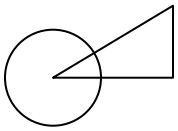
CS 415: Programming Languages

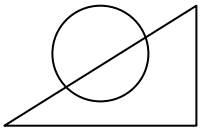
Homework 1: Fortran

Due Friday, 9 September by 10 a.m.

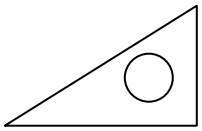
For this homework, you will need to write a Fortran program that will compute whether a circle and a triangle intersect. The program will get (as user input) the locations of a circle and a triangle, and then test them to see if they intersect. This assignment is completely in a 2-dimensional Cartesian coordinate system. There are four cases in which a circle and a triangle can intersect:



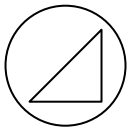
Case 1: One or more of the triangle points lies within the circle



Case 2: One of the triangle lines intersects the circle



Case 3: The circle lies completely within the triangle



Case 4: The triangle lies completely within the circle

Case 4 is covered by case 1 (as all the points are within the circle). For ease of implementation, we will be ignoring case 3. Thus, you will need to implement cases 1 and 2 in your program.

Your Fortran program must include (and use!) the following language features: function definitions, if statements, and looping constructs. You are welcome to use more language features; but at least one of each of these must be included. You do not need to worry about formatted output; indeed, you can print out anything you want, as long as the last line states whether the two shapes intersect (and which triangle points/lines intersect the circle are also printed out).

Implementation

The program will be run and tested using GNU's g77 compiler. If you are using Windows, the easiest way to install it is via Cygwin (http://www.cygwin.com/) – be sure to select the gcc-g77 compiler package (in the Devel section). Details about how to do this (and how to use Cygwin) can be found in the course lecture notes (in the Fortran slide set). You may also want to install the ocaml package, as we will be using that later in the course.

A tutorial on Fortran 77 is available online at http://www.aspire.cs.uah.edu/textbook/index_f77.html. The Fortran lecture notes also contains a short Fortran tutorial.

Part 1: Linear equation solver

You will need to write a Fortran function that solves linear equations of the form:

$$am_1 + bm_2 + c = 0$$

$$xm_1 + ym_2 + z = 0$$

This function will just return the value of m_1; the value of m_2 can then be determined from m_1 and either of the original equations.

How to solve a linear equation: first check if any of the coefficients (*a*, *b*, *x*, or *y*) are zero. If *a* or *x* are zero, just return zero. If *b* or *y* are zero, then the equation is of the form $am_1 + c = 0$, which can be easily solved. Otherwise, multiply one of the lines by a factor so that *b* and *y* are equal to each other. Then subtract *x* from *a*, *y* from *b*, and *z* from *c*. This will yield an equation of the form $am_1 + c = 0$, which can easily be solved. Note that this algorithm for solving a linear equation is not fool-proof, but it is sufficient for this homework assignment.

Part 2: Getting user input

The user should be prompted to enter a total of 9 real values:

- The radius of the circle
- The (x, y) coordinate of the circle
- The (x, y) coordinate of the first triangle point
- The (x, y) coordinate of the second triangle point
- The (x, y) coordinate of the third triangle point

The programs will be tested via an automated script, so please ensure that it takes in the 9 specified values in that order (one number per line).

Part 3: Intersection case 1

This case tests whether any of the triangle's points line within the circle. Recall that the formula to determine the distance between two points is (x_1, y_1) and (x_2, y_2) is $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$. If the distance between the point and the center of the circle is less than the radius, then the point lies within the circle. It does not matter in which order you test the points. Your program should print out which points of the triangle the circle contains (or, for each of the points, that none of them are contained within the circle).

Part 4: Intersection case 2

This case tests whether any of the line segments intersect the circle (if any of the points are within the circle, they would have been caught by the previous case). There are multiple algorithms to determine this; one is as follows. It does not matter in which order you test the sides.

1. Determine the formula for the line of the triangle we are dealing with. Recall that the formula for a line is of the form $(x, y) = (1,1) + m(2,2)$, where (1,1) is the intercept, and (2,2) is the vector (slope). You can not use the formula $y = mx + b$, as that will not handle vertical lines.
2. Find the point on that line that is closest to the center of the circle
    a. Find a second line that is perpendicular to the first, and runs through the center of the circle (i.e. has the circle center as the intercept). If the first line has a slope of (x, y), then a perpendicular line will have slope $(-y, x)$ (or $(y, -x)$).
    b. Find the point where these two lines meet. This is done by setting the line equations equal to each other, and using your linear equation solver to find the result for *m*; that value is then plugged back into the line equation.
3. Next, we need to check if that point is within the circle
    a. Done as in case 1, above
4. Lastly, check whether that point is between the points of the triangle side
    a. The circle may fall on the line formed by the triangle side, but not within the bounds of the triangle itself. An example of this is shown below.

Your program should print out which sides of the triangle the circle intersects with (or, for each of the sides, that none of them intersect the circle).

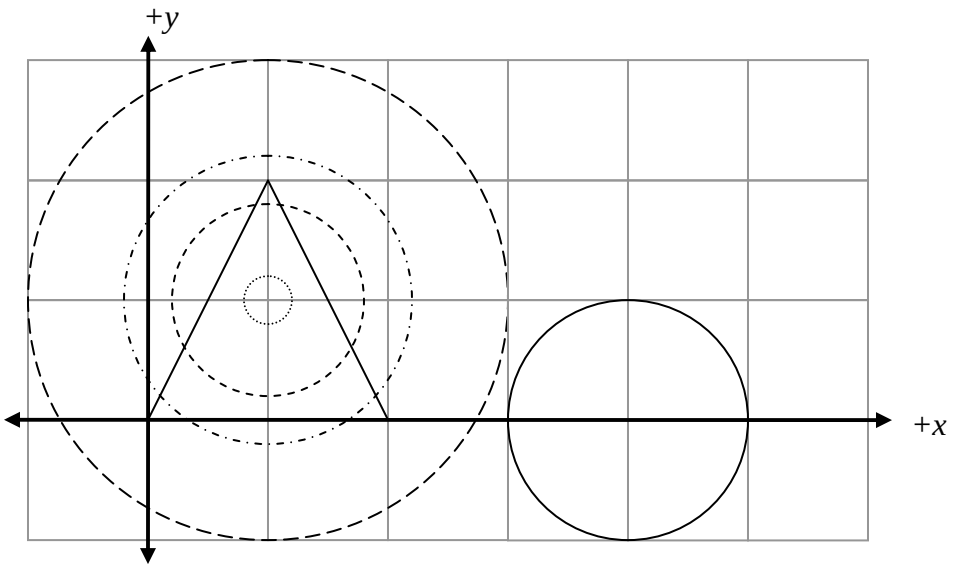
Submission

You should submit a fully commented, working program. All your code should be in one file. Submission details will be discussed in lecture.

Sample Input

Consider the triangle with points (0,0), (2,0), and (1,2). Unless specified otherwise, the circle is centered at (1,1). A diagram for this example is shown on the next page.

- If the circle has radius 0.2, then it is completely within the triangle (case 3), and this program can return any value, as we are not dealing with this case. This is the innermost circle with a dotted pattern.
- If the circle has radius 0.8, it will intersect two of the sides and contain none of the points. This is the circle with small dashes.
- If the circle has radius 1.2, it will intersect all three sides, and contain one point (1,2). This is the circle with a dash-dot pattern.
- If the circle has radius 2, it will intersect none of the sides, but contain all three points. This is the outermost circle with the long dash pattern.
- If the circle has radius 1 and is located at (4,0), then it will not intersect the triangle. This is the circle off to the right side of the triangle. Note that this circle does lie on one of the lines formed by the triangle edge (the base, in particular), but not within the bounds of the triangle.


+y
+x