Object Oriented Programming 1
Fall 2015-16
Lab Manual: 06

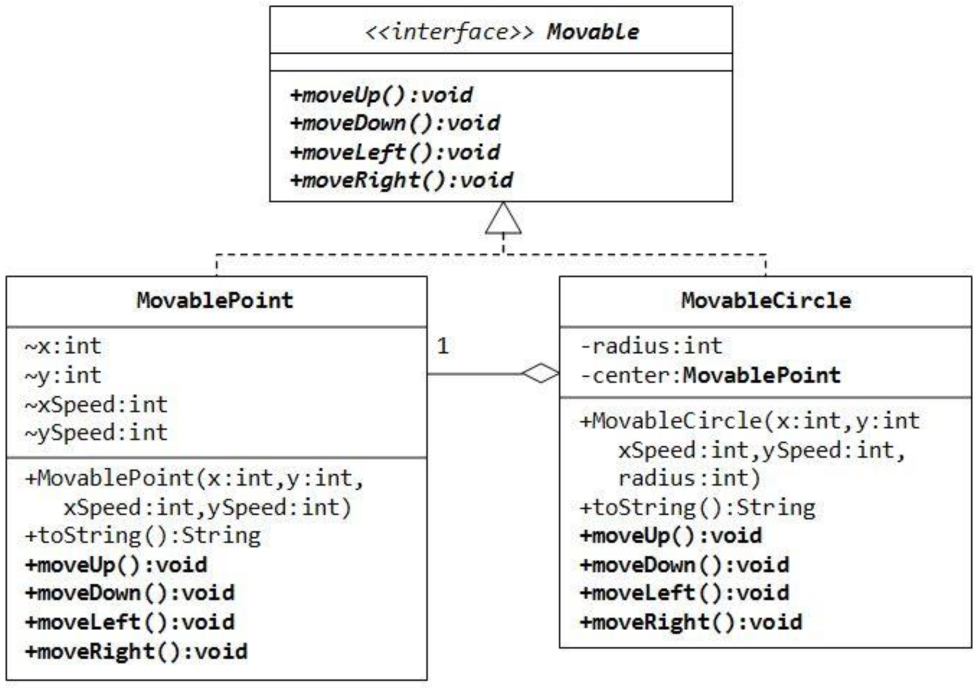
Lab Task:

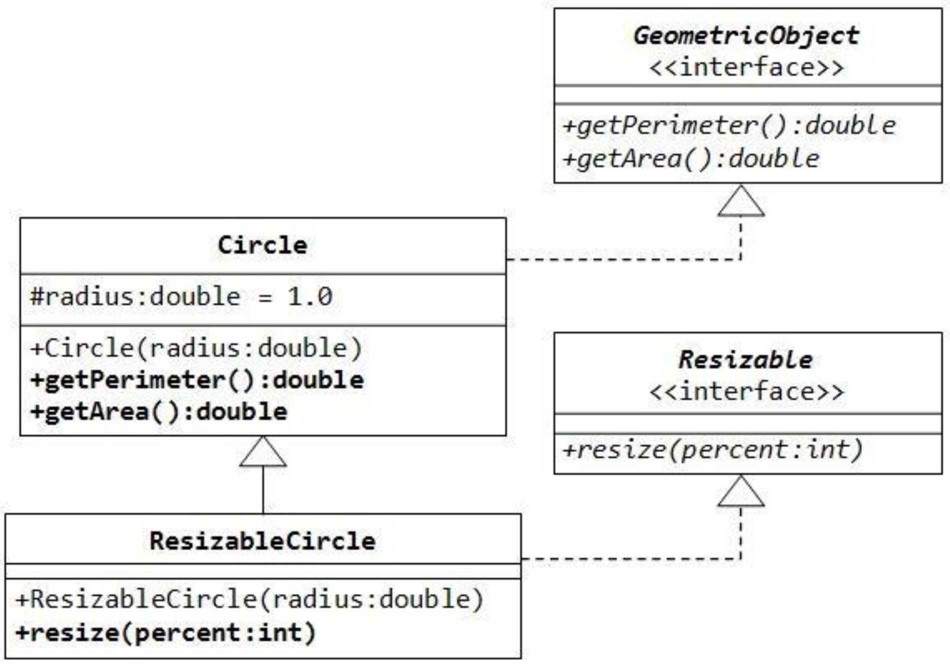
1. Lab Review, and start with unfinished classes from Lab_05
2. Develop Java classes

Note: Student must follow the name of class, member variables, and functions.
And students should use fully qualified names for these, as well camel notions.
And the syntax alignment has to be as it should be.

1. Lab_05 Classes – Review:

2. Develop Java Classes:

Test Interface 1	**Interface:** BasicCalculatorInterface which has these below abstract methods • int sum(int x, int y) • int sub(int x, int y) • int multiplication(int x, int y) • int division(int x, int y) ScientificCalculatorInterface • int XtoY(int x, int y) • Other scientific methods that you find best fit for you Develop necessary class to implement these two interfaces and test from main method.
Classes **MovablePoint** & **MovableCircle** implements **MovableInterface**.	 <<interface>> Movable +moveUp():void +moveDown():void +moveLeft():void +moveRight():void MovablePoint ~x:int ~y:int ~xSpeed:int ~ySpeed:int +MovablePoint(x:int,y:int, xSpeed:int,ySpeed:int) +toString():String +moveUp():void +moveDown():void +moveLeft():void +moveRight():void 1 MovableCircle -radius:int -center:MovablePoint +MovableCircle(x:int,y:int xSpeed:int,ySpeed:int, radius:int) +toString():String +moveUp():void +moveDown():void +moveLeft():void +moveRight():void

Class **Circle implements** Interfaces **GeometricObject** Class **ResizableCircle** implemets **ResizableInterface** & inherits/extends	 GeometricObject <<interface>> +getPerimeter():double +getArea():double Circle #radius:double = 1.0 +Circle(radius:double) +getPerimeter():double +getArea():double Resizable <<interface>> +resize(percent:int) ResizableCircle +ResizableCircle(radius:double) +resize(percent:int)
Try this	Consider a bank has three different types of accounts. An account holder (Person/Student) can have any one of these below type. 1. Current 2. Savings 3. Overdraft These three types of account has different advantages and disadvantages • Current Account ○ Current account holders can withdraw all of his/her amount • Saving Account ○ Saving account holders can withdraw maximum 80% of his/her total amount • Overdraft account ○ Overdraft account holders can withdraw additional amount which is set by at account creation time (overdraft limit amount) After implement this structure change the account type (ex: current to saving or else) and changes will automatically reflected as well. You have to develop this scenario on your own.

Special Note: Do not code interface and class on same .java file. Use separate files for interfaces. The interface should be public.