Homework

16

What's the Point

For this assignment define your own user-defined structure type, named **Point** that has two **int** data members, **x** and **y**.

Then, write the functions:

- → **get()** and **print()** which perform I/O on **Point** objects. **print()** displays the **Point** in the form **(x, y)** while **get()** reads a **Point** in the form **x, y** where the space is optional, but the comma is not.
- → **double distanceBetween(const Point& a, const Point& b)** computes the distance from **a** to **b**.
- → **Point midpoint(const Point& a, const Point& b)** computes the point that is halfway between **a** and **b**.

Then, you'll create a new type named **Triangle** containing three **Point** members, and finish up by writing a function to computes the perimeter of a **Triangle**.

Implementation Notes

The header file contains prototypes for each function, not the ***Point*** or ***Triangle***.

- Define the **Point** type as a **structure type** in the header file where noted. Make sure they are in the order **x** and then **y**.
- Where requested define a **Triangle** type. It has three data members, **p1**, **p2** and **p3**, of type **Point** and which must appear in this order.

Type **make**. You should have no compiler errors, only linker errors.

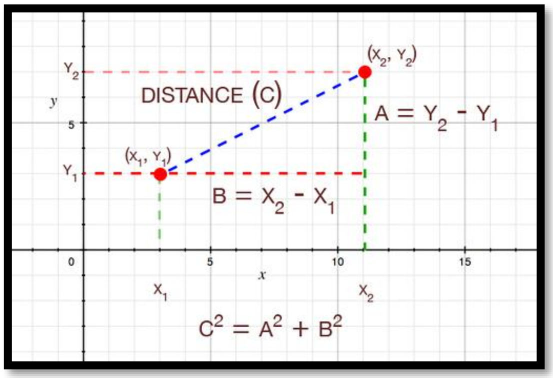
Writing Functions

Add the implementations for the functions in `h16.cpp`.

- The `print()` function should print `Point` objects as `(x, y)`.
- With `get()`, input is entered as ***x, y*** where `x` and `y` are `int`, separated by a comma and optional spaces. `get()` must **read and discard** the `char` value separating the two `int` values. If you use formatted input (`>>`) then the spaces will not be a problem. Don't worry about exceptions.

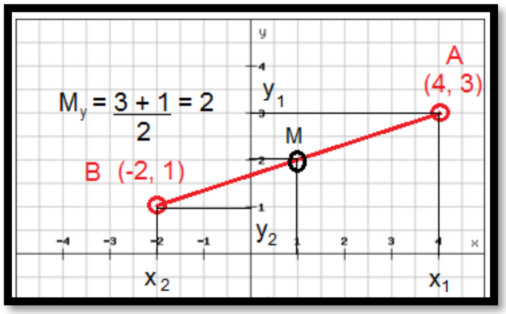
The distanceBetween Function

To find the distance, use the **Pythagorean** formula for finding the hypotenuse. Since you'll need the `sqrt()` function to do that, you have to be sure to include `<cmath>`.



The midpoint Function

The **midpoint** between the two `Points` `a` and `b`, is the `Point` on the distance line, equi-distant from `a` and `b` as shown here. The formula to calculate the midpoint is simply the sum of `a.x` and `b.x` divided by **2**, plus the sum of `a.y` and `b.y` divided by **2**.



Do the `perimeter()` function on your own. **Ask questions on Piazza if you get stuck.**