CSC 230 Assignment 1

Q1 (Programming):
Textbook Ch 8, Page 506, Programming Challenges, 1. Rainfall Class
Estimated time: 1 hour

Q2 (Programming):
Textbook Ch 8, Page 507, Programming Challenges, 2. Payroll Class
Estimated time: 2 hours

Q3 (Programming):
Textbook Ch 8, Page 510, Programming Challenges, 12. 2D Array Operations
Estimated time: 3 hours

Submission instructions:
Please submit a paper copy and an electronic copy.

Paper copy:
- Please submit the paper copy at the beginning of the class.
- Provide (create) **test input data** to all programming questions, and capture the related outputs as screen captures (or output files).
- Print the program source code files, test input data and **the output screen captures (or output files)**. If no output screen capture (or output file) is submitted, it would be assumed that the related program does not compile. If the print-out is not readable, **no mark will be awarded**.
- Identify each assignment question by writing the question number at the top of each page.
- Add the following statement to the first page of your submission: "I have abided by the UNCG Academic Integrity Policy on this assignment". Please write your full name and sign next to the statement. If the statement or the signature is not found, **75% of the possible points will be deducted**.

Electronic copy:
- Please submit a **lastname_firstname_assignment01.zip** (or lastname_firstname_assignment01.rar) file through the Blackboard Digital Dropbox. This zip (or rar) file should contain all submission files.
- Put the answers of all written questions in a **lastname_firstname_assignment01.doc** file.
- For programming questions, you only need to submit the *.java files.
- Name each *.java file (and related java class) according to the question number (e.g. Q1_*.java).

Grading guidelines (programming questions):
Your programs will be judged on several criteria, which are shown below.
- Correctness (50%): Does the program compile correctly? Does the program do what it's supposed to do?
- Design (20%): Are operations broken down into classes and methods in a reasonable way?
- Style (10%): Is the program indented properly? Do variables have meaningful names?
- Robustness (10%): Does the program handle erroneous or unexpected input gracefully (test input data)?

- Documentation (10%): Do all program files begin with a comment that identifies the author, the contents, and the compiler used for that particular file? Are all the classes, methods and data fields clearly documented? Are unclear parts of code documented?

A program that does not compile will get at most **50% of the possible points**.

Assignment 1 Marking Notes

Please take note of the following when you program for assignment 1 (affects marking).
:: Read the textbook questions carefully ::

Q1. "…The program should have methods that return the following…"
In other words, the class should have (at least) 4 **methods** doing the mentioned tasks. – affecting "Design (20%)" marking criteria.

Q1. "..Demonstrate the class in a complete program.."
Use **another** program (class) to use this class (instantiate the object). – affecting "Correctness (50%)" marking criteria.

Q1. "… Input Validation:…"
Use **different test input data** to show this works for both **valid** and **invalid** inputs. – affecting "Robustness(10%)" marking criteria.

Q2. "… following arrays as fields…"
In other words, the class should have the 4 arrays as **fields**. – affecting "Design (20%)" marking criteria.

Q2. "… method that accepts an employee's identification number as an argument and returns … "
In other words, the class should have this **method**, taking **argument**, and **returns** the calculated value. – affecting "Design (20%)" and "Correctness (50%)" marking criteria.

Q2. "Demonstrate the class in a complete program"
Use **another** program (class) to use this class (instantiate the object). – affecting "Correctness (50%)" marking criteria.

Q2. "… Input Validation:…"
Use **different test input data** to show this works for both **valid** and **invalid** inputs. – affecting "Robustness(10%)" marking criteria.

Q3. "…The program should have methods that return the following…"
In other words, the class should have (at least) 6 **methods** doing the mentioned tasks. Note the **arguments** mentioned. – affecting "Correctness (50%)" and "Design (20%)" marking criteria.

Q3 "…Demonstrate each of the methods in this program…"
Use **this program** (class) to call the methods. – affecting "Correctness (50%)" marking criteria.