Programming Exercise 02 - IO, Conditionals, Iteration

Problem Description

This assignment will test your understanding of input and output, conditionals, and iteration.

Solution Description

Class Standing Calculator

1. Create a class called StandingCalculator
2. Add to this class a main method
3. In the main method, do the following:
    1. Declare a variable called credits (this will always be a whole number)
    2. Declare a Scanner called myScanner that reads input from the console
    3. Print to the console “Credit hours taken:”
    4. Read a user inputted integer from the console and store it in credits. This should be on the same line as the query.
    5. Using a if/else conditional block and the below cutoffs, determine the class standing this equates to.
        - Freshmen have 0 to 29 hours (inclusive)
        - Sophomores have 30 to 59 hours (inclusive)
        - Juniors have 60 to 89 hours (inclusive)
        - Seniors have 90 or more hours
        - Negative numbers are invalid
    6. Print “Freshman”, “Sophomore”, “Junior”, or “Senior” depending on the number of hours inputted. If the number is negative, print “Invalid”. This should be printed on the line below the input and end in a new line character.

Day of the Week

1. Create a class called DayOfWeek
2. Add to this class a main method
3. In the main method, do the following:
    1. Declare a variable called dayNumber (this will always be a whole number)
    2. Declare a Scanner called myScanner that reads input from the console
    3. Print to the console “Days into the week:”
    4. Read a user inputted integer from the console (this should be on the same line as the query) and store it in dayNumber.
    5. Using a switch statement block and the below values, determine the day this equates to.
        - 1 is Sunday

    - 2 is Monday
    - 3 is Tuesday
    - 4 is Wednesday
    - 5 is Thursday
    - 6 is Friday
    - 7 is Saturday
    - Any other number is invalid
6. Print the day of the week depending on the number inputted. If the number is a valid day, this should print “The day is [day of week]”. If the number is invalid, print “Invalid”. This should be printed on the line below the input and end in a new line character.

Testing your app

In order to run your program, you must compile by running `javac FileName.java` in the same folder as `FileName.java`

When you run `java StandingCalculator`, your program should execute as follows:

```
java StandingCalculator
Credit hours taken:31
Sophomore
```

When you run `java DayOfWeek`, your program should execute as follows:

```
java DayOfWeek
Days into the week: 7
The day is Saturday
```

Allowed Imports

To prevent trivialization of the assignment, you can *only* import java.util.Scanner.

Feature Restrictions

There are a few features and methods in Java that overly simplify the concepts we are trying to teach or break our auto grader. For that reason, do not use any of the following in your final submission:

- var (the reserved keyword)
- System.exit

Collaboration

Collaboration Statement

To ensure that you acknowledge collaboration and give credit where credit is due, **we require that you place a collaboration statement as a comment**

at the top of at least one java file that you submit. That collaboration statement should say either:

I worked on the homework assignment alone, using only course materials.

or

In order to help learn course concepts, I worked on the homework with [give the names of the people you worked with], discussed homework topics and issues with [provide names of people], and/or consulted related material that can be found at [cite any other materials not provided as course materials for CS 1331 that assisted your learning].

Recall that comments are special lines in Java that begin with `//`.

Turn-In Procedure

Submission

To submit, upload the files listed below to the corresponding assignment on Gradescope:

- StandingCalculator.java
- DayOfWeek.java

Make sure you see the message stating "PE02 submitted successfully". From this point, Gradescope will run a basic autograder on your submission as discussed in the next section.

You can submit as many times as you want before the deadline, so feel free to resubmit as you make substantial progress on the homework. We will only grade your last submission: be sure to **submit every file each time you resubmit**.

Gradescope Autograder

For each submission, you will be able to see the results of a few basic test cases on your code. Each test typically corresponds to a rubric item, and the score returned represents the performance of your code on those rubric items only. If you fail a test, you can look at the output to determine what went wrong and resubmit once you have fixed the issue.

The Gradescope tests serve two main purposes:

1. Prevent upload mistakes (e.g. forgetting checkstyle, non-compiling code)
2. Provide basic formatting and usage validation

In other words, the test cases on Gradescope are by no means comprehensive. Be sure to thoroughly test your code by considering edge cases and writing your own test files. You also should avoid using Gradescope to compile, run, or checkstyle your code; you can do that locally on your machine.

Other portions of your assignment can also be graded by a TA once the submission deadline has passed, so the output on Gradescope may not necessarily reflect your grade for the assignment.

Important Notes (Don't Skip)

- Non-compiling files will receive a 0 for all associated rubric items
- Do not submit `.class` files.
- Test your code in addition to the basic checks on Gradescope
- Submit every file each time you resubmit
- Read the "Allowed Imports" and "Restricted Features" to avoid losing points
- Check on Piazza for all official clarifications