[**Department of Computer Science**](http://www.cs.umd.edu/)

[**CMSC131**](http://www.cs.umd.edu/class/fall2017/cmsc131-FC01) Fall 2017

**Project:** Money Identification Program

**Due Date:** Thursday, Sep 21, 8:00 pm

## Overview

You will write a very simple application that asks the user to identify which person appears on which denomination of U.S. currency (or vice versa). It's not the most exciting project, but we have to start somewhere!

## Objectives

This project will allow you to practice variables, strings, input/output facilities, conditional statements, logical operators, and the Eclipse IDE.

## Grading

* (90%) Release Test (7)
* (10%) Programming Style

## Clarifications

## Code Distribution

The project's code distribution is available by checking out the project named **MoneyIdProg**. The code distribution provides you with the following:

* A file named **OnTheMoney.java** - You will need to modify this file.

## Specifications

Your program knows three things: Washington appears on the $1 bill, Hamilton appears on the $10 bill, and Franklin appears on the $100 bill. That's it.

The program begins by prompting the user with:

Type 1 to enter a denomination, 2 to enter a last name:

The user will enter an integer in reply to your inquiry. The user should enter either 1 or 2.

If neither a 1 nor a 2 is entered by the user, the program should terminate after displaying the message "Invalid Menu Option".

## Processing an Entry of "1"

The program prompts the user with:

Choose a denomination:

The user will type an integer as a reply to this prompt. The user should enter either "1", "10", or "100". If any other number is entered, the program will print out "Invalid choice!" and will terminate with no further output. You may assume at this point that the user will enter a number (not a string).

Assuming that the user has entered one of the three valid denominations, the program prompts the user with:

Which person appears on the n dollar bill?

[In the prompt above, the "n" must actually be either 1, 10, or 100, depending on what the user has selected.]

The user will enter a String in reply to this prompt. If the name entered represents the correct person for this denomination, the program will output "Correct!" and terminate. But if the user has entered the wrong name, then the program will output "Incorrect!" and terminate. The name must be spelled correctly and have only the first character capitalized to be correct.

## Processing an Entry of "2"

The program prompts the user with:

Choose a name:

The user will type some string of characters at this point. The user should enter either "Washington", "Hamilton", or "Franklin". If any other string is entered, the program will print out "Invalid choice!" and will terminate with no further output. The name must be spelled and capitalized exactly as shown to be considered a valid entry.

Assuming that the user has entered one of the three valid names, the program prompts the user with:

Which denomination does name appear on?

[In the prompt above, "name" must actually be either Washington, Hamilton, or Franklin, depending on what the user typed as the previous response.]

The user will enter an integer in reply to this prompt. If the value entered represents the correct denomination for the name given, the program will output "Correct!" and terminate. But if the user has entered the value that is not the denomination corresponding to the name typed previously, then the program will output "Incorrect!" and terminate.

## Requirements

* Verify that your project passes the submit server test ([https://submit.cs.umd.edu/](https://submit.cs.umd.edu/fall2017/)).
* You have three tokens for this project in the submit server.
* Your program must use the single class OnTheMoney.
* You must use "named constants" (also called "symbolic constants") for any literal values that will not change during program execution.
* You must use meaningful variable names and good indentation.
* Input and output operations must be implemented as demonstrated in class.  (I.e. use the *Scanner* class for input, and use *System.out.print* or *System.out.println* for output.)
* You do not need to worry about the user entering the wrong types of values.  If an integer is expected, assume the user enters an integer; if a string is expected, assume the user enters a string.
* You should not get creative and modify the project specifications. **For example, do not add extra behavior or output.**
* We use an automated system to grade projects and you will not pass our tests if you do not follow the project specifications precisely. **In particular, you must spell things exactly as we have spelled them and use punctuation exactly as we have.** One missing or incorrect punctuation mark (a period, exclamation point, comma, or colon) could cause you to fail all of the tests.  If you spell something wrong ("denomniation" or "Hammillton" for example) you will fail many of our tests.  PROOFREAD CAREFULLY.  You do NOT need to worry about how many spaces to put between words in your output -- it doesn't matter.  Also, whether or not you capitalize letters in your output doesn't matter.
* Just in case you know what an array is -- you may not use arrays (or Java collections of any kind) while implementing this project.
* Your program must terminate at the end of the main method. It must terminate without using anything like return or *System.exit().  (*Don't worry if you don't know what those are.)
* Be sure to only use ONE Scanner for your project.  If you use more than one Scanner to read from the input stream, you will fail our tests.  (In other words, your program should only contain the expression "new Scanner(...)" in ONE place.)
* Follow the style guide provided at [Style Guide](http://www.cs.umd.edu/class/fall2017/cmsc131-FC01/content/resources/StyleGuidelines.html).

## Sample Runs

The image below illustrates show how your program should behave. Note that items in green represent the things that are entered by the user. Keep in mind these are just examples and not the only scenarios that your program is expected to handle.

## Submission

Submit your project from Eclipse (within Java perspective) by right-clicking the project folder and selecting "submit" . After you have submitted your project, you should visit the submit server. There you can obtain feedback about how well your project is performing. The number of times you can run our tests on your project (before the due date) is limited. The earlier you begin working on the project, the more opportunities you will have to see how your project performs on our tests before the due date!

## Academic Integrity

Please make sure you read the academic integrity section of the syllabus so you understand what is permissible in our programming projects. We want to remind you that we check your project against other students' projects and any case of academic dishonesty will be referred to the [Office of Student Conduct](http://www.jpo.umd.edu/).