CS 1400-03 Introduction to Programming and Problem Solving
Coding Practice #12
(Due: 11:59 PM, Friday, 5/7/2021)

Except Coding Practice #1, I will not grade your coding practice submissions. Instead, they will be treated as participation points. On blackboard, you will receive full points as long as you work on the exercises, which don't necessary mean they are all correct. Please check your own programs carefully and make sure they do generate the desired output.

Objectives:

- Be able to create custom exception classes
- Be able to throw exceptions manually, detect and handle exceptions
- Be able to test and debug a program

Change your working directory to `cs1400/codingPractice` for this assignment.

Task #1 Bank Account Exceptions

(a) We can create our own exception classes that represent error conditions in a bank account application. Refer to the power point notes for Chapter 14, slide 31, the `NegativeStartingBalanceException` class has already been given. Write three more exception classes `NegativeDepositException`, `NegativeWithdrawException`, and `WithdrawExceedBalanceException` for the following error conditions:

- A negative number is passed to the deposit method.
- A negative number is passed to the withdraw method.
- The amount passed to the withdraw method exceeds the account's balance.

(b) Modify the `BankAccount` class so that it throws the appropriate exception when any of these errors occurs.

(c) Modify the driver program `AccountTest.java` to include appropriate `catch` clauses for all above defined exceptions and test **all features** and methods described above.

Task #2 Month Exceptions

Design a `Month` class that holds information about the month. The class should have an `int` field named `monthNumber` that holds the number of the month. For example, January would be 1, February would be 2, and so forth. In addition, provide the following methods:

- A constructor that accepts the number of the month as an argument. It should set the `monthNumber` field to the value passed as the argument. If a value less than 1 or greater than 12 is passed, the class should throw `InvalidMonthNumberException`.
- A constructor that accepts the name of the month, such as "January" or "February" as an argument. It should set the `monthNumber` field to the correct corresponding value. If an invalid string is given, the class should throw `InvalidMonthNameException`.
- A `toString` method that returns the name of the month. For example, if the `monthNumber` field contains 1, then this method should return "January".

(a) Write the classes `Month`, `InvalidMonthNumberException`, and `InvalidMonthNameException`.

(b) Write a driver program called `MonthTest.java` to demonstrate your classes and exception handlers. For example, if the following four objects were created,

```
Month m1 = new Month(10);
Month m2 = new Month(25);
Month m3 = new Month("March");
Month m4 = new Month("Septober");
```

Your program should output:

```
The month you just created is October
Error - Invalid number given for the month: 25
The month you just created is March
Error - Invalid name given for the month: Septober
```

Submission:
Generate a `script` file `practice12.txt` with appropriate time stamps and the following steps visible:

1) a `pwd` to show the current working directory
2) a `ls -l` to show in long format the files in your `cs1400/codingPractice` directory
3) display `NegativeStartingBalanceException.java`, `NegativeDepositException.java`, `NegativeWithdrawException.java`, `WithdrawExceedBalanceException.java`, `BankAccount.java`, and `AccountTest.java`
4) compile `AccountTest.java`
5) run `AccountTest`
6) display `InvalidMonthNumberException.java`, `InvalidMonthNameException.java`, `Month.java`, and `MonthTest.java`
7) compile `MonthTest.java`
8) run `MonthTest`

Submit the script file `practice12.txt` on Bb, under the Coding Practice Folder, Practice #12 link.