Arab Academy of Science and Technology and Maritime Transport

COLLEGE OF ENGINEERING & TECHNOLOGY

Computer Engineering Department

Course: **Object Oriented Programming**

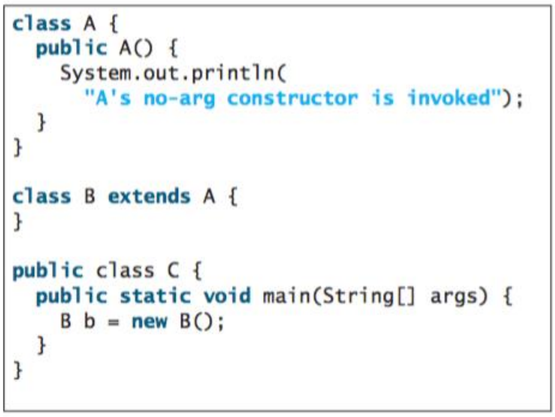
Course Code: **CC316**

Sheet No.: **2**

1. What are common relationships among classes?
2. What is association? What is aggregation? What is composition?
3. What is UML notation of aggregation and composition?
4. Write the **Course** class, which defines the variables: courseName, students (Arraylist of Strings). It also had a constructor that initiates objects using only the courseName; getter methods for the courseName, students and numberOfStudents; and a function that adds a new student.
   - Implement the **dropStudent** method.
   - Add a new method named **clear()** that removes all students from the course.

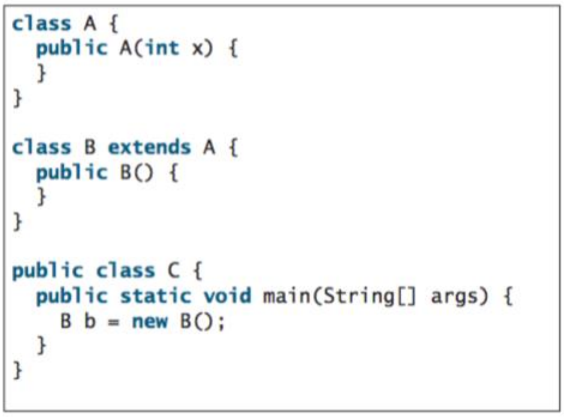
5. (The MyInteger class) Design a class named MyInteger. The class contains:
   - An int data field named value that stores the int value represented by this object.
   - A constructor that creates a MyInteger object for the specified int value. A getter method that returns the int value.
   - The methods isEven(), isOdd(), and isPrime() that return true if the value in this object is even, odd, or prime, respectively.
   - The static methods isEven(int), isOdd(int), and isPrime(int) that return true if the specified value is even, odd, or prime, respectively.
   - The static methods isEven(MyInteger), isOdd(MyInteger), and isPrime(MyInteger) that return true if the specified value is even, odd, or prime, respectively.
   - The methods equals(int) and equals(MyInteger) that return true if the value in this object is equal to the specified value.
   - Draw the UML diagram for the class and then implement the class. Write a client program that tests all methods in the class.

6. What is the output of running the class C in (a)? What problem arises in compiling the program in (b)?



```
class A {
  public A() {
    System.out.println(
      "A's no-arg constructor is invoked");
  }
}

class B extends A {
}

public class C {
  public static void main(String[] args) {
    B b = new B();
  }
}
```

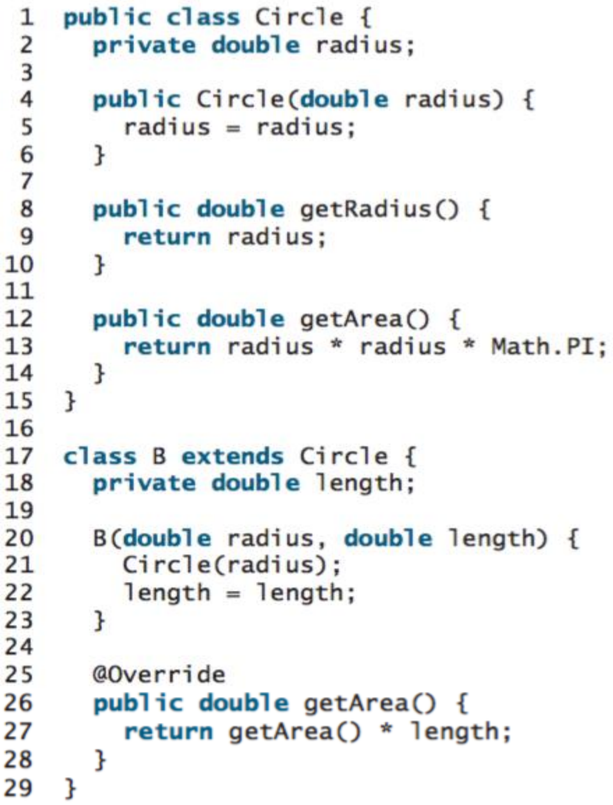
(a)



```
class A {
  public A(int x) {
  }
}

class B extends A {
  public B() {
  }
}

public class C {
  public static void main(String[] args) {
    B b = new B();
  }
}
```

(b)

7. Identify the problems in the following code:



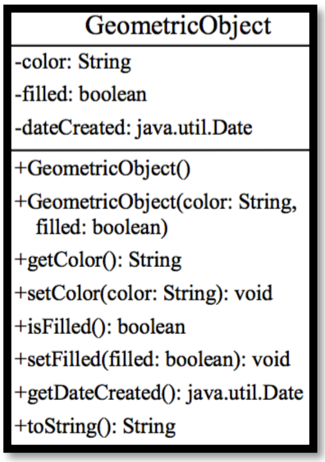
```
public class Circle {
  private double radius;

  public Circle(double radius) {
    radius = radius;
  }

  public double getRadius() {
    return radius;
  }

  public double getArea() {
    return radius * radius * Math.PI;
  }
}

class B extends Circle {
  private double length;

  B(double radius, double length) {
    Circle(radius);
    length = length;
  }

  @Override
  public double getArea() {
    return getArea() * length;
  }
}
```

8. If a method in a subclass has the same signature as a method in its superclass with the same return type, is the method overridden or overloaded?

9. If a method in a subclass has the same signature as a method in its superclass with a different return type, will this be a problem?

10. (*The Person, Student, Employee, Faculty, and Staff classes*) Design a class named **Person** and its two subclasses named **Student** and **Employee**. Make **Faculty** and **Staff** subclasses of **Employee**. A person has a name, address, phone number, and email address. A student has a class status (freshman, sophomore, junior, or senior). Define the status as a constant. An employee has an office, salary, and date hired. A faculty member has office hours and a rank. A staff member has a title. Override the **toString** method in each class to display the class name and the person's name. Draw the UML diagram for the classes and implement them. Write a test program that creates a **Person**, **Student**, **Employee**, **Faculty**, and **Staff**, and invokes their **toString()** methods.

11. (*The Triangle class*) Design a class named **Triangle** that extends **GeometricObject**. The class contains:
    - Three **double** data fields named **side1**, **side2**, and **side3** with default values **1.0** to denote three sides of the triangle.
    - A no-arg constructor that creates a default triangle.
    - A constructor that creates a triangle with the specified **side1**, **side2**, and **side3**.
    - The accessor methods for all three data fields.
    - A method named **getArea()** that returns the area of this triangle.
    - A method named **getPerimeter()** that returns the perimeter of this triangle.
    - A method named **toString()** that returns a string description for the triangle.
    - Draw the UML diagrams for the classes **Triangle** and **GeometricObject** and implement the classes. Write a test program that prompts the user to enter three sides of the triangle, a color, and a Boolean value to indicate whether the triangle is filled. The program should create a **Triangle** object with these sides and set the **color** and **filled** properties using the input. The program should display the area, perimeter, color, and true or false to indicate whether it is filled or not.



GeometricObject

-color: String
-filled: boolean
-dateCreated: java.util.Date

+GeometricObject()
+GeometricObject(color: String, filled: boolean)
+getColor(): String
+setColor(color: String): void
+isFilled(): boolean
+setFilled(filled: boolean): void
+getDateCreated(): java.util.Date
+toString(): String