

CS 61B Lab 6
October 10-11, 2006

Goal: This lab will introduce you to Java's built-in facilities for exception handling, and show you why they might be a good idea.

Copy the Lab 6 directory by doing the following, starting from your home directory. Don't forget the "-r" switch in the cp command.

```
mkdir lab6
cd lab6
cp -r $master/lab/lab6/* .
```

In this lab, you will be modifying a banking application. BankApp is a command-line driven program that allows you to create and manipulate customer bank accounts.

Part I: Compile It (1 point)

Compile BankApp.java and all the classes it depends on.

```
javac -g BankApp.java
```

Fix the compile-time errors by adding a try/catch statement in the appropriate place(s). (Don't add a "throws" clause to the offending method; use a try/catch.)

Once you have the program up and running, experiment with it to see how it works. What happens if you try to deposit a negative amount of money? What happens if you try to deposit to or withdraw from a non-existent account?

There is a subtle bug in the withdraw operation. What is it?
(Hint: Under what `_condition_` can you foil `VirtualTeller.withdraw()`?)

Excepting the withdraw bug, the program handles errors correctly, but does so rather clumsily in the code. For instance, when you try to deposit to a nonexistent account, two error messages are printed. (Why?) Error-handling is not done well.

Part II: Throwing Exceptions (2 points)

An exception class called `BadAccountException` has been created for your use; it should be thrown whenever an account cannot be found. Add the appropriate "throw" statement to `VirtualTeller.java`. (You should only need one.) Next, eliminate all the print statements generated when an account is not found. Add an appropriate "throws" clause wherever necessary. Be sure to catch the `BadAccountException` in `main()`; errors should not cause the application to halt.

Is your modified code for `VirtualTeller` easier to read? Have you eliminated the problem of double error messages? (If not, you put the "throw" statement in the wrong place.)

PART III: A New Exception (1 point)

The withdraw method still has a bug, and when you try to deposit an invalid (e.g. negative) amount, a nonsensical "New balance" statement is printed. Fix these problems by creating a `BadTransactionException` class, and adding the appropriate "throw" statements and "throws" clauses to `VirtualTeller.java`. Again, make sure the exception is caught in `main()`. Note that you will have to recompile `AccountData` explicitly.

EXTRA: Slightly different behavior (optional)

Observe that when an exception occurs during a transaction, you are prompted for a new command. Suppose you want slightly different behavior: if an exception occurs during a command, then the command is automatically repeated so you don't have to retype it. How can you do this by moving ONE line of code in `BankApp.java`?

Check-off

Demonstrate the application for your TA or Lab Assistant. Show her/him the changes in your code.

- 1 point: Run your program and show that it behaves well if an invalid command is entered.
- 2 points: Show that your program behaves well when you try to deposit to a nonexistent account. Explain which method you put the "throw" statement in.
- 1 point: Show what happens when you try to deposit or withdraw an invalid amount.