CIS280 Project #3 (Chapters 4 and 5)

Name(s):_____ Due: _____

Personalization and Commenting Requirements:

- ***All source code must contain a comment with the source file name, programmer(s) names(s) and date created***
- ***Commenting at top should include the purpose of the program***
- ***Appropriate, clear commenting must be used throughout the code; be professional and pay attention to spelling***
- ***All applications should start with an output to the screen that contains the name of the application and the programmer(s***)
- ***Name the programs as shown on this project sheet* Requirements**

1. *ValidateInput1.java* – create a GUI program that validates a user's input of a number. Utilize a WHILE loop to enforce data validation.
    a. Ask the user for a number in a dialog box (GUI) – inform them that only 101 through 199 is acceptable
    b. Only numbers that are 101 through 199 should be accepted, force the user to keep trying until they get it right
    c. Congratulate the user on making a good entry when they have done so, using a dialog box
2. *ValidateInput2.java* – modify your ValidateInput1.java to utilize a DO WHILE loop to exhibit the same behavior.
3. *ProcessGrades.java* -- specified at the end/back of this handout. Use the Scanner class to accept keyboard input; this is a console-based application.
4. *Student.java* and *DemoStudent.java* updates (copy these from Project 2 work into a new folder)
    a. Add a method to *Student.java* called *getLevel*. This method returns a String, depending on the number of credits:
        i. 0-5 credits – returns "Part-Time Student"
        ii. 6-8 credits – returns "Half-Time Student"
        iii. 9-11 credits – returns "Three-Quarter Student"
        iv. 12 or more credits – returns "Full-Time Student"
    b. Add code to *DemoStudent.java:*
        i. In the keyboard input section, do not let the user enter less than 0 or more than 20 for the number of credits. Be sure the user is informed that these are the limits.
        ii. Display the student's level for *firstStudent*, *secondStudent*, and *thirdStudent* by calling the *getLevel* method.
5. *WriteMyFile.java* and *ReadMyFile.java* - Write an application that creates a text file that consists of an inventory of items you would want with you if you were ever stranded on a desert island. Then write the application that reads and displays that data.

Submission Requirements

- Zip your .java source files into a folder labeled with your name(s) and project number. Submit this zipped folder in Blackboard. Please use the onboard zip utility in Windows rather than a third-party application.

--- over ---

ProcessGrades.java

Operation

- The user enters a numerical grade from 0 to 100 in console-based mode.
- The application displays the corresponding letter grade.
- The application prompts the user to continue.

Specifications

- The grading criteria is as follows:

```
A    90-100
B    80-89
C    70-79
D    60-69
E    <60
```

- The application should continue only if the user enters “y” or “Y” to continue.
- When the user chooses not to continue, display the number of passing grades and failing grades. A passing grade is anything over 59.
- Validate the data and do not allow entries lower than zero or higher than 100.

Example Console Output

```
Welcome to the Letter Grade Converter
by Your Name

Enter numerical grade: 40
Letter grade: E

Continue? (y/n): y

Enter numerical grade: 55
Letter grade: E

Continue? (y/n): y

Enter numerical grade: 65
Letter grade: D

Continue? (y/n): y

Enter numerical grade: 70
Letter grade: C

Continue? (y/n): y

Enter numerical grade: 89
Letter grade: B

Continue? (y/n): y

Enter numerical grade: 90
Letter grade: A

Continue? (y/n): n

The number of passing grades: 4
The number of failing grades: 2
```