COSC 1436 Fall 2023
Lab 02

Goal: The goal of this session is to introduce you to the environment in which you will be working this semester and to establish the procedures you need to follow in order to successfully design, code, debug, and document Java programs.

Part 1

If you are using BlueJ, start a new project, and turn on line numbers.

- Click on Project/New Project
  - Name the Project **Lab 02**
- Click on Tools/Preferences
  - Check the box that says **Display line numbers**

Take note of which folder BlueJ is using to store your programs. You will need it to submit your programs.

Exercise 1 : Calculating Crash Forces (40 pts)

Unknown Author is licensed under CC BY

Imagine crash testing a car into a wall. The 5000 pound car, starts moving at 30 m/s, and then stops in 0.05 seconds. Write a Java program to calculate how much force the car absorbs.

Here is how to calculate the net force:

According to Newton's Second Law of Motion, the force that an object exerts on another object is equal to the mass of the object times its acceleration. Thus, the formula:

Force = mass * acceleration

For this formula, we will measure mass in kilograms (kg), acceleration in meters/sec^2, and force in Newtons.

What are the inputs needed to calculate the crash force?

1. Mass of the crashing object (car)
   - Determine how much **mass** the crashed object contains.
     - The car is 5000 pounds. But the formula requires kilograms (kg), so we need to convert pounds to kg. On Earth, there are 2.2 pounds for every kg of mass, so:

   Mass of car = 5000 pounds ÷ 2.2 kg/pound = 2272.7272... kg

2. Deceleration of the car
    - Determine the acceleration (or deceleration) involved in the crash.
        - Imagine that the car was traveling as **30 meters per second (m/s)**— roughly 60 miles per hour – when it hit a wall, coming to a complete stop in 0.05 seconds – 5 hundredths of a second. To **calculate the acceleration**, simply divide the change in speed by the time it took to change.

          Acceleration of the car = (0 m/s - 30 m/s) ÷ 0.05 s = -600 m/s^2

          Note 1: (0 m/s) is the speed after the vehicle hit the wall.

          Note 2: The answer is negative because it was decelerating. When calculating the net force in the formula below, make this answer positive by adding the formula

          ```
          acceleration = -acceleration;
          ```

Using Newton's Second Law to calculate the net force involved in the crash:

mass * acceleration = Force

2272.7272... kg * 600 m/s^2 = 1,363,636.3636... Newtons (N)

The correct answer is 1,363,636.3636... N. This is over 500 times the car's weight!

Now that we have shown you how to do the calculation, create a new class in your Lab 02 project and write a Java program that does the calculations above.

- Ask the user to input the following:
    - Mass of the car in pounds.
    - Starting speed of the car in (m/s)
    - The duration in seconds when the car completely stops.
- Calculate and print out the following outputs:
    - Mass of car in kg
    - Acceleration (deceleration) of the car
    - Net force

You have already written programs that calculate one formula. Use them as a template and expand on it so it calculates all 3 formulas in one program. Take it one formula at a time.

Exercise 2 :

1) Programming Exercise (DIY) (40 pts)

Create a new class in your Lab 1 project and write a program that asks the user to enter the height, width, and depth of a box (in inches), and calculates the volume of the box in cubic feet.

Hint: 1 cubic foot = 1728 cubic inches

Part 2

Multiple Choice Questions (20 pts) :

At the end of your 2nd program, create some comments and type in your answers there.

1) Which one of the following statements is true? (3 pts)

   a. `int, char, short, long,` and `String` are primitive data types but `double and float` are not
   b. `int, char, double, float,` and `String` are primitive data types but `short` and `long` are not
   c. `int, char, short, long, double,` and `float` are primitive data types but `String` is not.
   d. `int, double, char, short, long, float, and String` are all primitive data types

2) Which of the following is NOT a rule that must be followed when naming identifiers? (3 pts)

   a. Uppercase and lowercase characters are distinct.
   b. After the first character, you may use the letters a-z, A-Z, the underscore, a dollar sign, or digits 0-9.
   c. The first character must be one of the letters a-z, A-Z, and underscore or a dollar sign.
   d. Identifiers can contain spaces.

3) Given the following two statements, please answer each question. (2 pts each)

```
int side1 = 10;
int side2 = 20;
```

Is each of the following statements valid (i.e., no syntax (or compile-time) error)? Answer **Yes** or **No**. ***If your answer is "no", please correct the left hand side (LHS) of the assignment so that the statement can be valid***.

```
3.1    int result = side1 / side2;
3.2    int result = side1 / 20.0;
3.3    double result = side1 % side2;
```

4) What is the result of evaluating the following expression? (2 pts)

```
double answer=0;
answer=(1/2 + 3.5) * 2.0;
```

a.8.0 b. 8 c. 7.0 d. 0

5) What does the following program output? (6 pts)

```
public class question5
{
  public static void main (String[] args)
     {
      int unitCost = 8;
      int items = 5;

      System.out.println("total cost: " + (unitCost * items) );
     }
}
```

```
a. total cost: + 40

b. total cost: 8*5

c. total cost: 40

d. "total cost: " 40
```

Submitting your work

BlueJ creates a `.java` file for each of your classes. Look for them in the folder BlueJ is using to save your work. Submit the `.java` file for each of your programs (there will be 2 of them).

For the multiple choice questions, create comments at the end of your 2nd program and type in the answers.