

Classes (Lab09)

 cse.msu.edu/~cse232/Labs/lab09.html

Remote Access

Often you'll want to be able to connect to and run programs on computers that you don't have a keyboard attached to. You already do this everytime to do anything on a network or over the internet. The primary means that you use to connect to computers at the command line is a tool called 'ssh'. SSH allows you to authenticate (usually your username and password) to a remote computer so you can access its command line.

The 'scp' command allows you to copy files to and from your local computer with a remote one.

The remote computer that you all have access to is called `chuck.egr.msu.edu`, and your username is your MSU Net ID and your password is your EGR password (not your CSE password). Note, because this class has transitioned away from using DECS servers to Mimir, you may be able to access the Chuck server. Also, certain countries are firewalled as well unfortunately.

Read this [tutorial from The New Stack](#)



Show your TA the hypothetical command you would run to copy a file on your laptop's Downloads folder named 'test.png' to your CSE account's Desktop.

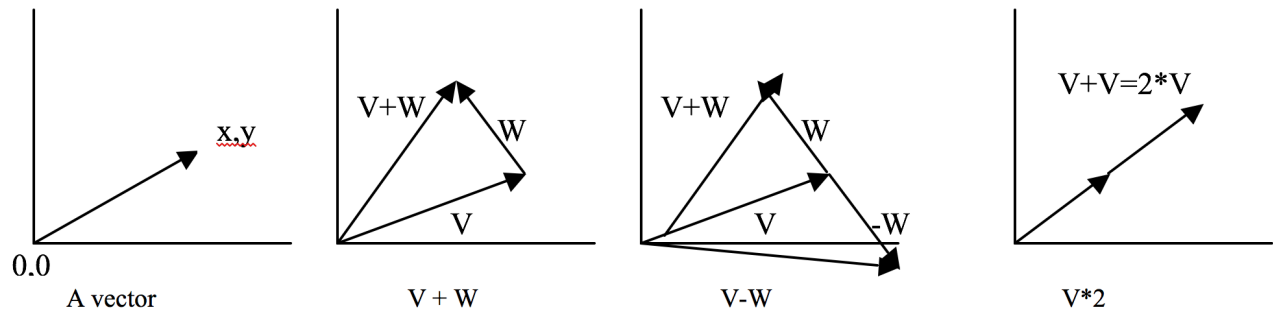
Lab Assignment

The Problem

We are going to work on making our own data structures using a C++ `struct`. Specifically we are going to create both data members and member functions. We are going to make a 2D MathVector `struct`. Too keep it straight, these are the mathematical entitles called Vectors (a geometric entity with direction and magnitude).

Some Backgroud

So if you don't remember, here is a little background on two-dimensional vectors. A vector is basically an arrow that has a magnitude (a length) and a direction (an angle with respect to typically the x axis). It usually is represented as an x,y pair, where the origin of the vector is assumed to be at 0,0 and the head of the vector is as the listed x,y pair.



Here are some of the operations you can perform on your new `MathVector struct` .

- `MathVector` addition. If V_1 is (x,y) and V_2 is (a,b) , the $V+W$ is $(x+a,y+b)$, returning a new `MathVector`
- `MathVector` multiplication by a scalar integer type. If V_1 is (x,y) , then $V*n$ is $(x*n,y*n)$, returning a new `MathVector`
- `MathVector` multiplication with another `MathVector` . There are two possibilities, dot product or cross product. We'll do dot product. If $V=(x,y)$ and $W=(a,b)$, then $V*W = x*a + y*b$, a scalar. Thus the dot product returns a scalar type `long` , not a `MathVector`
- `MathVector` magnitude (no parameters needed). The magnitude based on the Pythagorean theorem for a $V=(x,y)$ says that the magnitude is $\sqrt{x^2 + y^2}$. Returns a `double`

Your Tasks

Take this header file (`math_vector.h`) and this main file (`main.cpp`). You need to create the implementation file (named "math_vector.cpp") that provides the implementation for all of the member functions.

The parts of the `MathVector struct` . Data members are:

- `long x`
- `long y`


Constructors are:

- default constructor
- two args, each a long, constructor: first arg is the x value, the second is the y value. No defaults.

Make the following **regular function** (not a member):

```
string vec_to_str(const MathVector &v) . Single arg a const ref to
MathVector . Returns a string representation of the MathVector in the format:
" x:y "
```

The member functions are:

- `MathVector add(const MathVector&)` . Single arg a const ref to `MathVector` . Adds two `MathVectors` as described. Returns a new `MathVector` .
 Show your TA when your `MathVector` can perform addition.
- `MathVector mult(long)` . Multiplies a single `MathVector` element by a long as described. Returns a new `MathVector` .
- `long mult(const MathVector&)` . Single arg a const ref to `MathVector` . Multiplies the two `MathVectors` as a dot product, yielding a `long` as described above.
- `double magnitude()` . No args. Calculate the magnitude of the `MathVector` as described. Returns a `double` .

 Show your TA when your `MathVector` can perform all of the desired functions.