

Bonus 1

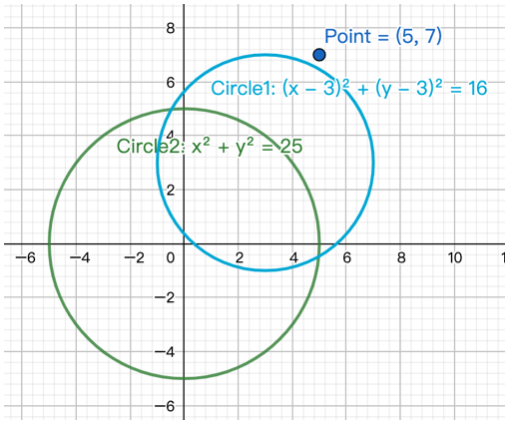
Create a new Eclipse project named **YourStudentId_OOP_Bonus1** and add a class named **Circle** to the project. You **must** comply with the following variable names.

1. Applications of Circle (Write in Circle class)

The question will ask the user to input the circle radius and the position of the circle center to define the *Circle1* object. In the figure, *Circle1* assumes that the circle radius is 4 and the center axis is (3,3). The question will ask you to create the *area*, *circumference* method to calculate area, and circumference of Circle1.

Then, the user is required to arbitrarily input the coordinates of a point and establish the *rangePoint* method to get the relationship between this point and Circle1.

In this question, please define *Circle2* object with a circle radius of 5 and a circle center coordinate of (0,0), and then create a *rangeCircle* method to determine the relationship between *Circle1* and *Circle2*.

Circle	
<p>Attributes:</p> <p>radius, x, y, area, circumference</p> <p>Public Interface:</p> <p>area(one parameter)</p> <p>circumference(one parameter)</p> <p>rangePoint(six parameters)</p> <p>rangeCircle(six parameters)</p>	

a. In the *main* method

- Declare the Circle object named *circle1*, the radius and center coordinates of the circle are determined by user input.
- Declare the Circle object named *circle2*, the radius of this circle is 5, and the center coordinate is (0,0).
- Use Scanner object named *inputScanner* to read the radius and center coordinate of the *circle1* and assign to the *radius*, *x*, and *y* attributes.
- Use Scanner object named *inputScanner* to read the point coordinates input by the user.
- Call the *area* method and assign the return value to the *area* attribute.
- Call the *circumference* method and assign the return value to the *circumference* attribute.
- Call the *rangePoint* method to determine the relationship between the *point* and the *circle1*.
- Call the *rangeCircle* method to determine the relationship between the two circles.

- Print out as sample output.

b. In the *area* method

- Insert the parameter into the *circle area formula* to calculate the circle area.
- Declare a local variable named *area* and assign the calculation result to the variable.
- Return the result.

c. In the *circumference* method

- Insert the parameter into the *circumference formula* to calculate the *circumference*.
- Declare a local variable named *circumference* and assign the calculation result to the variable.
- Return the result.

d. In the *rangePoint* method

- Pass the radius and center coordinates of *Circle1* together with the *point* coordinates entered by the user into this method.
- Use the following basis to determine the relationship between the point and the circle.

$$d = \sqrt{(x - circle_x)^2 + (y - circle_y)^2}$$

Relationship	Output
$d < \text{radius}$	the point inside the circle.
$d = \text{radius}$	the point on the circle.
$d > \text{radius}$	the point outside the circle.

- Print out as sample output.

e. In the *rangeCircle* method

- Pass the radius and center coordinates of *Circle1* and *Circle2* into this method.
- Use the following formular to determine the relationship between the two circles.

$$d = \sqrt{(circle_x1 - circle_x2)^2 + (circle_y1 - circle_y2)^2}$$

Relationship	Output
$d < r1 - r2 $	the centre of one circle will lie on the other circle.
$d = r1 - r2 $	two circles are concentric.
$ r1 - r2 < d < r1 + r2$	two circles will intersect at two points.
$d = r1 + r2$	the circles touch externally.
$d > r1 + r2$	the circles do not touch or intersect each other.

- Print out as sample output.

Sample output:

Please input the radius of the circle1: 4

Please enter xy-coordinate of center for the circle1 (separated by spaces): 3 3

please enter the xy-coordinate of the point: 5 7

The area of the circle1 is 50.27

The circumference of the circle1 is 25.13

The relationship between point and circle:the point outside the circle.

The relationship between the two circles:two circles intersect at two points.

API reference:

Modifier and Type	Method and Description
static int	abs(int a) Returns the absolute value of an int value.
static int	round(double a) Returns the closest int to the argument, with ties rounding up.
static double	pow(double a, double b) Returns the value of the first argument raised to the power of the second argument.
static double	sqrt(double a) Returns the correctly rounded positive square root of a double value.
static double	cos(double a) Returns the trigonometric cosine of an angle.
static double	toRadians(double angdeg) Converts an angle measured in degrees to an approximately equivalent angle measured in radians.
static double	toDegrees(double angrad) Converts an angle measured in radians to an approximately equivalent angle measured in degrees.
static double	acos(double a) Returns the arc cosine of a value; the returned angle is in the range 0.0 through pi.

Submission: Submit your project as “.zip file” via Moodle. No other submissions will be graded.

Reminder: Please zip **the whole project**

Deadline: Tomorrow’s midnight (for both Mon56 and Tue23)