Hands-on Activity
Blood Bank (Part 1)

Objective:

At the end of the activity, the students should be able to:

- Create and overload constructors.

Procedure:

1. Develop a simple program that stores a patient's blood details. Create two (2) classes named **BloodData** (no class modifier) and **RunBloodData** (public).
2. For the **BloodData** class: declare two (2) static **String** fields named **bloodType** for accepting O, A, B, and AB) and **rhFactor** (stands for Rhesus factor, an inherited protein found on the surface of red blood cells) for accepting **+** and **-**.
3. For the default constructor (public) of the **BloodData** class, set **bloodType** to **"O"** and **rhFactor** to **'+'**.
4. Create an overloaded constructor (public) with two (2) String parameters: **bt** and **rh**. In this constructor, **bloodType** should store **bt** while **rhFactor** should store **rh**.
5. Create a public method named **display**. This method will be used to display the values of **bloodType** and **rhFactor**.
6. In the **RunBloodData** class, import the **Scanner** class for the user input.
7. In the main method, add statements to ask the user to input the blood type and the Rhesus factor (+ or -). Instantiate a **BloodData** object name with arguments based on the user input. For example, `BloodData bd = new BloodData(input1, input2);` where **input1** and **input2** are String variables that stored what the user entered. If the user does not input anything, instantiate a **BloodData** object without an argument.
8. Print the confirmation message by invoking the display method through the object you created. For example, `bd.display();`

Note: Keep a copy of your code. It will be revised on your next laboratory session.

Sample Output:

```
Enter blood type of patient:
Enter the Rhesus factor (+ or -):
O+ is added to the blood bank.

Enter blood type of patient: B
Enter the Rhesus factor (+ or -): -
B- is added to the blood bank.
```

Explanation: In the first run, the user did not enter both values. Hence, the values stored in the default constructor are displayed.

GRADING RUBRIC (100 points):

Criterion	Description	Max Points
Correctness	The code produces the expected result.	40
Logic	The code meets the specifications of the problem.	40
Efficiency	The code is concise without sacrificing correctness and logic.	10
Syntax	The code adheres to the rules of the programming language.	10