SCHOOL OF COMPUTER SCIENCES
UNIVERSITI SAINS MALAYSIA
Semester II Session 2023/2024

Tutorial CPT113
`friend`, Operator Overloading

Learning Outcomes:
- Building friend function and established friend class relationship
- Writing operator overloading function

---

1. Based on the tutorial from previous week, answer the following questions:
   a) Write a friend function to compare the volume and the surface area of two objects: Prism and Cube. Print out which object has a bigger volume and bigger surface area from the friend function.
   b) Write an operator overloading for operator '==' which compare the surface and volume of the same object type.

2. Write two classes called Person and Student. Make the Person class as a friend class for students. The class Person has name, hometown state, and list of schools the person went to before. The class Student has the student ID, school and desa name. The program should have print function to print all the information.
   Then, modify the above program so it uses composition.

3. Based on the tutorial from Week Inheritance & Composition, answer the following questions:
   a. Write a friend function to compare the volume and the surface area of two objects: Cone and Cylinder. Print out which object has a smaller volume and smaller surface area from the friend function.
   b. Write an operator overloading for operator '==' which compare the surface and volume of the same object type.

4. Write two classes called Book and Author. Make the Book class as a friend class for Author. The class Author has name, hometown, and list of genre of books the author wrote. The class Book has the book ID, book type and genre. The program should have print function to print all the information.
   Modify the above program so it uses composition.

5. Assuming that a year has 365 days, write a class named DayOfYear that takes an integer representing a day of the year and translates it to a string consisting of the month followed by day of the month. For example,
   Day 2 would be *January 2* .
   Day 32 would be *February 1* .
   Day 365 would be *December 31*.

The constructor for the class should take as parameter an integer representing the day of the year, and the class should have a member function `print()` that prints the day in the month–day format. The class should have an integer member variable to represent the day and should have static member variables holding string objects that can be used to assist in the translation from the integer format to the month-day format.

Test your class by inputting various integers representing days and printing out their representation in the month–day format.

6. **Day of the Year Modification**
   Modify the `DayOfYear` class, written in Programming Challenge 2, to add a constructor that takes two parameters: a string object representing a month and an integer in the range 0 through 31 representing the day of the month. The constructor should then initialize the integer member of the class to represent the day specified by the month and day of month parameters. The constructor should terminate the program with an appropriate error message if the number entered for a day is outside the range of days for the month given.

   Add the following overloaded operators:
   ++ **prefix and postfix increment operators.** These operators should modify the `DayOfYear` object so that it represents the next day. If the day is already the end of the year, the new value of the object will represent the first day of the year.

   –– **prefix and postfix decrement operators** . These operators should modify the `DayOfYear` object so that it represents the previous day. If the day is already the first day of the year, the new value of the object will represent the last day of the year.

7. **FeetInches Modification**
   Modify the FeetInches class discussed in this chapter so it overloads the following operators:
   <=
   >=
   !=
   Demonstrate the class's capabilities in a simple program.

8. **`FeetInche s` Class Copy Constructor and `multiply` Function**

Add a copy constructor to the `FeetInches` class. This constructor should accept a `FeetInches` object as an argument. The constructor should assign to the `feet` attribute the value in the argument's `feet` attribute, and assign to the `inches` attribute the value in the argument's `inches` attribute. As a result, the new object will be a copy of the argument object.

Next, add a `multiply` member function to the `FeetInches` class. The `multiply` function should accept a `FeetInches` object as an argument. The argument object's `feet` and `inches` attributes will be multiplied by the calling object's `feet` and `inches` attributes, and a `FeetInches` object containing the result will be returned.

9. Carpet Calculator

The Westfield Carpet Company has asked you to write an application that calculates the price of carpeting for rectangular rooms. To calculate the price, you multiply the area of the floor (width times length) by the price per square foot of carpet. For example, the area of floor that is 12 feet long and 10 feet wide is 120 square feet. To cover that floor with carpet that costs $8 per square foot would cost $960. (12 X 10 X 8 = 960.)

First, you should create a class named `RoomDimension` that has two `FeetInches` objects as attributes: one for the length of the room and one for the width. (You should use the version of the `FeetInches` class that you created in Programming Challenge 11 with the addition of a `multiply` member function. You can use this function to calculate the area of the room.) The `RoomDimension` class should have a member function that returns the area of the room as a `FeetInches` object.

Next, you should create a `RoomCarpet` class that has a `RoomDimension` object as an attribute. It should also have an attribute for the cost of the carpet per square foot. The `RoomCarpet` class should have a member function that returns the total cost of the carpet.

Once you have written these classes, use them in an application that asks the user to enter the dimensions of a room and the price per square foot of the desired carpeting. The application should display the total cost of the carpet.