Homework 4

In this homework we will build a simple programmable remote control that implements the Singleton and uses an abstract class and an interface. You have been given the file Test.java with a main function and the file output.txt which contains the expected program output for both parts of the assignment. You should not change Test.java except as indicated in comments in that file.

Part 1:

This will be programmed using an abstract class.

You will have a **Remote** class that has the following public functions:

A function **buildRemote( )** that build a remote with 5 slots that hold Command references if no Remote object has been created, and otherwise returns the previously created remote object. An array of command references of length 5 can be created by:

```
private Command[ ] slots;

slots = new Command[5];
```

A function **addCommand(int s, Command command)** that adds the command command to slot s of the remote.

A function **removeCommand(int s)** that removes the command in slot s. Check the output from Test to see what happens if an attempt to remove a command from a slot with no command, or to remove a command from a non-existent slot.

A function **executeCommand(int s)** that calls the execute( ) method on the object pointed to the by reference in that slot. See the output to see what happens if the slot is null (contains no command) or if s is out of bounds for the slots.

A **toString( )** function that prints the remote and its commands.

You will need an abstract **Command** class.

You will need the following concrete classes:

Turndown whose **execute( )** method prints “Volume turned down to “ and the volume that it is turned down to. The volume goes down by 1 at each call.

Turnup whose **execute( )** method prints “Volume turned up to “ and the volume that it is turned up to. The volume goes up by 1 at each call.

Turnon whose **execute( )** method prints “Turning on the TV”

Turnoff whose **execute( )** method prints “Turning off the TV”

I had an additional class that Turndown and Turnoff extended that contained a shared int that held the volume.

Part 2:

Like Part 1 except instead of an abstract class use an interface. This should require changing about 5 to 10 lines max in the previous program.

What to turn in:

A directory *userid* that contains subdirectories **1** and **2**. The subdirectories should contain your .java from Part 1 and Part 2, respectively. The graders should be able to descend into either 1 or 2 and type "javac Test.java; java Test" and execute your program.