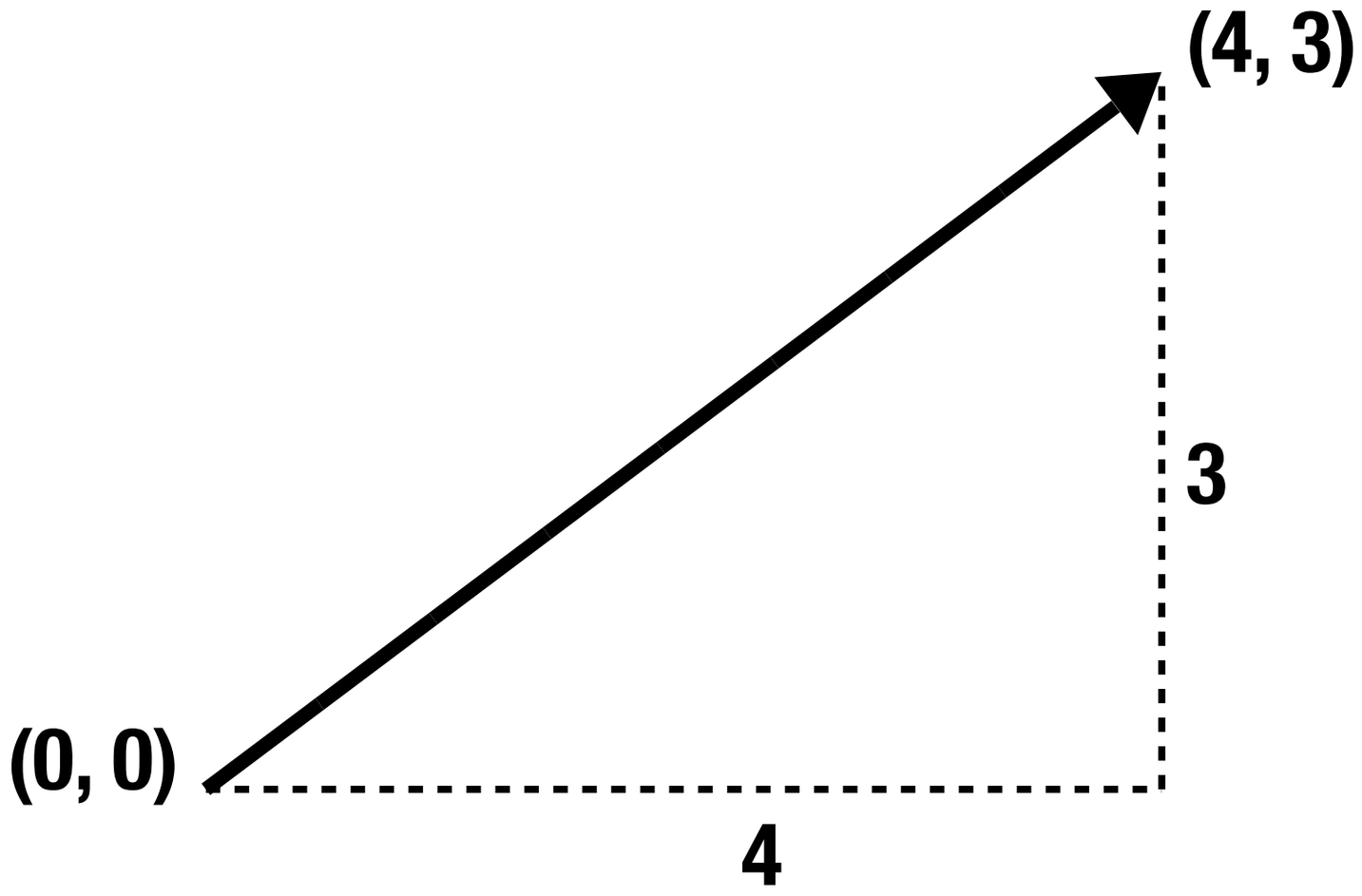
The Vector

step01_whatdoesavectorlooklike

a vector is made up of components.

a 2D vector is made up of X and Y.

vectors are often visualized as arrows point from (0,0) to (X,Y)

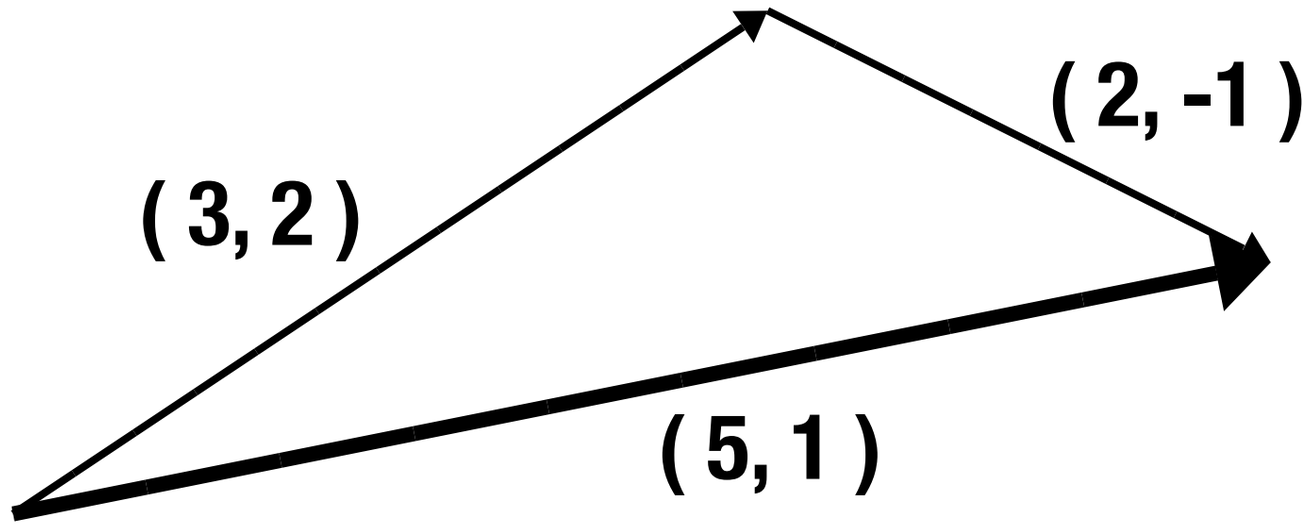


```
class Vector2f {

  float x = 0;
  float y = 0;
}
```

step02_addingvectors

adding 2 vectors means to connect the tip of one vector to the end of another.



(3, 2) + (2, -1) = (5, 1)

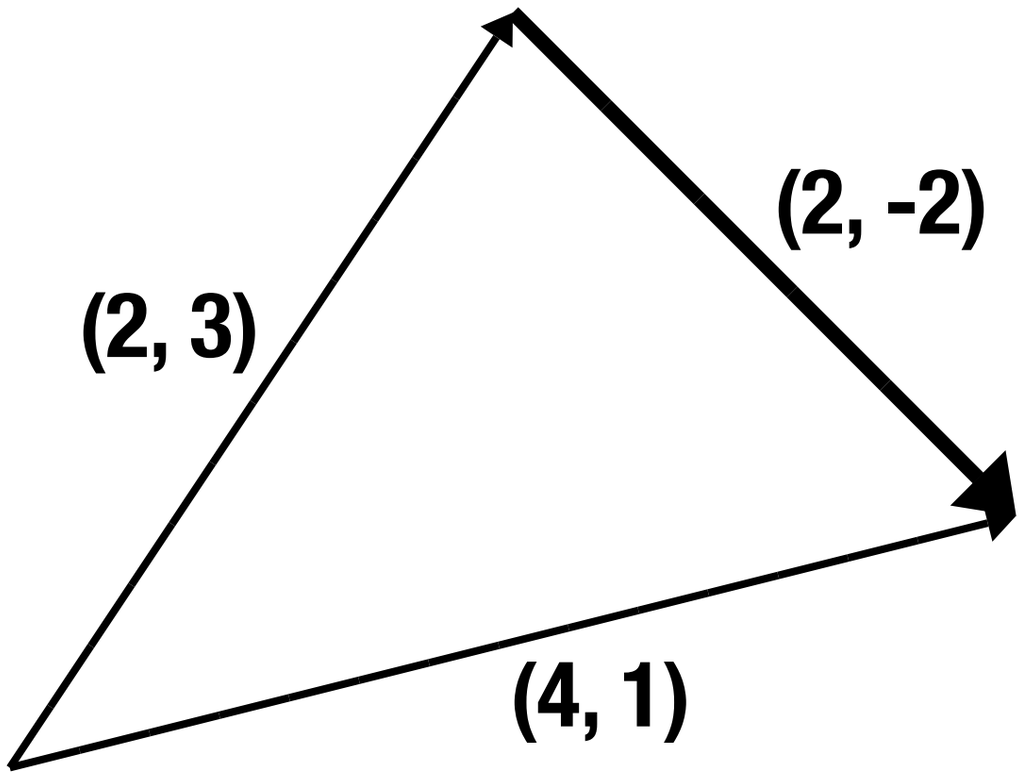
```
class Vector2f {

  float x = 0;
  float y = 0;

  void add(Vector2f theVector) {
    x += theVector.x;
    y += theVector.y;
  }
}
```

step03_subtractingvectors

the result of the subtraction of two vectors is a vector drawn between the two tips of the original vectors. the tip point towards the first of the two operands.



(4, 1) - (2, 3) = (2, -2)

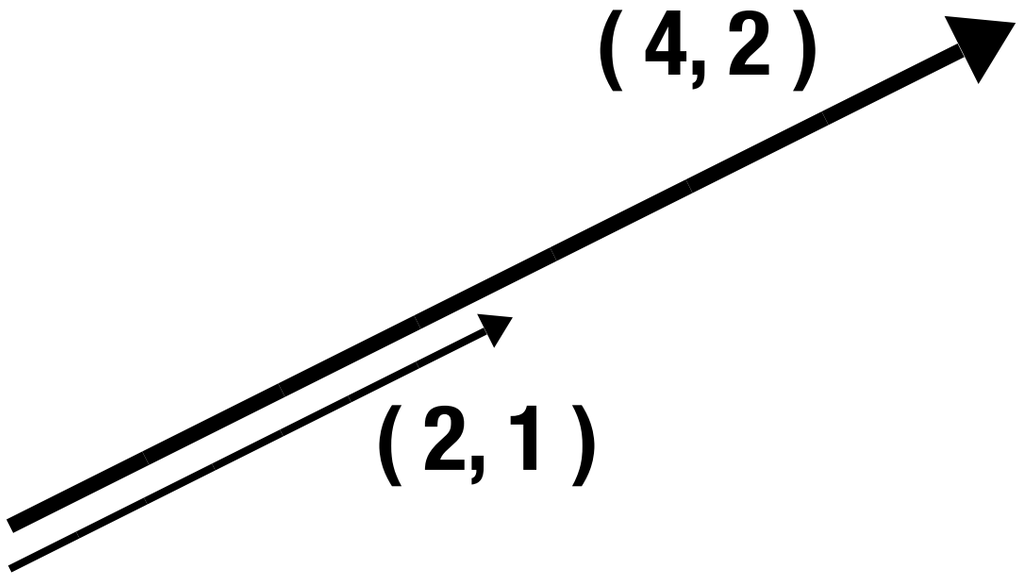
```
class Vector2f {

  float x = 0;
  float y = 0;

  void sub(Vector2f theVector) {
    x -= theVector.x;
    y -= theVector.y;
  }
}
```

step04_multiplyingvectors

multiplying a vector with a scalar (a number) changes the length of the vector.
numbers between 0 and 1 will give you a shorter vector.
numbers between 1 and ∞ will give you a longer vector.
negative numbers will change the direction of a vector.



(2, 1) * 2 = (4, 2)

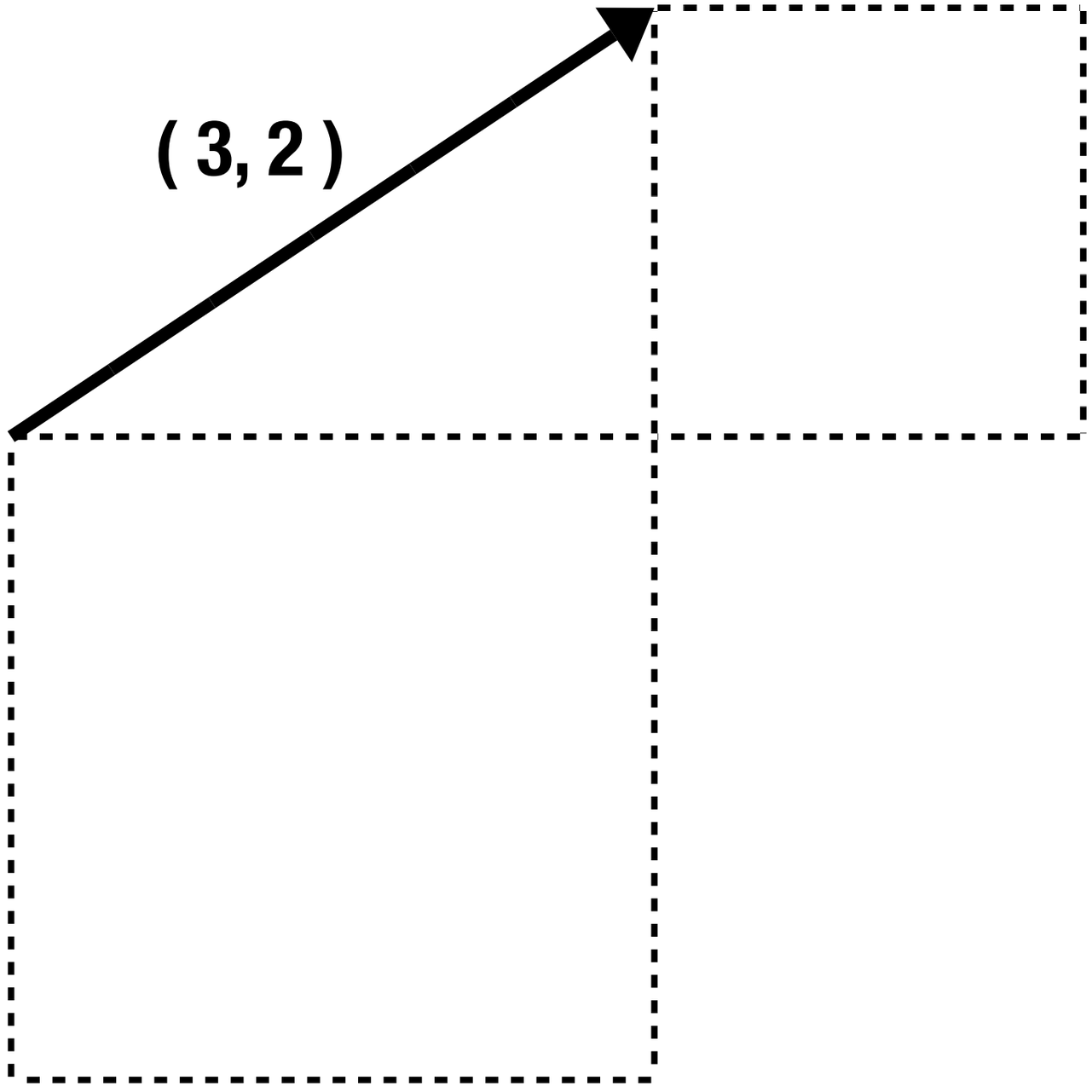
```
class Vector2f {

  float x = 0;
  float y = 0;

  void multiply(float s) {
    x *= s;
    y *= s;
  }
}
```

step05_lengthofvectors

the length of a vector is calculated with the help of the Pythagoras' theorem.



LENGTH = √ (3 * 3) + (2 * 2) = √13

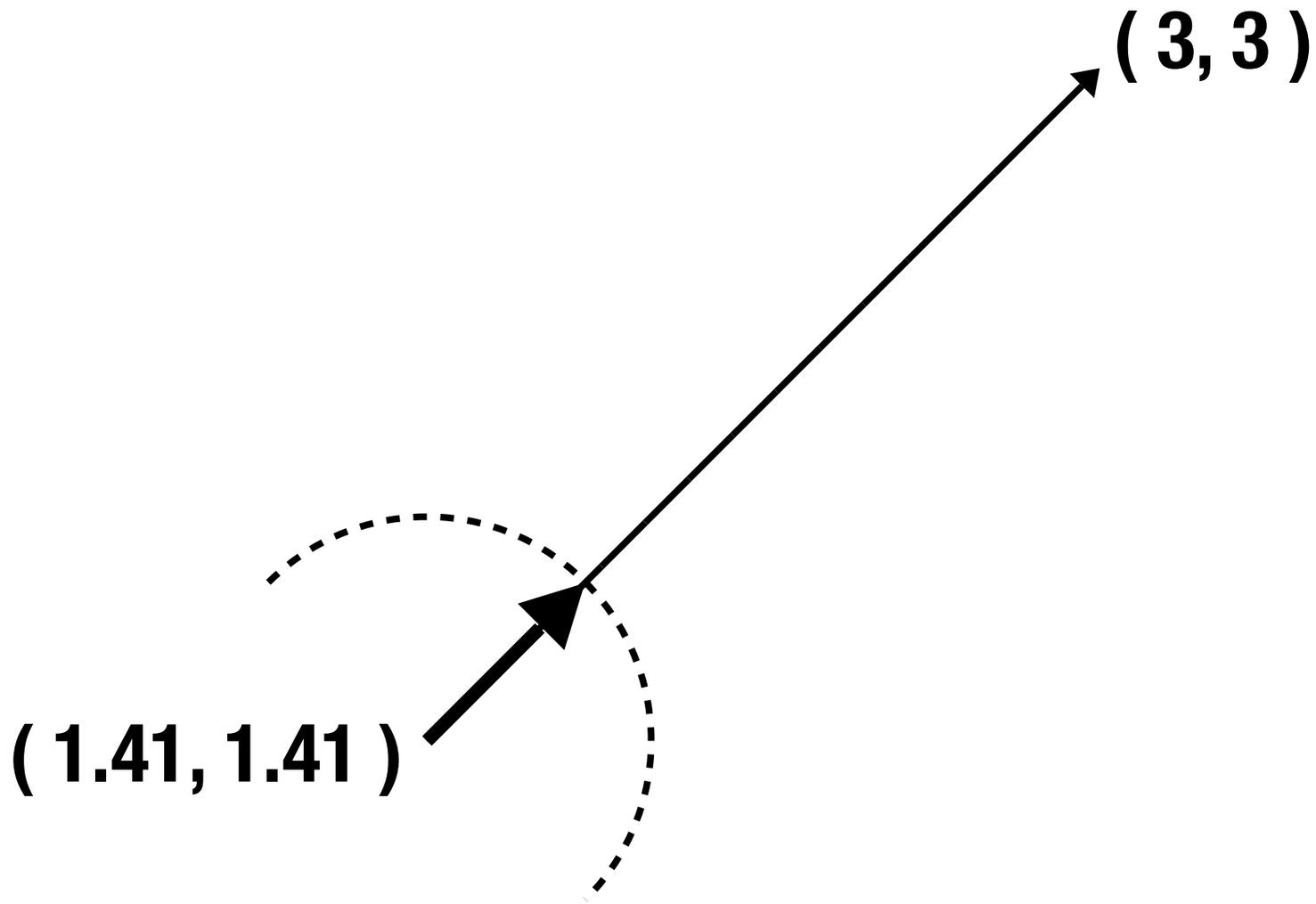
```
class Vector2f {

  float x = 0;
  float y = 0;

  float length() {
    float myLengthSquard = x*x + y*y;
    float myLength = (float)Math.sqrt(myLengthSquard);
    return myLength;
  }
}
```

step06_normalizingvectors

normalizing a vector results in a vector of the length of 1,
still pointing in the same direction.



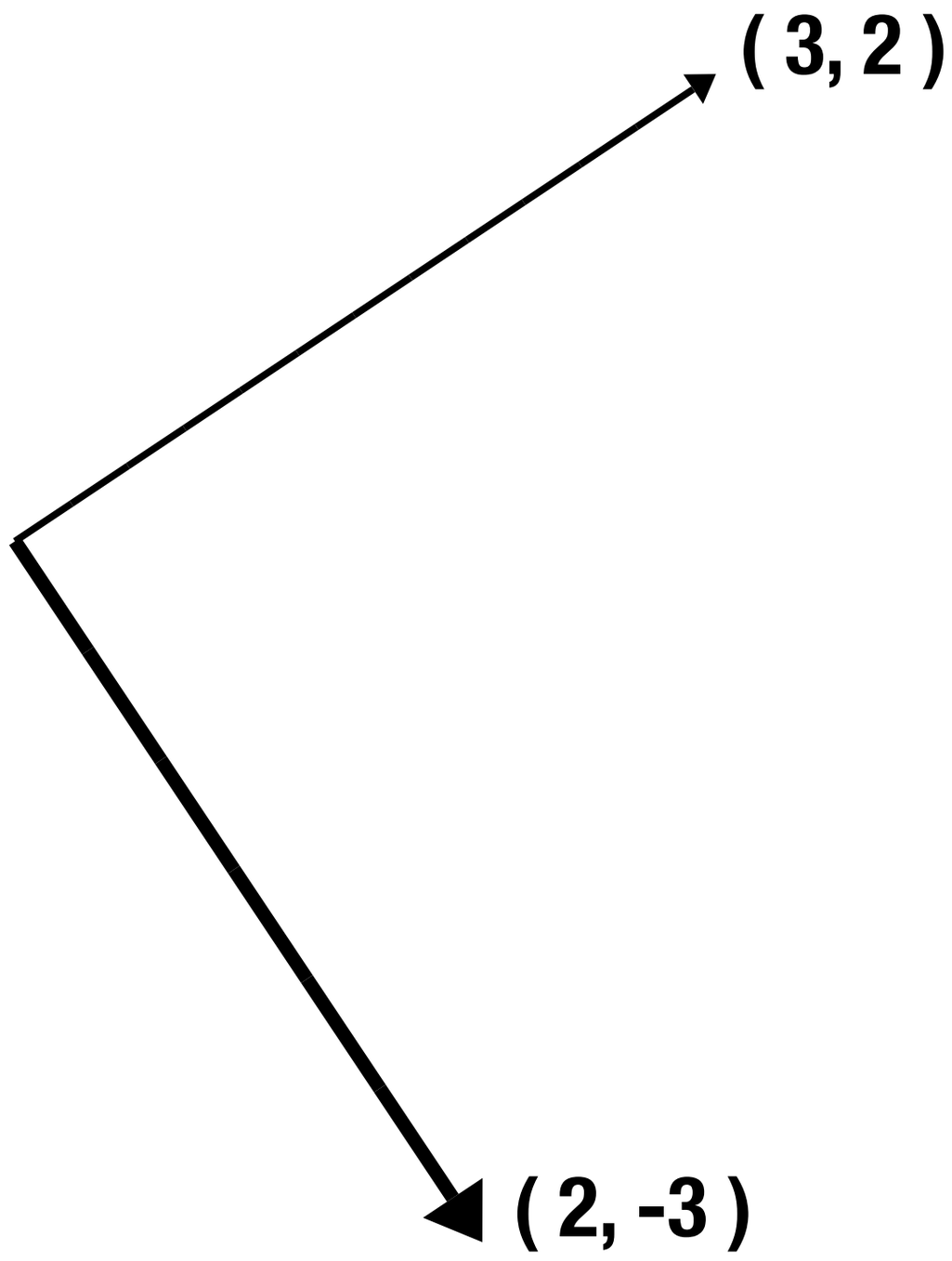
```
class Vector2f {

  float x = 0;
  float y = 0;

  void normalize() {
    float d = length();
    x /= d;
    y /= d;
  }
}
```

step07_crossproduct

the result of a cross product is a vector perpendicular to the original.
in 3D space it is defined as a vector perpendicular to 2 original vectors.
in 2D space it actually doesn t exist, but used here for the sake of vocabulary consitency.
think of it as the result of the vector (X,Y,0) and the vector along the Z-axis (0,0,1).



```
class Vector2f {

  float x = 0;
  float y = 0;

  void cross(Vector2f a) {
    x = a.y;
    y = -a.x;
  }
}
```

```
class Vector2f { // all together

  float x = 0;
  float y = 0;

  void add(Vector2f theVector) {
    x += theVector.x;
    y += theVector.y;
  }

  void sub(Vector2f theVector) {
    x -= theVector.x;
    y -= theVector.y;
  }

  void multiply(float s) {
    x *= s;
    y *= s;
  }

  float length() {
    float myLengthSquard = x*x + y*y;
    float myLength = (float)Math.sqrt(myLengthSquard);
    return myLength;
  }

  void normalize() {
    float d = length();
    x /= d;
    y /= d;
  }

  void cross(Vector2f a) {
    x = a.y;
    y = -a.x;
  }
}
```

MATRIX & QUATERNION FAQ
http://www.j3d.org/matrix_faq/matrfaq_latest.html