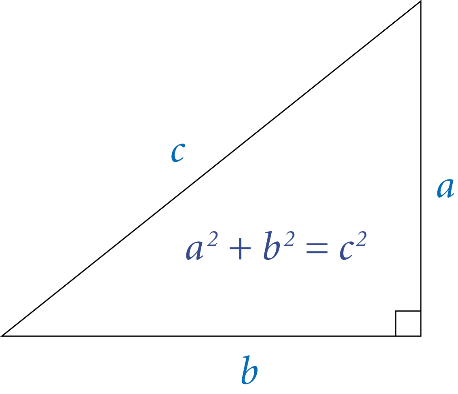
***Basic programming concepts***

*The purpose of this assignment is to introduce you to programming in Java and familiarize you with the mechanics of preparing and submitting assignment solutions.*

1. **Strings and command-line arguments.** Write a program HelloGoodbye.java that takes two names as command-line arguments and prints hello and goodbye messages as shown below (with the names for the hello message in the same order as the command-line arguments and with the names for the goodbye message in reverse order).
2. ~/Desktop/hello> javac HelloGoodbye.java
3. ~/Desktop/hello> java HelloGoodbye Kevin Bob
4. Hello Kevin and Bob.
5. Goodbye Bob and Kevin.
6. ~/Desktop/hello> java HelloGoodbye Alejandra Bahati
7. Hello Alejandra and Bahati.
8. Goodbye Bahati and Alejandra.

2**. Integers and booleans.** Write a program RightTriangle that takes three int command-line arguments and determines whether they constitute the side lengths of some right triangle.



The following two conditions are necessary and sufficient:

* + Each integer must be positive.
  + The sum of the squares of two of the integers must equal the square of the third integer.

~/Desktop/hello> javac RightTriangle.java

~/Desktop/hello> java RightTriangle 3 4 5

true

~/Desktop/hello> java RightTriangle 13 12 5

true

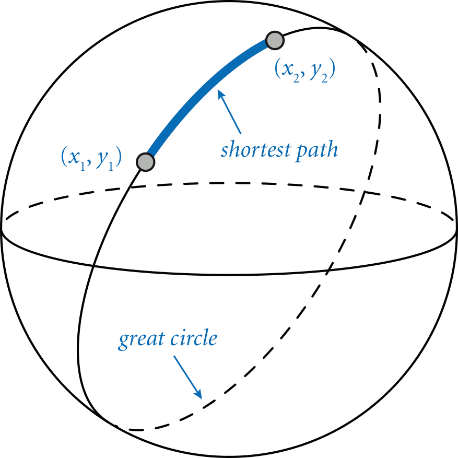
~/Desktop/hello> java RightTriangle 1 2 3

false

~/Desktop/hello> java RightTriangle -3 4 -5

false

3. **Floating-point numbers and the Math library.** The *great-circle distance* is the length of the shortest path between two points (x1,y1) and (x2,y2) on the surface of a sphere, where the path is constrained to be along the surface.



Write a program GreatCircle.java that takes four double command-line arguments x1, y1, x2, and y2—the latitude and longitude (in degrees) of two points on the surface of the earth—and prints the great-circle distance (in kilometers) between them. Use the following [Haversine formula](https://en.wikipedia.org/wiki/Haversine_formula)

where r=6,371.0 is the mean radius of the Earth (in kilometers).

~/Desktop/hello> javac GreatCircle.java

~/Desktop/hello> java GreatCircle 40.35 74.65 48.87 -2.33 // Princeton to Paris

5902.927099258561 kilometers

~/Desktop/hello> java GreatCircle 60.0 15.0 120.0 105.0 // for debugging

4604.53989281927 kilometers

*Hint:* The command-line arguments are given in degrees but Java’s trigonometric functions use radians. Use Math.toRadians() to convert from degrees to radians.

*Although the Earth is not a perfect sphere, this formula is a good approximation to the true distance.*

4. **Type conversion.** Several different formats are used to represent color. For example, the primary format for LCD displays, digital cameras, and web pages—known as the *RGB format*—specifies the level of red (R), green (G), and blue (B) on an integer scale from 0 to 255. The primary format for publishing books and magazines—known as the *CMYK format*—specifies the level of cyan (C), magenta (M), yellow (Y), and black (K) on a real scale from 0.0 to 1.0.

Write a program CMYKtoRGB.java that converts from CMYK format to RGB format using these mathematical formulas:

**white = 1 – black**

**red = 255 \* white \* (1 -cyan)**

**green = 255 \* white \* (1- magenta)**

**blue = 255 \* white \* (1 -yellow)**

Your program must take four double command-line arguments cyan, magenta, yellow, and black; compute the corresponding RGB values, each rounded to the nearest integer; and print the RGB values, as in the following sample executions:

~/Desktop/hello> javac CMYKtoRGB.java

~/Desktop/hello> java CMYKtoRGB 0.0 1.0 0.0 0.0 // magenta

red = 255

green = 0

blue = 255

~/Desktop/hello> java CMYKtoRGB 0.0 0.4392156862745098 1.0 0.0 // Princeton orange

red = 255

green = 143

blue = 0

**Submission.** Submit a .zip file containing HelloWorld.java, HelloGoodbye.java, RightTriangle.java, GreatCircle.java, and CMYKtoRGB.java. You may not call library functions except those in the java.lang (such as Integer.parseInt() and Math.sqrt()). Do not use loops, conditional statements, arrays, or other Java features that have not yet been introduced in the course.