Exercise 6

Implementing Classes 1: Cylinder Tank Problem

Introduction:

When the code base of problem will go in size so will its complexity. But when we think of the entire program as single entity, composed of autonomous sub-entities then we can simplify its complexity. In computer science, abstracting information for the perusal of the user is the first step in describing the features of a program. The class in C++ is the means in which the abstraction can be translated into a user-defined type.

For this exercise, the cylinder would be represented as an object. The volume is the property used to describe the cylinder. The equation is as follows,

$$v = \pi r^2 h$$

Learning Outcomes:

- Create a class for a cylinder tank with the appropriate member variables and functions.
- Implement and tests the functions that are in the class by creating and instance of the class (object).

Problem background

1. Create a project file for this exercise.
2. On a header file, `cylinderTank.h`, create a class named `cylinderTank` with the private variables indicated as follows:

```
double height;       // the height of the tank (meters)
double radius;       //the width of the tank (meters)
double aveDrainRate; //average drain rate of the tank (liters/sec)
double aveFillRate;  //average fill rate of the tank (liters/sec)
```

3. In your class, write the function prototypes as private functions indicated below

```
cylinderTank();      //default constructor, sets all variables to 0
void setHeight (double);   //set tank height
void setRadius (double);   //set tank radius
void setDrainRate (double);//set tank drain rate
void setFillRate (double); //set tank fill rate
double getHeight ();       //return the value of the height
double getRadius ();       //return the value of the radius
double getDrainRate ();    //return the value of the drain rate
double getFillRate ();     //return the value of the fill rate
double volume ();          //computes and returns the volume

void timeToFillEmpty()
/*computes and displays the time to fill in hr:min:sec if tank is empty*/

void timeToFillPartial(double)
/*computes and displays the time to fill in hr:min:sec if the tank is
partially filled from the value of the passing parameter.*/
```

```
~cylinderTank()   //default destructor, no content on function body
```

4. On another `cylinderTankImp.cpp`, implement the function body for the class `cylinderTank`.
5. Test the operation of these function on `main.cpp` function `main()`.