CSCI 1101 – 2017
Laboratory Report 5

Your task is to complete the assigned work using JGrasp or an IDE of your choosing. You may use your own computer, or one of the lab computers provided.

Your submission should be a **ZIP** file containing your source code files. You should submit your **ZIP file** on Brightspace:
http://dal.brightspace.com

Submission Deadlines (firm):
Monday Labs: due Wednesday by 12:00pm (noon)
Friday Labs: due Sunday by 12:00pm (noon)

NB:

- Try to submit this report *during* the lab so that your TA can check it for you before you submit!
- Attendance is mandatory in all labs, and will form part (10%) of your overall lab grade
- Acknowledge any help that you obtained from friends, TAs, the Learning Centre, etc., by adding comments to your code where appropriate. Obtaining help is fine, *so long as you acknowledge it!*
- Any students who cannot log on to the lab computers should speak to the Help Desk to set up their account.
- Textbooks, class handouts, and any other materials are welcomed and encouraged in all labs!
- Food and drink are not permitted in the computer labs
- Late labs are ***not*** accepted! It is known that computer errors, power outages, and network lag are 105% more likely to occur between 11:55-11:59am, in the moment they can do the most damage. Account for this, and give yourself the chance to make a timely submission!

Header Comments

Your code should now include header comments for **all** of your class (.java) files. The comment should include the lab/assignment number, the course (CSCI 1101), the name of your program and a short description of the entire class, the date, your name and Banner ID, and a declaration that matches the first page of this document (e.g., whether you received help). See the example below for what a header comment should include:

```
/*Lab1, Question 1 CSCI 1101
   Student.java holds information about a student at Dalhousie in CSCI1101 and
   their grades
   June 29, 2015
   John Smith B00112345
   This is entirely my own work. */

public class Student {
//rest of Code
```

If applicable, your demo class should then also have a similar header:

```
/*Lab1, Question 1  - demo class CSCI 1101
   StudentDemo.java is a demo program for the Student class. It creates student
   objects, and compares different students.
   June 29, 2015
   John Smith B00112345
   I received help with creating Student objects from my TA but the rest is my
   own work. */

public class StudentDemo {
//rest of Code
```

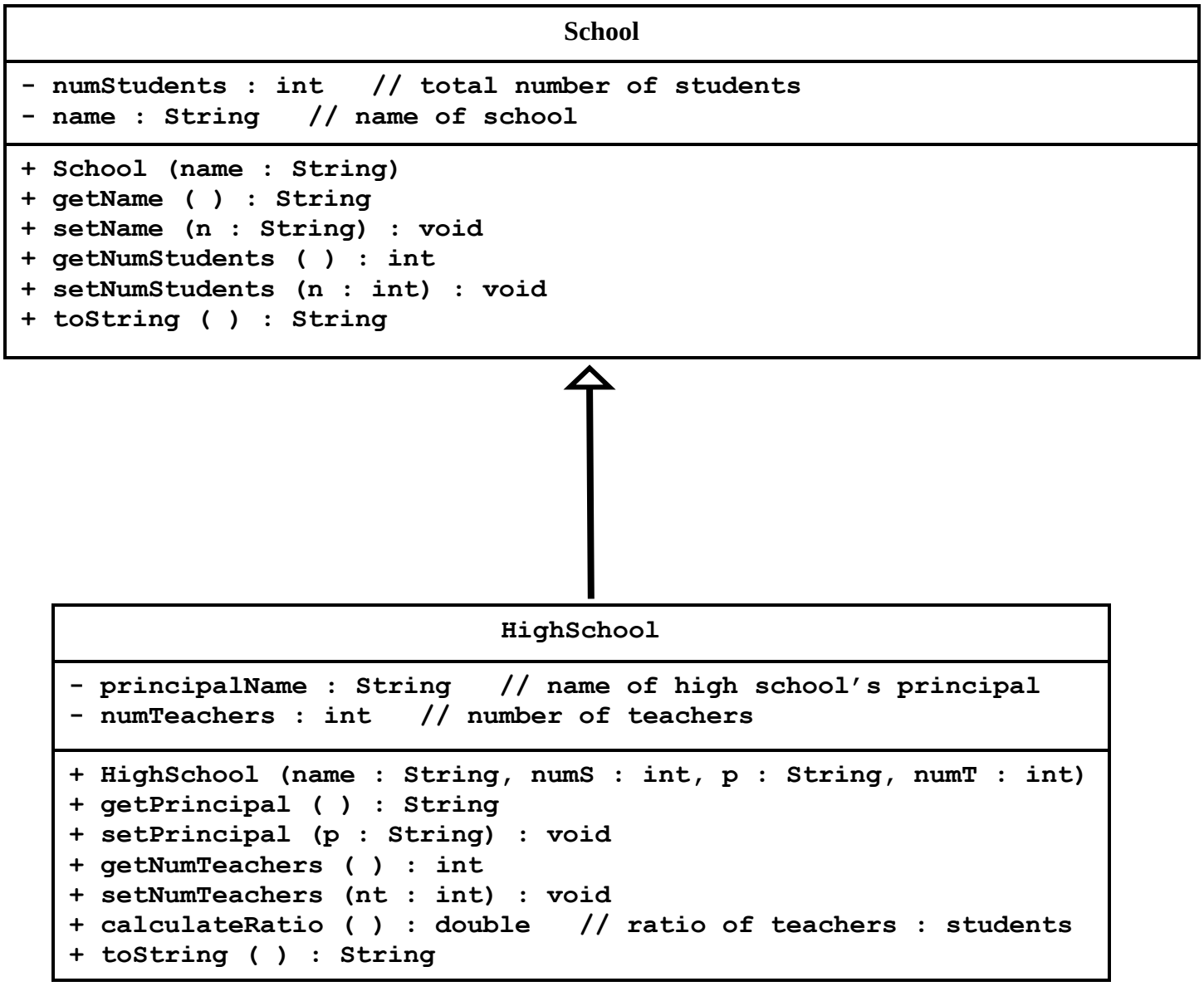
Exercise 1

Create a class called **`Restaurant`** that is the base class for all restaurants. It should have attributes for the restaurant's name and numSeats (number of seats inside the restaurant). It should have appropriate accessor and mutator methods for each of the attributes and a toString that prints the name and the number of seats.

Derive a class **`FastFoodRestaurant`** from **`Restaurant`**. The new class should override the accessor method for the numSeats, reporting actual seats times 2 (twice the number of seats in a restaurant because it also counts the parking spots where people eat in their cars as restaurant seating). It also should override the toString method so that it includes the correct number of seats and the restaurant ad slogan “Best Burgers Ever!”. For example, the toString in the **`FastFoodRestaurant`** might return [Welcome to Bob's Burgers with 30 seats. "Best Burgers Ever!"]

Write an appropriate demo class that full tests both **`Restaurant`** and **`FastFoodRestaurant`**.

Exercise 2
Implement the following UML diagram.



Notes:
`School` class:

- The toString method returns the name of the school and the number of students.

`HighSchool` class

- calculateRatio( ) returns the ratio of teachers to students in the HighSchool
- The toString method builds on the super's toString method by adding the Principal's name, the number of teachers and the ratio of teachers to students.

Write an appropriate demo class that full tests both **`School`** and **`HighSchool`**.