Quick Start - vectors32

Check Installation

The installation command:

```
sudo setup.py install
```

will probably put vectors32.py in the following directory on your ubuntu PC:

```
/usr/local/lib/python3.2/dist-packages/vectors32/
```

To check the installation, start python3.2 or higher as follows:

```
python3
```

Now execute the following commands from the Python shell:

```
>>> from vectors32 import vectors32
```

If exception was not raised, it is likely that the file vectors32.py with module vectors32 has been installed correctly.

Now change to the test directory to the place where you have expanded the tar ball:

```
cd vectors32/vectors32/tst/
python3 testVectors32.py
```

You are ready for the journey

Vector Algebra

Let us create two vectors **v1** and **v2**:

```
>>> from vectors32 import vectors32
>>> Vector = vectors32.Vector
>>> v1 = Vector(2.0, 3.0, 4.0)
>>> v2 = Vector(6.6, 5.5, 4.5)
```

Have we really created vectors? Let us see by printing them out:

```
>>> print(v1)
Vector(2.0, 3.0, 4.0)
>>> print(v2)
Vector(6.6, 5.5, 4.5)
```

Looks like we have! Let us add them:

```
>>> v3 = v1 + v2
>>> print(v3)
Vector(8.6, 8.5, 8.5)
```

Yes, vectors have been added!

Scalar Product

The syntax for scalar product is similar to multiplication of two scalars:

```
>>> res = v1 * v2
>>> print(res)
47.7
```

Yes, scalar product is a scalar and is equal to (2.0 * 6.6) + (3.0 * 5.5) + (4 * 4.5). which is indeed 47.7!

Vector (Cross) Product

The syntax for vector product is two vectors with '**' between them. Let us try it:

```
>>> v3 = v1 ** v2
>>> print(v3)
Vector(-8.5, 17.4, -8.799999999999997)
```

Yes, vector product is a vector. Check out the numerical value as an exercise. (The strange value of x component is the result of round off errors - it would be -8.8 in infinite precision.)

The components of a vector are simply as follows:

```
>>> v3.x
-8.5
>>> v3.y
17.4
>>> v3.z
-8.799999999999997
```

What about the vectors in two dimensions? If you want a vector in xy plane, simply do not specify the z component - the program sets a default value of zero for the unspecified component. You can simply continue with the same syntax for the planar vectors as for three dimensional vectors. Just bear in mind that a cross product will be a vector in the direction of omitted component.

The magnitude of a vector is given by the **size** property:

```
>>> length = v1.size
>>> length
5.385164807134504
```

Yes, the properties are accessible as if they were scalars. Vectors can be **normalized** to unit length:

```
>>> normal = v1.normalize
>>> print(normal)
Vector(0.3713906763541037, 0.5570860145311556, 0.7427813527082074)
```

The components of a normalised vector are its **direction cosines.** This can be useful in many applications.

Happy vector analysis!