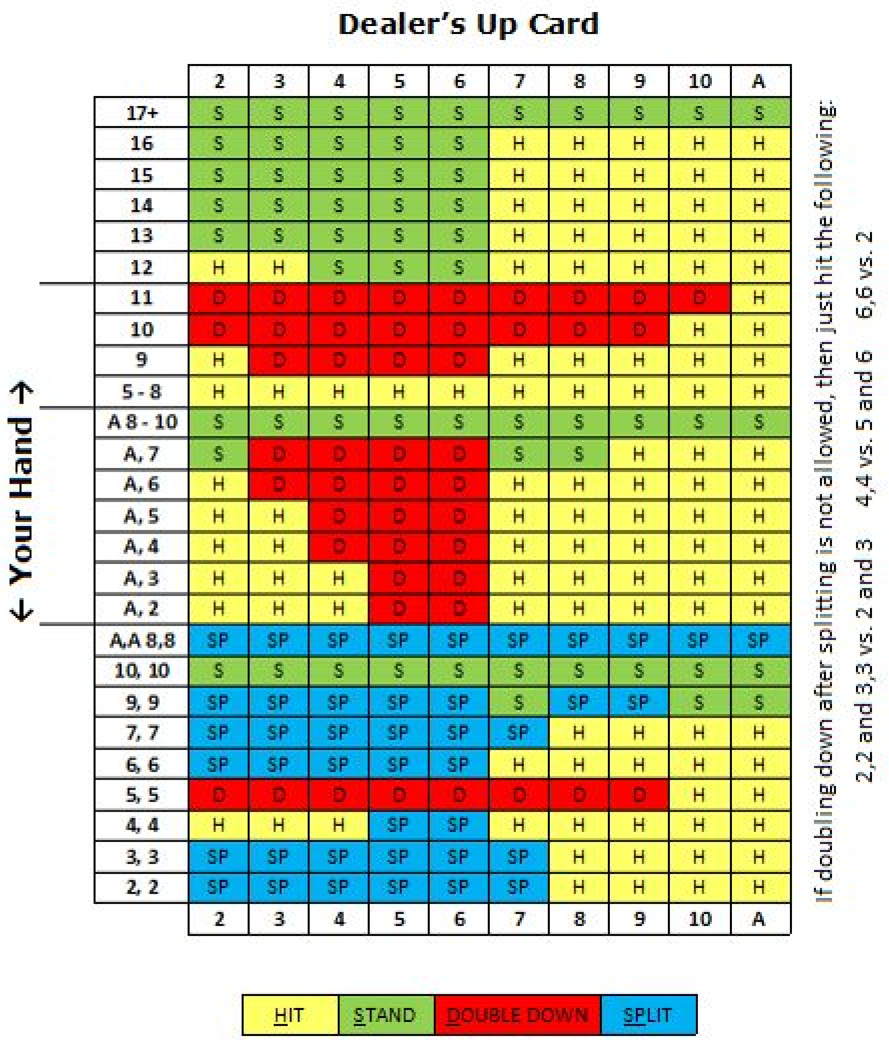
I will evaluate the assignment as follows:

Table 1

Criteria	Points
16 test cases successful	100
Design	30
Project delivered as specified	15
Good programming style	5
Total	150

The project will be penalized 10% for being late and 10% if I return the project with compiler errors, link errors, request for resubmission, etc.

Attachment 1



Dealer's Up Card

Your Hand	2	3	4	5	6	7	8	9	10	A
17+	S	S	S	S	S	S	S	S	S	S
16	S	S	S	S	S	H	H	H	H	H
15	S	S	S	S	S	H	H	H	H	H
14	S	S	S	S	S	H	H	H	H	H
13	S	S	S	S	S	H	H	H	H	H
12	H	H	S	S	S	H	H	H	H	H
11	D	D	D	D	D	D	D	D	D	H
10	D	D	D	D	D	D	D	D	H	H
9	H	D	D	D	D	H	H	H	H	H
5 - 8	H	H	H	H	H	H	H	H	H	H
A 8 - 10	S	S	S	S	S	S	S	S	S	S
A, 7	S	D	D	D	D	S	S	H	H	H
A, 6	H	D	D	D	D	H	H	H	H	H
A, 5	H	H	D	D	D	H	H	H	H	H
A, 4	H	H	D	D	D	H	H	H	H	H
A, 3	H	H	H	D	D	H	H	H	H	H
A, 2	H	H	H	D	D	H	H	H	H	H
A,A 8,8	SP	SP	SP	SP	SP	SP	SP	SP	SP	SP
10, 10	S	S	S	S	S	S	S	S	S	S
9, 9	SP	SP	SP	SP	SP	S	SP	SP	S	S
7, 7	SP	SP	SP	SP	SP	SP	H	H	H	H
6, 6	SP	SP	SP	SP	SP	H	H	H	H	H
5, 5	D	D	D	D	D	D	D	D	H	H
4, 4	H	H	H	SP	SP	H	H	H	H	H
3, 3	SP	SP	SP	SP	SP	SP	H	H	H	H
2, 2	SP	SP	SP	SP	SP	SP	H	H	H	H
	2	3	4	5	6	7	8	9	10	A

If doubling down after splitting is not allowed, then just hit the following:

2,2 and 3,3 vs. 2 and 3   4,4 vs. 5 and 6   6,6 vs. 2

🟨 HIT	🟩 STAND	🟥 DOUBLE DOWN	🟦 SPLIT

Attachment 2

Table 2 Test grid, shaded cells are the test filename suffixes.

Section	Your hand ↓ / Up card →	2 - 6	7 - A
1	12 - 17+	12__2	12__7
2	5 - 11	5__2	5__7
3	A,2 - 10	A2__2	A2__7
4	2,2 - A,A	22__2	22_7