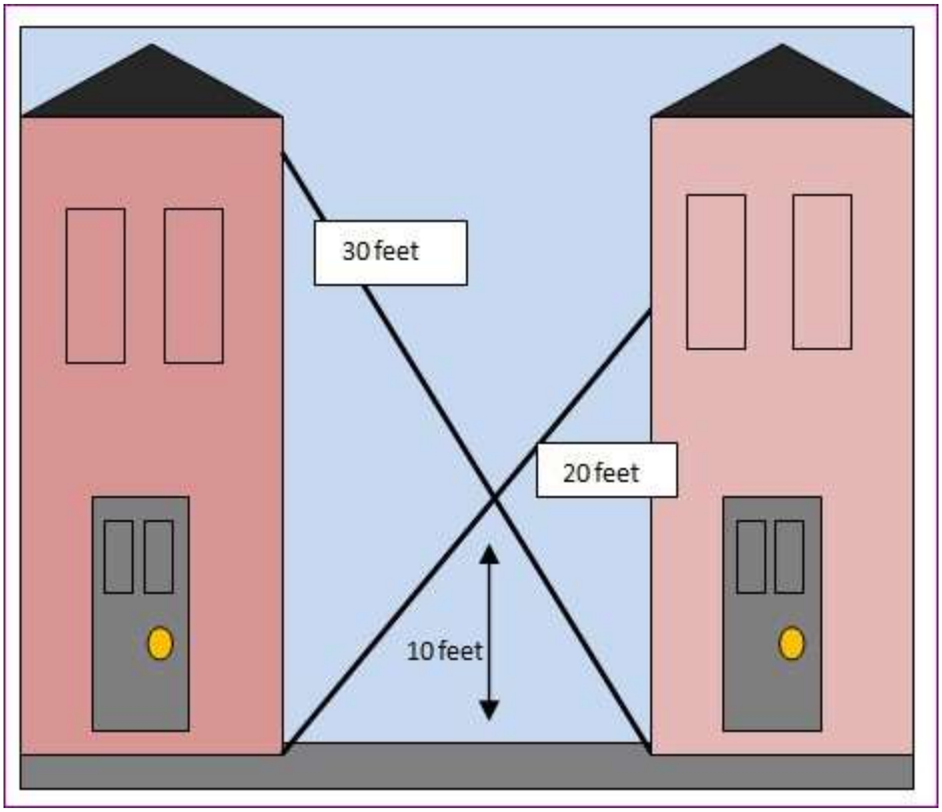
The twin-ladder Puzzle lab

In an alleyway between two buildings two ladders cross each other, the problem is to find the width of the alley given that the ladders are 20 feet and 3o feet long, and the point at which they cross is 10 feet above the ground.   See the picture below.

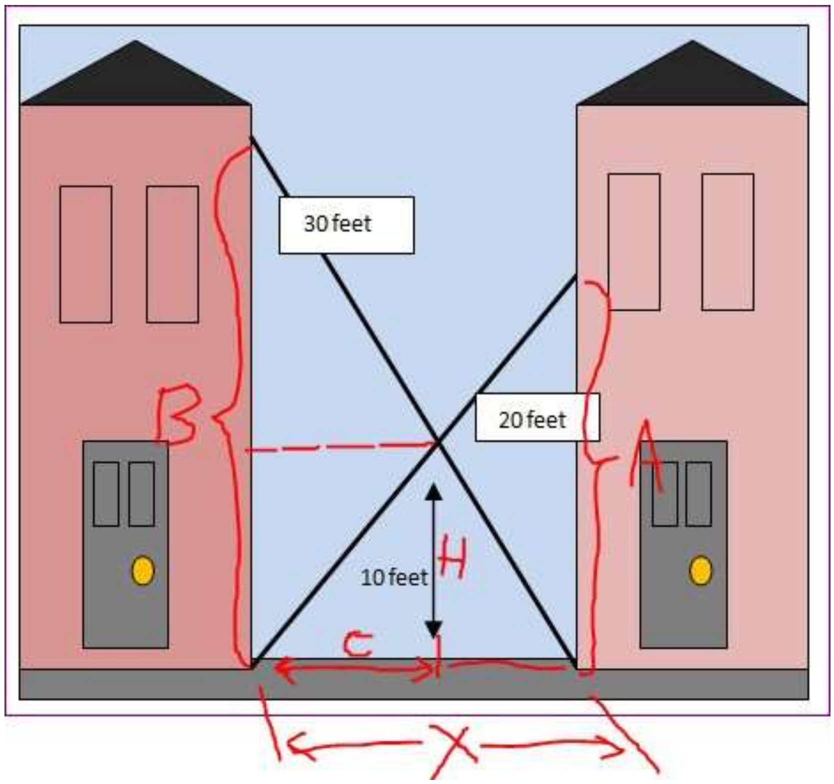


The problem looks simple, but you can find yourself spending hours and not coming up with a suitable solution.   Our approach will be use some basic math: *Pythagoras'* theorem and *similar triangles* to find a relationship where we can guess X; then see if our guess gives us the correct height, 10 feet.

Manually this is not a great solution, because a poor guess will us take a long time---and that does not include computational errors. If write down the steps for a single attempt (that is, create an algorithm), then we can rewrite those step in a programming language such as C++, place them in a loop and let the computer do the [brute] repetitive work.

The twin-ladder Puzzle lab

Step 1:
Name parts of the figure



Step 2:
Try and find the relationship (connections) between ladder lengths and the alley width using *Pythagoras' rule*:

$$A^2 + X^2 = 20^2 \quad \text{and} \quad B^2 + X^2 = 30^2$$

Now we rewrite both relationship solving for **A** and **B**, to reveal how their relationship to X (i.e., alley width); thus if we guess an **X** and plug-in its value we see how close it is to **A** and **B**.

$$A^2 = 20^2 - X^2 \quad \text{and} \quad B^2 = 30^2 - X^2$$

Step 3:
Okay, we can find a common X (maybe), now we need to ensure that our X also satisfies the intersection height (H); here we use similar triangles to establish this relationship.

$$A/H = X/C \quad \text{and} \quad B/H = X/(X-C)$$

Solve both equations for C, and then set them equal; thus, removing C.

The twin-ladder Puzzle lab

Now ***you*** solve for the height, ***H***, in terms of ***A*** and **B**. You will need to build this relationship***,***
H = f(A,B). [Hint: use the algebra].

Step 4:
You now have the following relationships:

A = p(X) = sqrt(20^2 - X^2)
B = q(X) = sqrt(30^2 - X^2)
H = f(A, B)

Begin to refine/detail our Algorithm:

A. Let's try guessing by adding or subtracting a value to X, call it DX, and say DX = 0.125. We code his with: `const double DX = 0.125;`
B. Guess at X, say **X= ???;**
C. Given X Compute A=p(x), and B=q(x)
D. Compute H = f(A,B).
E. How close is H to 10?
   a. Closeness = H -10
   b. if (closeness almost 0) then Bingo we've found X, go to step H
F. create a new guess: X = X + DX
G. go to step 2C
H. The value in location X holds our approximate solution.

LAB:

Complete a C++ solution to this problem.
Design a program that is "easy" to understand.

Questions:

How might Almost be implemented?
A naïve solution is something like:
Select a value for almost zero, say, 0.125, then in code:
`const double DELTA = 0.125;`
Now you can express *almost* in code: `abs(closeness)<=DELTA`

The twin-ladder Puzzle lab

What is our closeness does not change and never equals zero?

This is actually asking if our algorithm converges on a solution or is if it is not-stable (e.g., cycles under and over the solution, but never near enough to be an acceptable solution). We can keep track of the OLD closeness, with an extra variable called Old_closeness, and can initialize it to zero (i.e., Old_Closeness =0.0 at the beginning of the algorithm) Now we can say that if (Closeness == Old_Closeness) then we are done too.

Is there a better way to generate a new X ?

Yes. Invent one.

Imagine starting with a large +DX then decreasing it the closer we get, and if we overshoot, create a -DX, and if we undershoot create a +DX