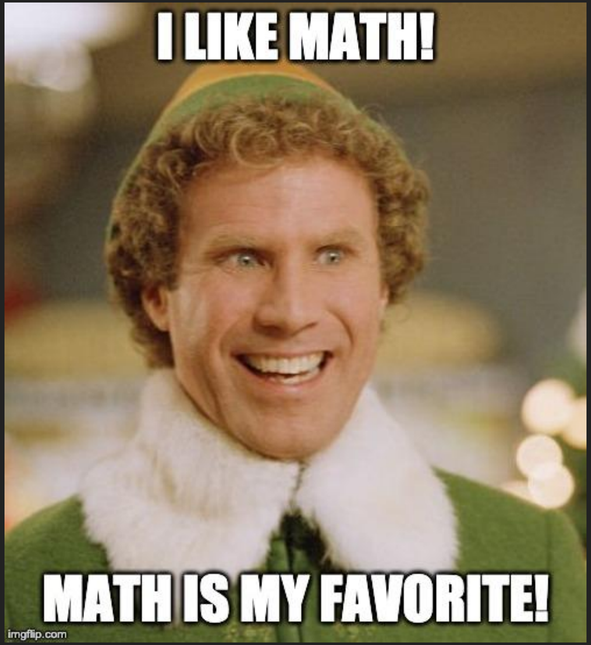
Math!



Where is the Empire State Building?

Empire State Building
103-story landmark
with observatories
E 34th St
E 35th St
E 33rd St
OWN

34th and 5th

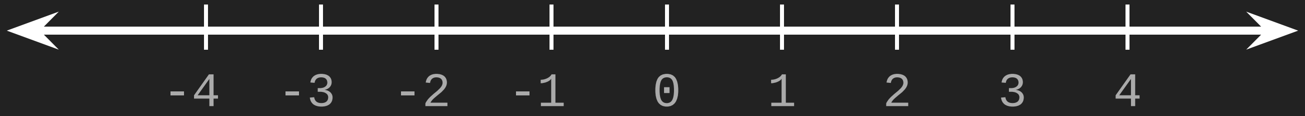
40.7484° N, 73.9857° W

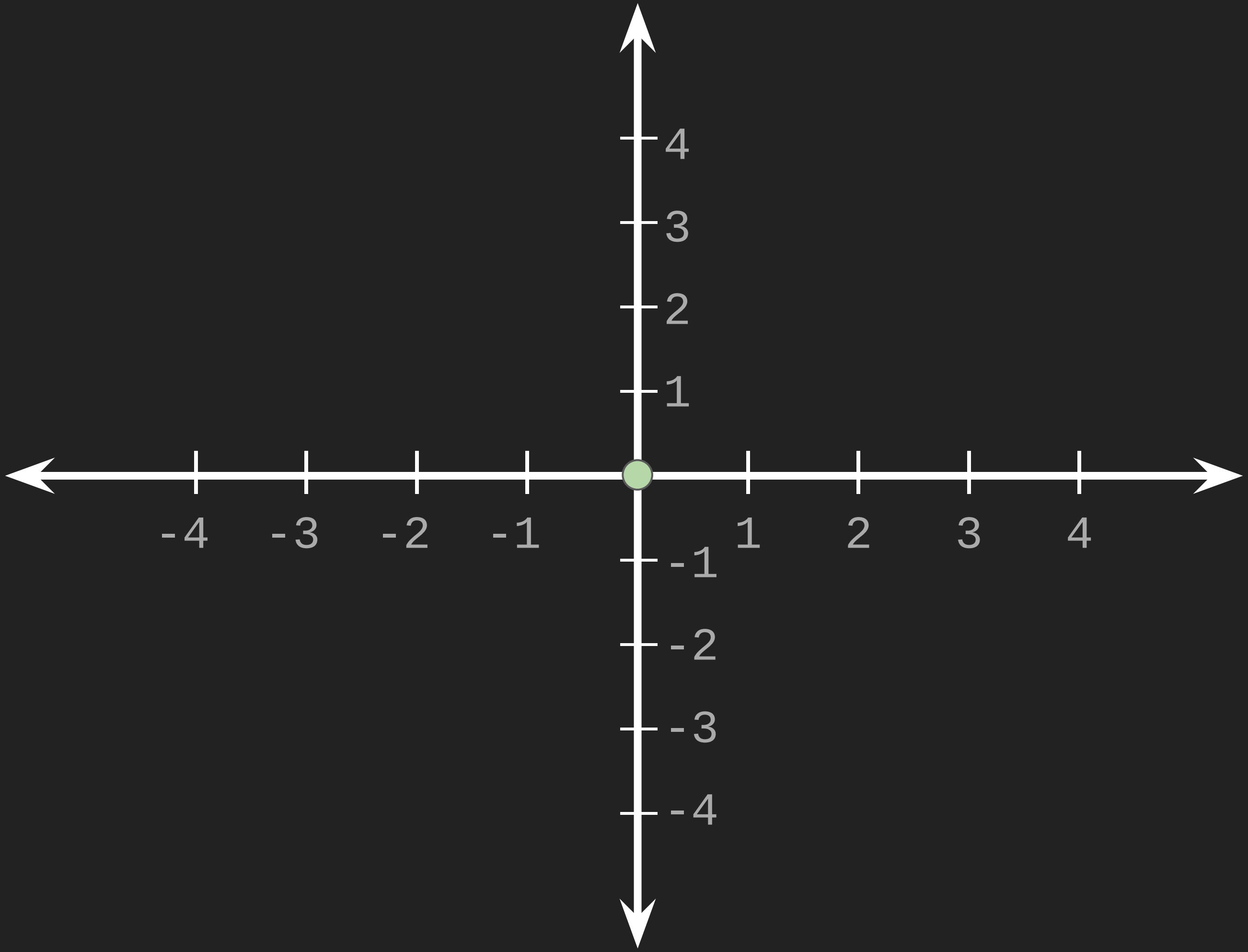
Cartesian Coordinate System

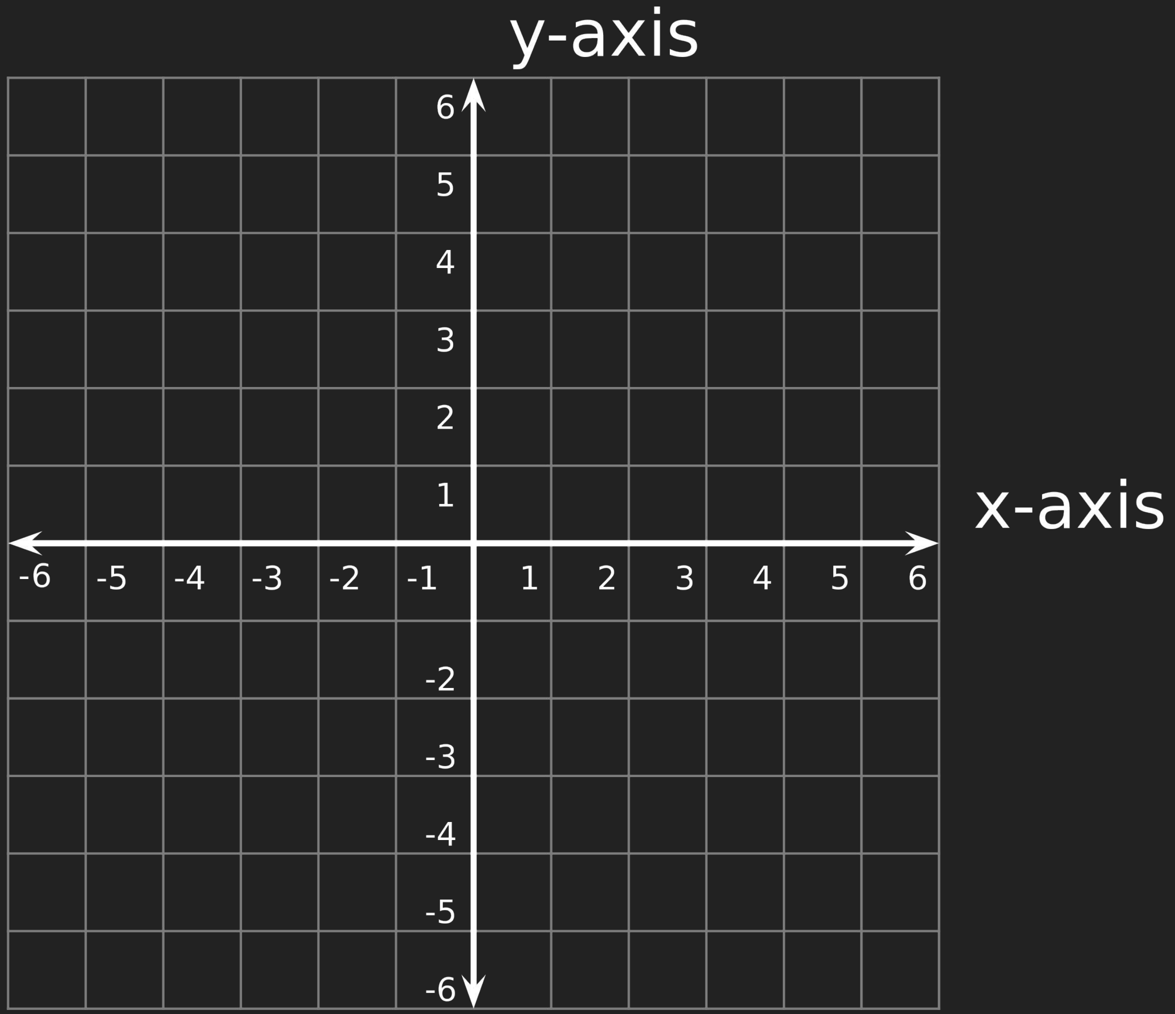
René Descartes

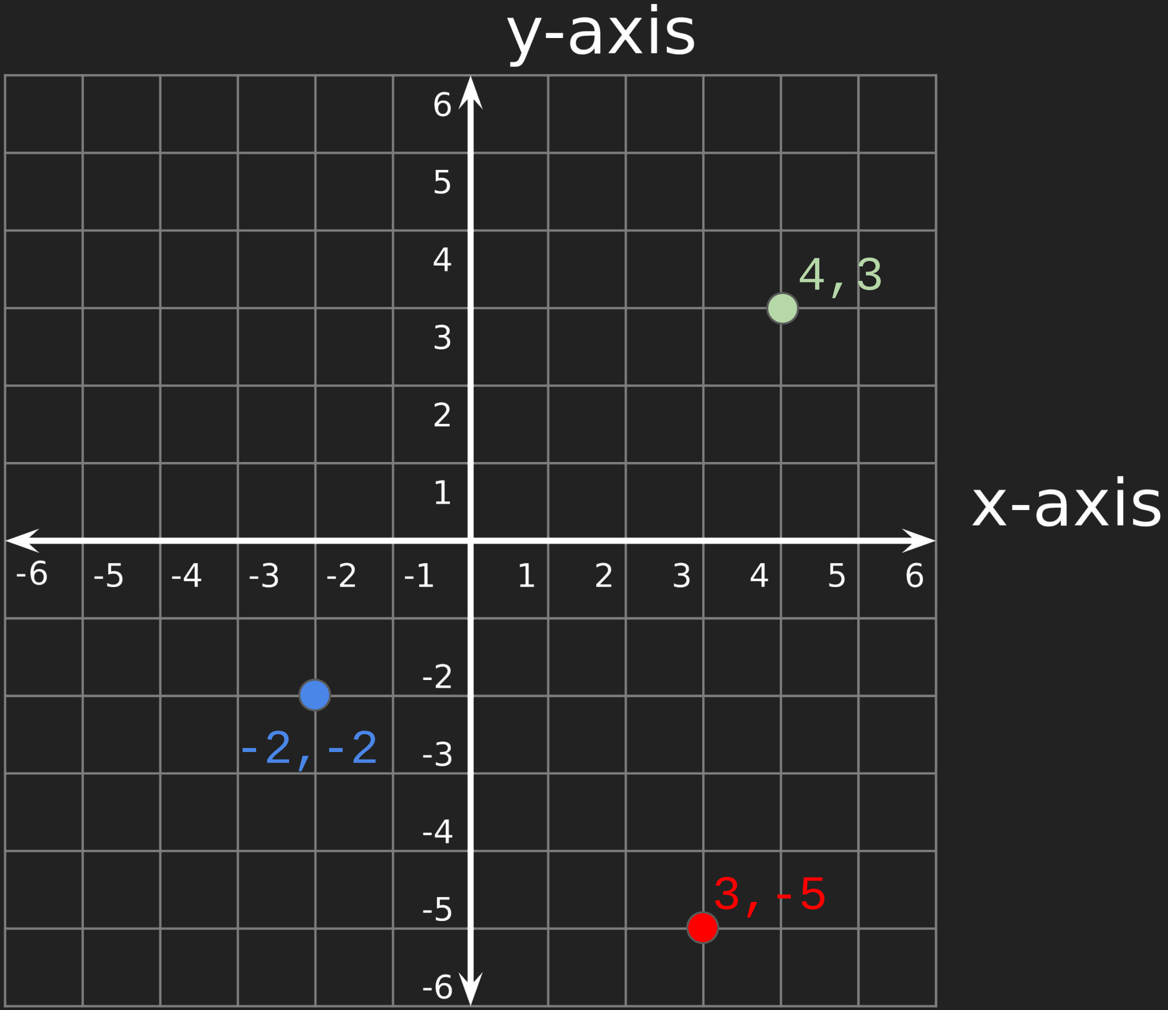
Not just a mathematician and scientist but also a philosopher:

“I think, therefore I am.”


-4
-3
-2
-1
0
1
2
3
4


4
3
2
1
-4
-3
-2
-1
1
2
3
4
-1
-2
-3
-4


y-axis
x-axis
6
5
4
3
2
1
-6 -5 -4 -3 -2 -1 1 2 3 4 5 6
-2
-3
-4
-5
-6


y-axis
x-axis
6
5
4
3
2
1
-2
-3
-4
-5
-6
-6
-5
-4
-3
-2
-1
1
2
3
4
5
6
4,3
-2,-2
3,-5

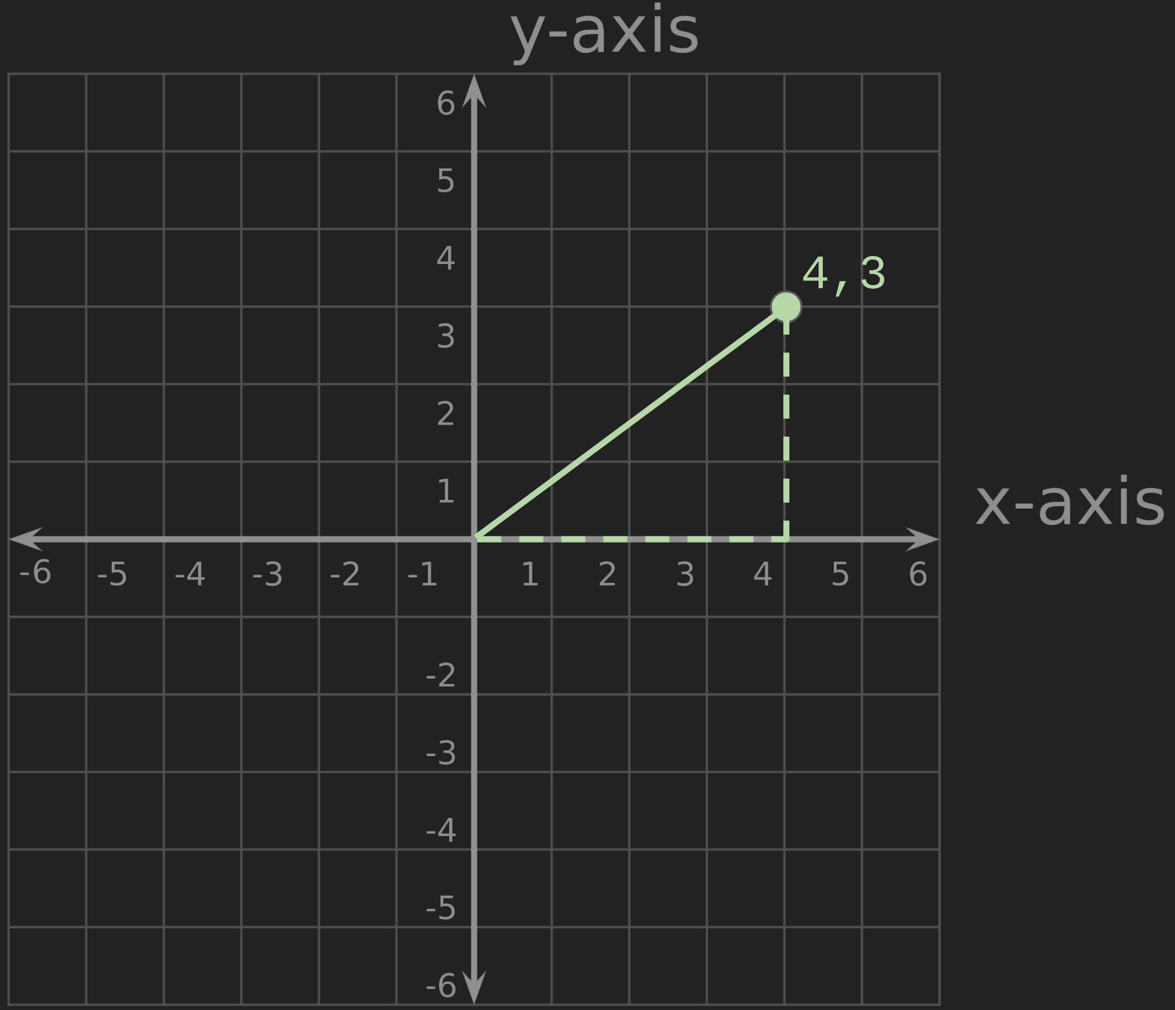
$$A^2 + B^2 = C^2$$

$$4^2 + 3^2 = C^2$$

$$16 + 9 = C^2$$

$$25 = C^2$$

5



A point is a location on a plane.

(we can’t exactly math a point)

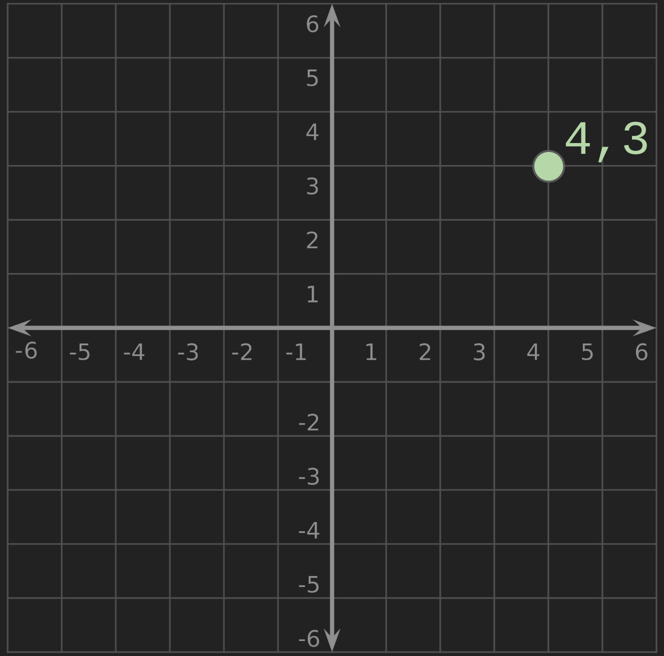
Empire State Building + The Brooklyn Bridge = ?

2 * Pace Plaza = ?

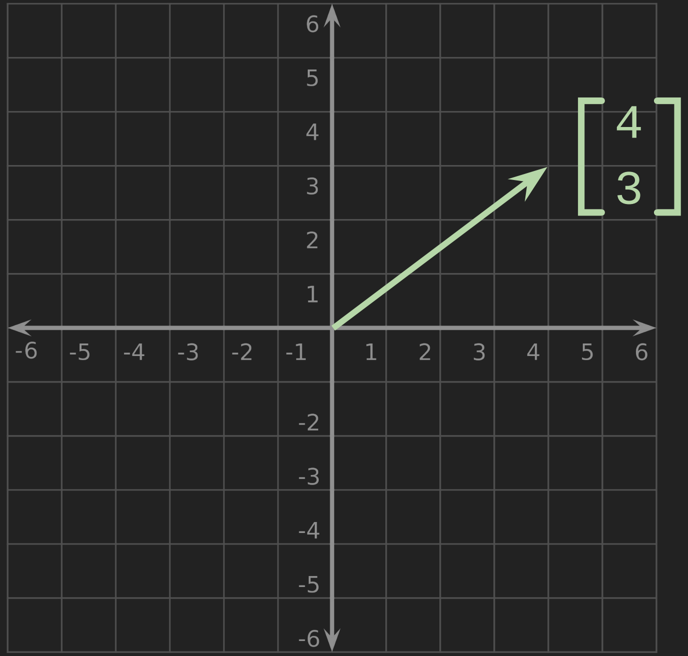
Rotate Carnegie Hall 45^0 = ?

Vectors!

Point

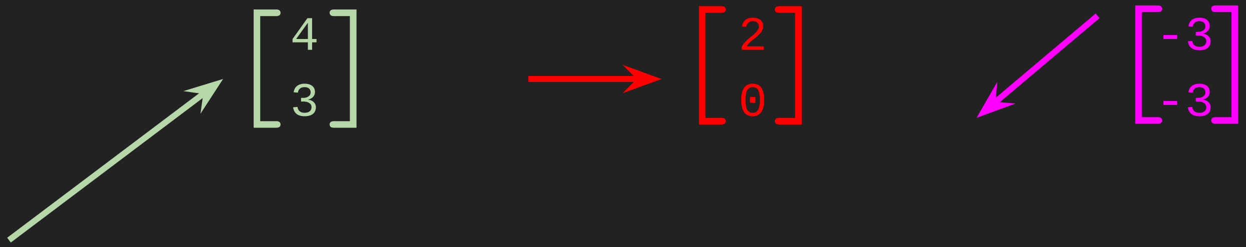


Vector



Vectors

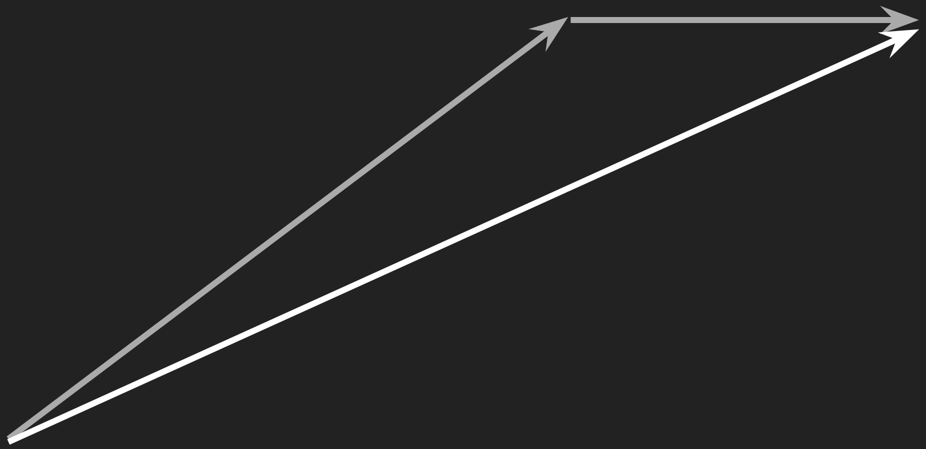
Vectors have a **direction** and **magnitude** (length).
Can be thought of as a **displacement**.



$\begin{bmatrix}4\\3\end{bmatrix}$ $\begin{bmatrix}2\\0\end{bmatrix}$ $\begin{bmatrix}-3\\-3\end{bmatrix}$

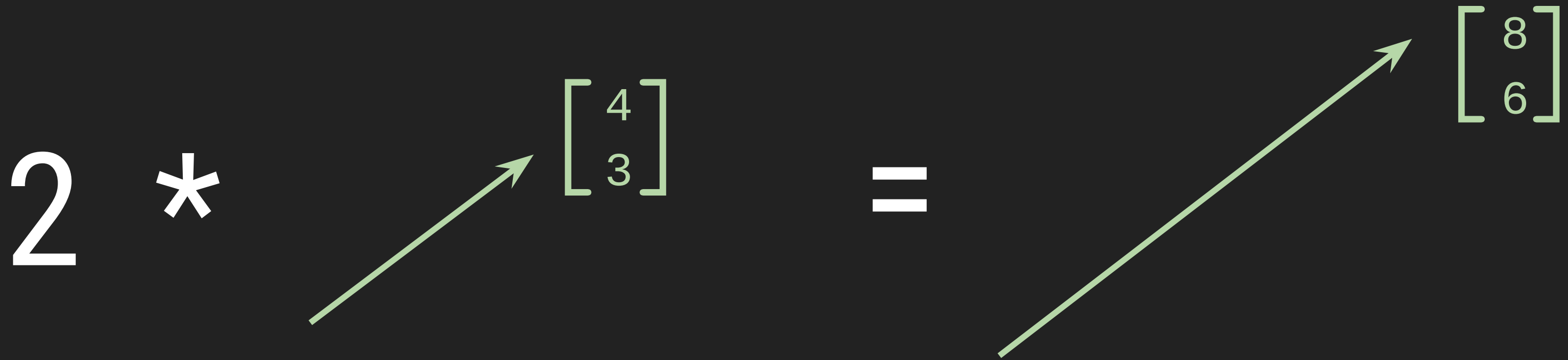
Vectors

We can **add** vectors.



Vectors

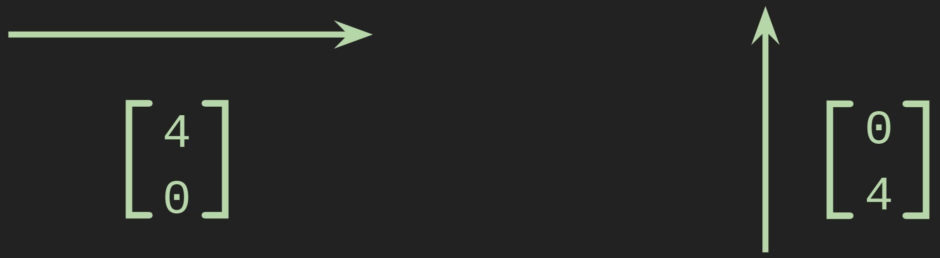
We can **multiply** a vector by a scalar.



$$2 * \begin{bmatrix}4\\3\end{bmatrix} = \begin{bmatrix}8\\6\end{bmatrix}$$

Vectors

We can **rotate** a vector.



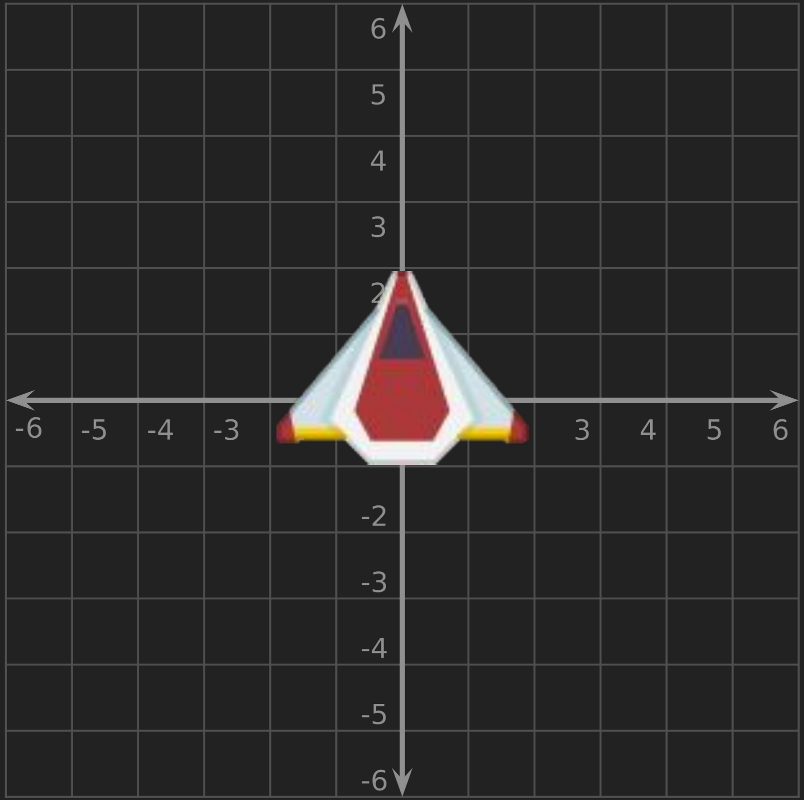
$\begin{bmatrix} 4 \\ 0 \end{bmatrix}$

$\begin{bmatrix} 0 \\ 4 \end{bmatrix}$

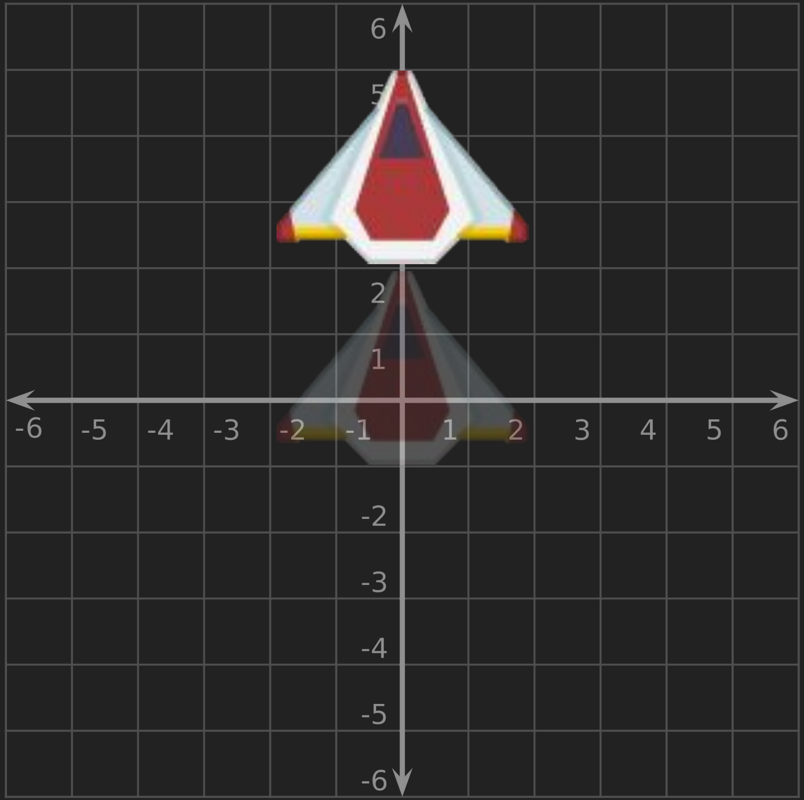
Transformations

Translation, Rotation and Scale

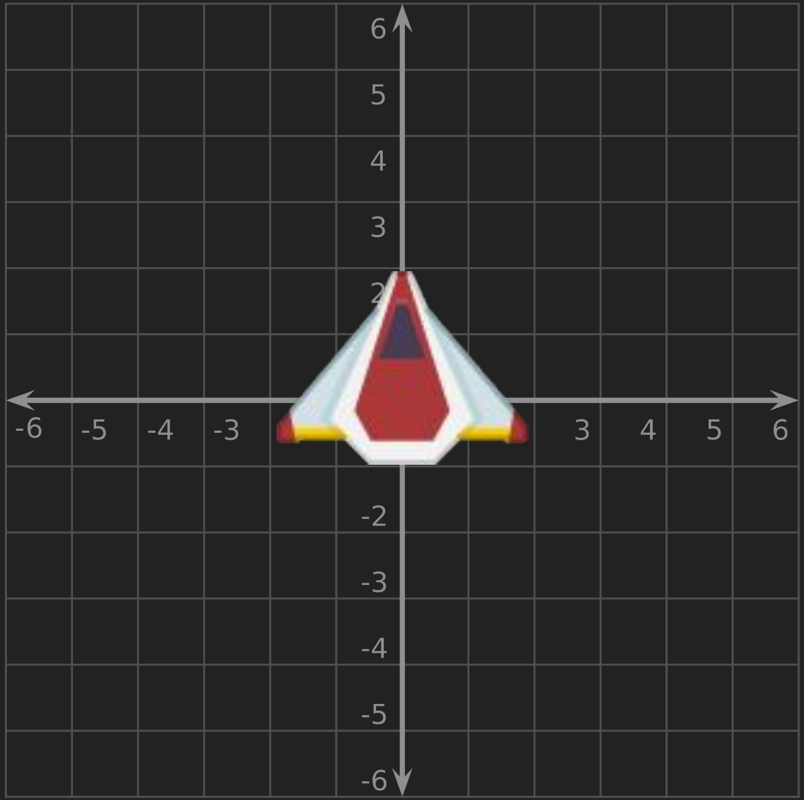
Translation



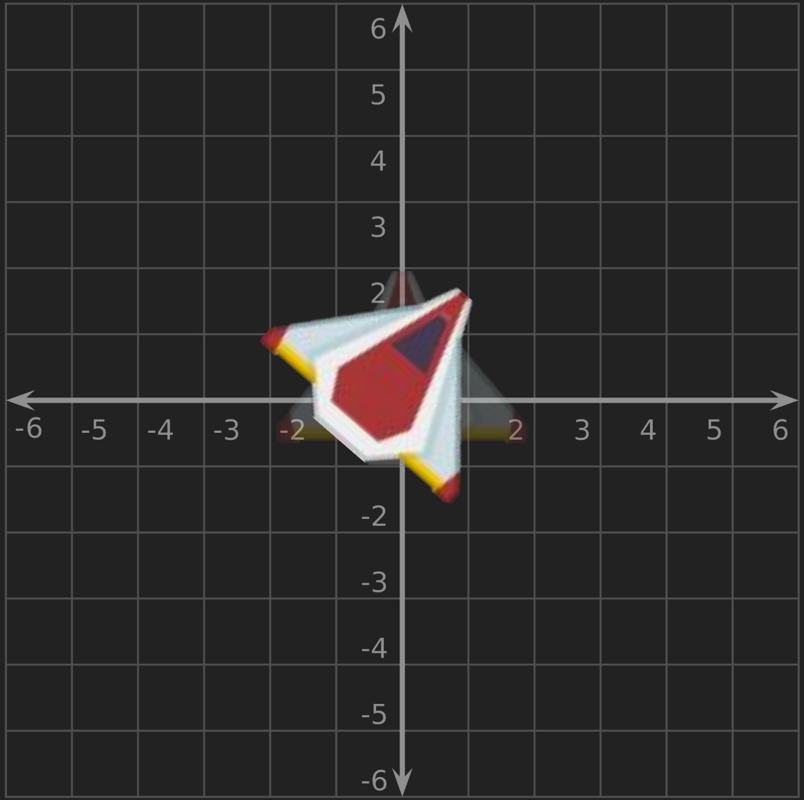
Translation



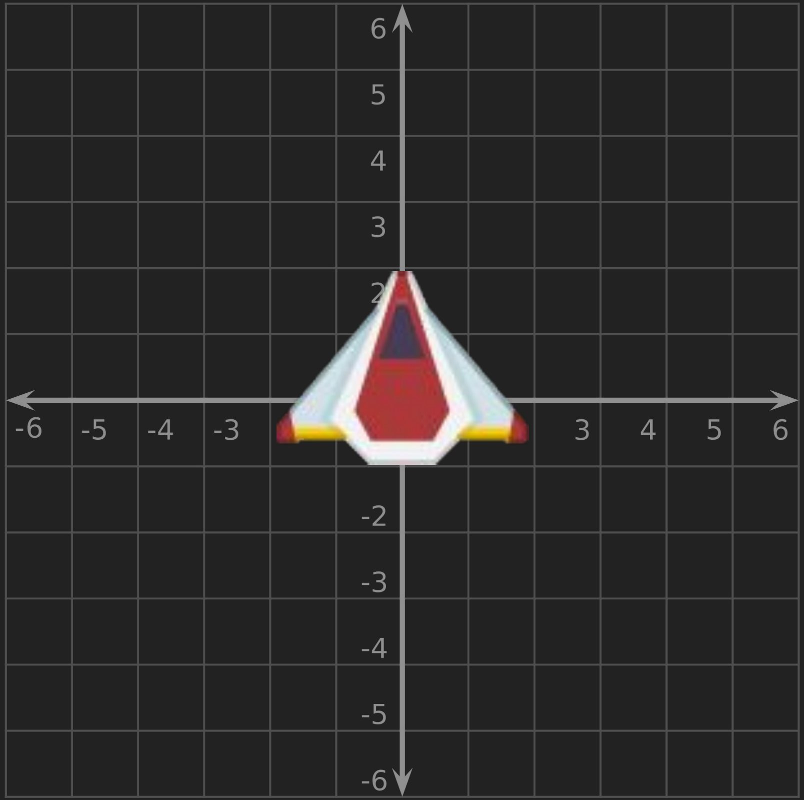
Rotation



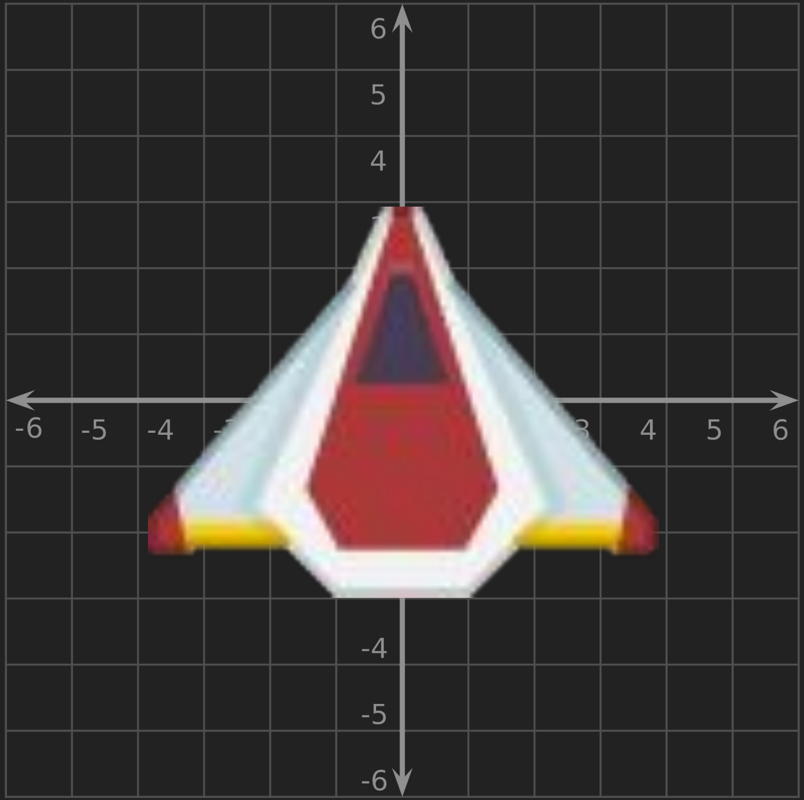
Rotation



Scale

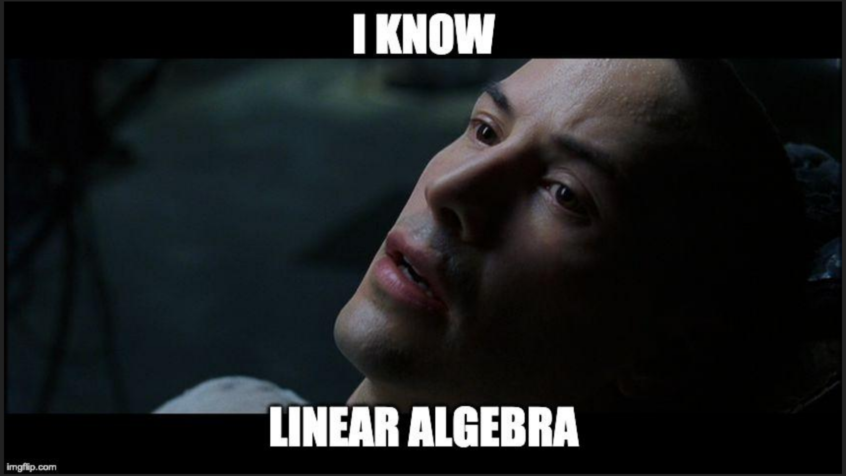


Scale



How do we do that stuff?

Matrices!



m-by-n Matrix

m rows -by- n columns

2x3 matrix

$$\begin{bmatrix} 1 & 2 & 0 \\ 4 & 3 & 2 \end{bmatrix}$$

3x2 matrix

$$\begin{bmatrix} 1 & 2 \\ 0 & 3 \\ 4 & 3 \end{bmatrix}$$

3x3 matrix

$$\begin{bmatrix} 1 & 2 & 7 \\ 0 & 3 & 2 \\ 4 & 3 & 1 \end{bmatrix}$$

We can treat a 2D Vector as a 2x1 matrix.

(this will come in handy later)

$$\begin{bmatrix} 4 \\ 3 \end{bmatrix} \qquad \begin{bmatrix} X \\ Y \end{bmatrix}$$

Matrix Operations

(and rules)

Matrix Addition

Add their corresponding entries.

They must be the **same size**!

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} + \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} A+E & B+F \\ C+G & D+H \end{bmatrix}$$

Matrix Addition

Add their corresponding entries.

They must be the **same size**!

$$\begin{bmatrix} 1 & 0 \\ 4 & 3 \end{bmatrix} + \begin{bmatrix} 2 & 3 \\ 1 & 3 \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 5 & 6 \end{bmatrix}$$

Matrix Subtraction

Subtract their corresponding entries.

They must be the same size!

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} - \begin{bmatrix} E & F \\ G & H \end{bmatrix} = \begin{bmatrix} A-E & B-F \\ C-G & D-H \end{bmatrix}$$

Matrix Subtraction

Subtract their corresponding entries.
They must be the same size!

$$\begin{bmatrix} 1 & 0 \\ 4 & 3 \end{bmatrix} - \begin{bmatrix} 2 & 3 \\ 1 & 3 \end{bmatrix} = \begin{bmatrix} -1 & -3 \\ 3 & 0 \end{bmatrix}$$

Transpose

Changes an MxN matrix into an NxM matrix.
Rows and Columns are switched.

$$\begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} \qquad \begin{bmatrix} A & D \\ B & E \\ C & F \end{bmatrix}$$

$$M \qquad\qquad\qquad M^T$$

Transpose

Changes an MxN matrix into an NxM matrix.
Rows and Columns are switched.

$$\begin{bmatrix}1 & 4 & 3\\ 2 & 3 & 5\end{bmatrix} \qquad \begin{bmatrix}1 & 2\\ 4 & 3\\ 3 & 5\end{bmatrix}$$

$$M \qquad\qquad\qquad M^T$$

Scalar Multiplication

Multiply each entry by S.

$$2 \times \begin{bmatrix} 1 & 2 & 0 \\ 2 & 3 & 4 \\ 1 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 2 & 4 & 0 \\ 4 & 6 & 8 \\ 2 & 0 & 4 \end{bmatrix}$$

Matrix Multiplication

(this is where things get interesting)

Matrix Multiplication

The number of **columns** in the first matrix must match the number of **rows** in the second matrix.

$$\begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} \times \begin{bmatrix} G & H \\ I & J \\ K & L \end{bmatrix} = ?$$

$$\begin{tikzcd} 2x3 \arrow[rr, leftrightarrow, bend right] & x & 3x2 & = & ? \end{tikzcd}$$

Matrix Multiplication

The result is the number of **rows** in the first matrix by the number of **columns** in the second matrix.

$$\begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} x \begin{bmatrix} G & H \\ I & J \\ K & L \end{bmatrix} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix}$$

2x3 x 3x2 = 2x2

$$\begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} \times \begin{bmatrix} G & H \\ I & J \\ K & L \end{bmatrix} =$$

$$\begin{bmatrix} A \text{x} G + B \text{x} I + C \text{x} K & \\ & \end{bmatrix}$$

$$\begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} \times \begin{bmatrix} G & H \\ I & J \\ K & L \end{bmatrix} =$$

$$\begin{bmatrix} AxG + BxI + CxK & AxH + BxJ + CxL \\ & \end{bmatrix}$$

$$\begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} x \begin{bmatrix} G & H \\ I & J \\ K & L \end{bmatrix} =$$

$$\begin{bmatrix} AxG + BxI + CxK & AxH + BxJ + CxL \\ DxG + ExI + FxK & \end{bmatrix}$$

$$\begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} \text{x} \begin{bmatrix} G & H \\ I & J \\ K & L \end{bmatrix} =$$

$$\begin{bmatrix} A\text{x}G + B\text{x}I + C\text{x}K & A\text{x}H + B\text{x}J + C\text{x}L \\ D\text{x}G + E\text{x}I + F\text{x}K & D\text{x}H + E\text{x}J + F\text{x}L \end{bmatrix}$$

We can use matrix multiplication to transform things!

Identity Matrix

A NxN matrix with 1 on the diagonal and 0 for the other values.

1x1

$$\begin{bmatrix} 1 \end{bmatrix}$$

2x2

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

3x3

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

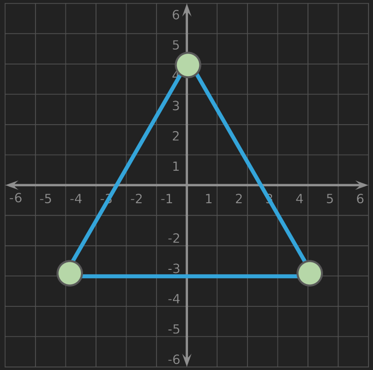
Multiplying by the Identity Matrix has no effect.

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} 1*X + 0*Y \\ 0*X + 1*Y \end{bmatrix}$$

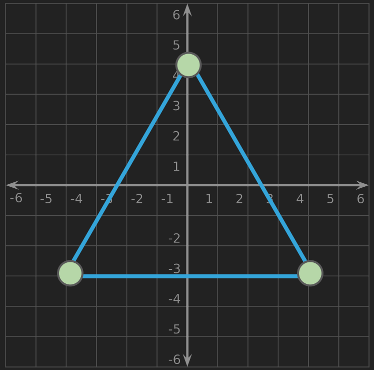
From our example code...

```
modelMatrix = glm::mat4(1.0f);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times$$



=



Scaling

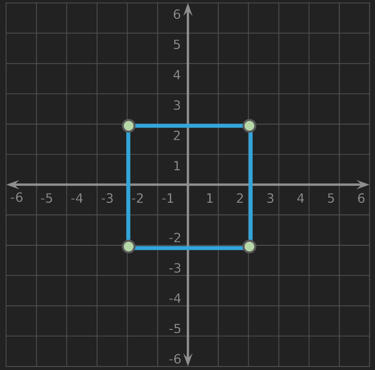
$$\begin{bmatrix} Sx & 0 \\ 0 & Sy \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} Sx*X + 0*Y \\ 0*X + Sy*Y \end{bmatrix} = \begin{bmatrix} Sx*X \\ Sy*Y \end{bmatrix}$$

$$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}\begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 2*4 + 0*3 \\ 0*4 + 2*3 \end{bmatrix} = \begin{bmatrix} 8 \\ 6 \end{bmatrix}$$

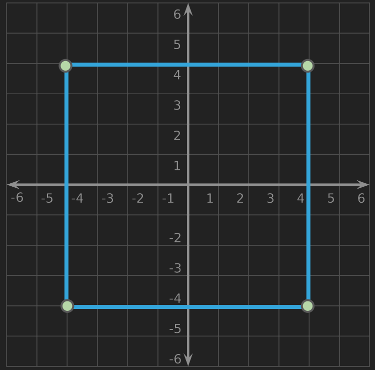
Scaling

(uniform and non-uniform)

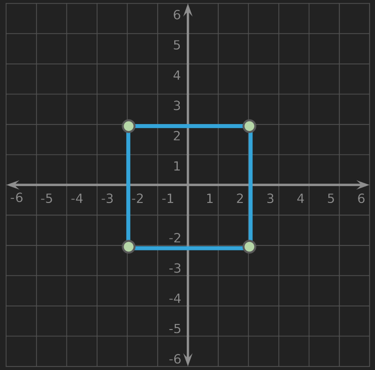
$$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \text{ x }$$



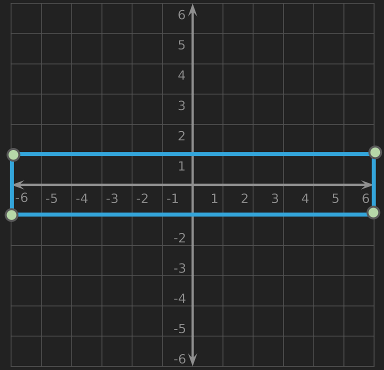
=



$$\begin{bmatrix} 3 & 0 \\ 0 & 0.5 \end{bmatrix} \text{ x }$$



=



Coding!

We transform the **matrix** that will transform the **vertices**.
This matrix is used by the **vertex shader**.

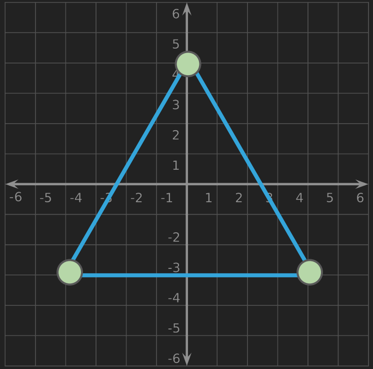
```
// Start off with Identity.
modelMatrix = glm::mat4(1.0f);

// Scale what we have so far.
modelMatrix = glm::scale(modelMatrix, glm::vec3(2.0f, 2.0f, 1.0f));
```

Rotation

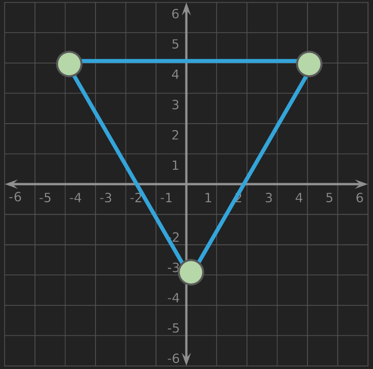
$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \cos\theta*X + -\sin\theta*Y \\ \sin\theta*X + \cos\theta*Y \end{bmatrix}$$

Rotation

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$



=



$$\theta = 180^{o}$$

Coding!

```
// Start off with Identity.
modelMatrix = glm::mat4(1.0f);

// Rotate what we have so far.
modelMatrix = glm::rotate(modelMatrix,
                           angle_in_radians,
                           glm::vec3(0.0f, 0.0f, 1.0f));
```

Translation

First we need to represent our 2D vector as a 3D vector.

(Homogeneous Coordinates)

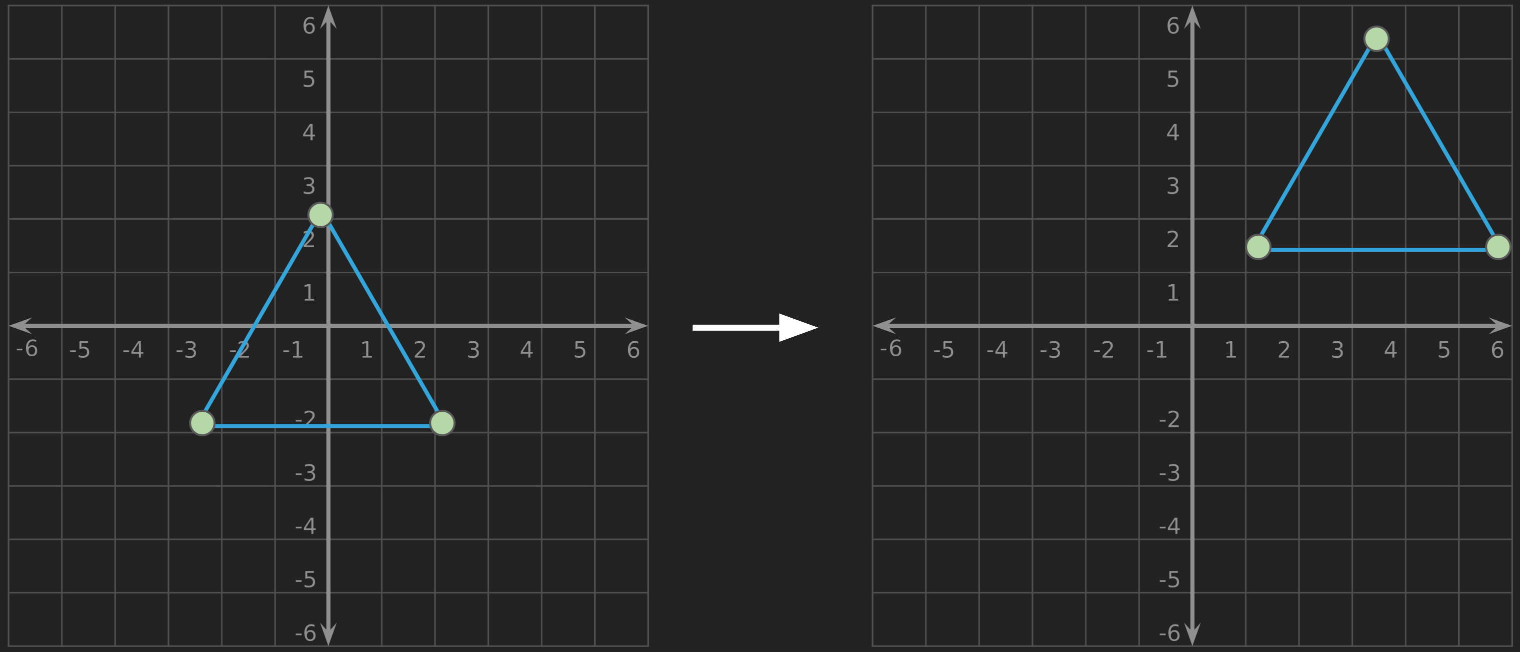
$$\begin{bmatrix} X \\ Y \end{bmatrix} \longrightarrow \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Translation

$$\begin{bmatrix} 1 & 0 & Tx \\ 0 & 1 & Ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \begin{bmatrix} 1*X + 0*Y + Tx*1 \\ 0*X + 1*Y + Ty*1 \\ 0*X + 0*Y + 1*1 \end{bmatrix}$$

$$\begin{bmatrix} X + Tx \\ Y + Ty \\ 1 \end{bmatrix}$$

Translation



Coding!

```
// Start off with Identity.
modelMatrix = glm::mat4(1.0f);

// Translate what we have so far.
modelMatrix = glm::translate(modelMatrix,
                           glm::vec3(3.0f, 2.0f, 0.0f));
```

mat4?

You may have noticed our code used **4x4** matrices.
That is because **everything is really 3D** and we need to use
Homogeneous Coordinates to transform 3D vectors.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \longrightarrow \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Let’s Code!