A9 (20 marks)

***Focus**: basics of Object Oriented Programming*

Q1. [10 marks] Creates a class named `Cuboid` to represent cuboid objects and contains:

- Three double attributes *l*, *w*, and *h* specifying the length, width and height of the cuboid.
- A String attribute *color* that specifies the color of the cuboid.
- A constructor (with 4 arguments) that creates a cuboid with specified values.
- A constructor (with no arguments) that sets *l*, *w*, and *h* to 1 and *color* to "white". This constructor should invoke the 4-argument constructor using `this`.
- Your program should have these methods:
  - Getter methods for all fields (e.g. `getColor()` which returns the color)
  - `getVolume()` : returns the cuboid volume which is $l.\ w.\ \mathrm{h}$
  - `getSurfaceArea()` : returns the surface area of the cuboid: $2(l.w + l.\mathrm{h} + w.\mathrm{h})$
  - `displayInfo()` : displays on the screen the color, dimensions, surface area, and volume of this cuboid.

Write a test program that creates two objects of the `Cuboid` class — first object should have default values and the second object must be green of length 8, width 3.5, and height 5.9. Print the dimensions, color, surface area, and volume of each object as shown in the sample run below.

Sample run

```
Cuboid 1
  Color: White
  Dimensions: 1.00 X 1.00 X 1.00
  Sruface Area: 6.00
  Volume: 1.00
Cuboid 2
  Color: Green
  Dimensions: 8.00 X 3.50 X 5.90
  Sruface Area: 191.70
  Volume: 165.20
```

Q2. [10 marks] Write a program that creates a class named `BankAccount` and contains:

- Private attributes:
  - `id` (`int`), `balance` (`double`), and `annualInterestRate` (`double`).
  - `count` (`static int`) to keep a record of the number of created objects.
- Constructors:
  - A 2-argument constructor that creates an account with given `annualInterestRate` and `balance`, increments `count` by 1, and then stores the new `count` into `id`.
  - A no-argument constructor that invokes the above 2-arg constructor and sets both `balance` and `annualInterestRate` to 0.
- Methods:
  - Getter methods for `balance`, `annualInterestRate`, and `id`.
  - Setter methods for `balance` and `annualInterestRate`.
  - `getMonthlyInterest()`: returns the monthly interest (not the interest rate). Monthly interest is `balance * annualInterestRate / 12`. Note that the interest rate is a percentage, e.g. 4.5%. You need to divide it by 100.
  - `withdraw(double amount)`: withdraws a specified amount from the account.
  - `deposit(double amount)`: deposits a specified amount to the account.
  - `displayInfo()`: displays the information of a bank account as shown in the sample run below.

Write a test program that creates an object of `BankAccount` with a balance of $33,000, and an annual interest rate of 6.7%. Use `withdraw` to withdraw $1,500, use `deposit` to deposit $1,000, then use `displayInfo` to display the account information.

Sample run

```
Account ID: 1
Current balance: $32500.0
Annual interest rate: 6.700 %
Monthly interest rate: 0.558 %
Monthly interest: $181.458
```

Submission Instructions

For this assignment, you need to do the following:

1. Create a Java project of which name consists of **your student number followed by the assignment number,** e.g., “1234567_A1”.
2. Create one class for each question and write your answer inside that class. Your classes should have the same name as the question number (e.g., Q1)
3. After solving all questions, open your file explorer.
4. Navigate to your Java project folder (can be found inside your Eclipse workspace folder).

5. Locate the “src” folder for this project (the folder that includes the source code for all questions).
6. Zip the “src” folder and rename the zipped file to match your project name (e.g., 1234567_A1.zip).
7. Submit the zipped file **to Canvas.**

Note that you can resubmit an assignment, but the new submission overwrites the old submission and receives a new timestamp.