Using a Simple Vector Class

In [1]:
```python
from skvectors import create_class_Simple_Vector
```

In [2]:
```python
# Create a 3-dimensional simple vector class

# The first argument is a string with the name of the class
# to be created.

# The number of elements in the iterable given as the second
# argument determines the number of dimensions for the class.

SVC = create_class_Simple_Vector('VC', 'IJK')

# Explicit alternative:
# SVC = \
#     create_class_Simple_Vector(
#         name = 'SVC',
#         component_names = [ 'I', 'J', 'K' ],
#         brackets = [ '<', '>' ],
#         sep = ', '
#     )
```

In [3]:
```python
# Create a vector by applying abs to the I-component of a vector
v = SVC(-2, 3, -4)
v.c_abs_I()
```

Out[3]:
```
VC(I=2, J=3, K=-4)
```

In [4]:
```
# Create a vector by applying unary minus to the K-component of a vector
v = SVC(2, 3, 4)
v.c_neg_K()
```
Out[4]:
```
VC(I=2, J=3, K=-4)
```

In [5]:
```
# Create a vector by applying unary minus to all the components of a vector except the K-component
v = SVC(2, 3, 4)
v.c_neg_bar_K()
```
Out[5]:
```
VC(I=-2, J=-3, K=4)
```

In [6]:
```
# Create a vector by applying unary plus to the J-component and the K-component of a vector
v = SVC(2, 3, 4)
v.c_pos_J_K()
```
Out[6]:
```
VC(I=2, J=3, K=4)
```

In [7]:
```
# Create a vector by adding 100 to the K-component of a vector
v = SVC(2, 3, 4)
v.c_add_K(100)
```
Out[7]:
```
VC(I=2, J=3, K=104)
```

In [8]:
```
# In-place addition of 100 to the K-component of a vector
v = SVC(2, 3, 4)
v.c_iadd_K(100)
v
```
Out[8]:
```
VC(I=2, J=3, K=104)
```

In [9]:
```
# Create a vector by subtracting 3 from the J-component of a vector
v = SVC(2, 3, 4)
v.c_sub_J(3)
```
Out[9]:
```
VC(I=2, J=0, K=4)
```

In [10]:
```
# In-place subtraction of 3 from the J-component of a vector
v = SVC(2, 3, 4)
v.c_isub_J(3)
v
```
Out[10]:
```
VC(I=2, J=0, K=4)
```

In [11]:
```
# Create a vector by multiplying all the components of a vector except none by 8
v = SVC(2, 3, 4)
v.c_mul_bar(8)
```
Out[11]:
```
VC(I=16, J=24, K=32)
```

In [12]:
```
# In-place multiplication of all the components of a vector except none by 8
v = SVC(2, 3, 4)
v.c_imul_bar(8)
v
```
Out[12]:
```
VC(I=16, J=24, K=32)
```

In [13]:
```
# Create a vector by raising the I-component of a vector to the power of 10
v = SVC(2, 3, 4)
v.c_pow_I(10)
```
Out[13]:
```
VC(I=1024, J=3, K=4)
```

In [14]:
```
# In-place raising the I-component of a vector to the power of 10
v = SVC(2, 3, 4)
v.c_ipow_I(10)
v
```
Out[14]:
```
VC(I=1024, J=3, K=4)
```

In [15]:
```
# Create a vector by true dividing none of the components of a vector by 0
v = SVC(2, 3, 4)
v.c_truediv(0)
```
Out[15]:
```
VC(I=2, J=3, K=4)
```

```
In [16]: # In-place true division of all the components of a vector by 10
         v = SVC(2, 3, 4)
         v.c_itruediv_bar(10)
         v

Out[16]: VC(I=0.2, J=0.3, K=0.4)
```

```
In [17]: # Create a vector by floor dividing all the components of a vector by 2
         v = SVC(2, 3, 4)
         v.c_floordiv_I_J_K(2)

Out[17]: VC(I=1, J=1, K=2)
```

```
In [18]: # In-place floor division of all the components of a vector by 2
         v = SVC(2, 3, 4)
         v.c_ifloordiv_I_J_K(2)
         v

Out[18]: VC(I=1, J=1, K=2)
```

```
In [19]: # Create a vector by applying modulus to all the components of a vector and 2
         v = SVC(2, 3, 4)
         v.c_mod_I_J_K(2)

Out[19]: VC(I=0, J=1, K=0)
```

```
In [20]: # In-place application of modulus to all the components of a vector and 2
         v = SVC(2, 3, 4)
         v.c_imod_I_J_K(2)
         v

Out[20]: VC(I=0, J=1, K=0)
```

```
In [21]: # Create a vector by multiplying the K-component of a vector by 100
         v = SVC(2, 4, 6)
         v.c_mul_K(100)

Out[21]: VC(I=2, J=4, K=600)
```

In [22]:
```python
# In-place multiplication of the K-component of a vector by 100
v = SVC(2, 4, 6)
v.c_imul_K(100)
v
```
Out[22]:
```
VC(I=2, J=4, K=600)
```

In [23]:
```python
# Create a vector by applying several operations to the components of vectors
v = SVC(2, 3, 4)
f = v.c_mul_K
f(10).c_add_bar(88).c_mul_I_J(88).c_sub_bar_J_K(100000).c_neg_K()
```
Out[23]:
```
VC(I=-92080, J=8008, K=-128)
```

In [24]:
```python
# Create a vector by rounding the components of a vector to 3 decimals
v = SVC(2.22222, 4.44444, 6.66666)
round(v, ndigits=3)
```
Out[24]:
```
VC(I=2.222, J=4.444, K=6.667)
```

In [25]:
```python
# Create a vector by rounding the components of a vector to integer value
v = SVC(2.222, 4.444, 6.666)
round(v)
```
Out[25]:
```
VC(I=2.0, J=4.0, K=7.0)
```

In [26]:
```python
# Create a vector by rounding the components of a vector
v = SVC(-55555555.5, -33333333.3, 55555555.5)
round(v, -4)
```
Out[26]:
```
VC(I=-55560000.0, J=-33330000.0, K=55560000.0)
```

In [27]:
```python
# Create a vector by applying unary minus to a vector
v = SVC(-3, 4, 5)
-v
```
Out[27]:
```
VC(I=3, J=-4, K=-5)
```

In [28]:
```
# Create a vector by applying unary plus to a vector
v = SVC(-3, 4, 5)
+v
```
Out[28]:
```
VC(I=-3, J=4, K=5)
```
In [29]:
```
# Create a vector by adding a vector to another
u = SVC(-3, 4, 5)
v = SVC(1, 1, -1)
u + v
```
Out[29]:
```
VC(I=-2, J=5, K=4)
```
In [30]:
```
# In-place addition of a vector to another
u = SVC(-3, 4, 5)
v = SVC(1, 1, -1)
u += v
u
```
Out[30]:
```
VC(I=-2, J=5, K=4)
```
In [31]:
```
# Create a vector by subtracting a vector from another
u = SVC(-3, 4, 5)
v = SVC(1, 1, -1)
u - v
```
Out[31]:
```
VC(I=-4, J=3, K=6)
```
In [32]:
```
# In-place subtraction of a vector from another
u = SVC(-3, 4, 5)
v = SVC(1, 1, -1)
u -= v
u
```
Out[32]:
```
VC(I=-4, J=3, K=6)
```
In [33]:
```
# Create a vector by multiplying a vector by another
u = SVC(-1, 2, 3)
v = SVC(2, 0, -2)
u * v
```
Out[33]:
```
VC(I=-2, J=0, K=-6)
```

In [34]:
```
# In-place multiplication of a vector by another
u = SVC(-1, 2, 3)
v = SVC(2, 0, -2)
u *= v
u
```
Out[34]:
```
VC(I=-2, J=0, K=-6)
```

In [35]:
```
# Create a vector by multiplying a vector and a scalar
v = SVC(-1, 2, 3)
s = 2
s * v, v * s
```
Out[35]:
```
(VC(I=-2, J=4, K=6), VC(I=-2, J=4, K=6))
```

In [36]:
```
# In-place multiplication of a vector by a scalar
v = SVC(-1, 2, 3)
s = 2
v *= s
v
```
Out[36]:
```
VC(I=-2, J=4, K=6)
```

In [37]:
```
# Create a vector by dividing a vector by another
u = SVC(-3, 4, 6)
v = SVC(2, -2, 2)
u / v
```
Out[37]:
```
VC(I=-1.5, J=-2.0, K=3.0)
```

In [38]:
```
# In-place true division of a vector by another
u = SVC(-3, 4, 6)
v = SVC(2, -2, 2)
u /= v
u
```
Out[38]:
```
VC(I=-1.5, J=-2.0, K=3.0)
```

```
In [39]: # Create a vector by true dividing a vector by a scalar
         v = SVC(-3, 4, 6)
         s = 6
         v / s

Out[39]: VC(I=-0.5, J=0.6666666666666666, K=1.0)
```

```
In [40]: # In-place true division of a vector by a scalar
         v = SVC(-3, 4, 6)
         s = 2
         v /= s
         v

Out[40]: VC(I=-1.5, J=2.0, K=3.0)
```

```
In [41]: # Create a vector by raising a vector to the power of another
         u = SVC(-3, 4, 6)
         v = SVC(2, -2, 2)
         u**v

Out[41]: VC(I=9, J=0.0625, K=36)
```

```
In [42]: # In-place raising a vector to the power of vector
         u = SVC(-3, 4, 6)
         v = SVC(2, -2, 2)
         u **= v
         u

Out[42]: VC(I=9, J=0.0625, K=36)
```

```
In [43]: # Create a vector by raising a vector to the power of a scalar
         v = SVC(-3, 5, 6)
         s = 2
         v**s

Out[43]: VC(I=9, J=25, K=36)
```

In [44]:
```python
# In-place raising a vector to the power of a scalar
v = SVC(-3, 5, 6)
s = 2
v **= s
v
```

Out[44]:
```
VC(I=9, J=25, K=36)
```

In [45]:
```python
# Create a vector by floor dividing a vector by another
u = SVC(-3, 5, 6)
v = SVC(2, -2, 2)
u // v
```

Out[45]:
```
VC(I=-2, J=-3, K=3)
```

In [46]:
```python
# In-place floor division of a vector by another
u = SVC(-3, 5, 6)
v = SVC(2, -2, 2)
u //= v
u
```

Out[46]:
```
VC(I=-2, J=-3, K=3)
```

In [47]:
```python
# Create a vector by floor dividing a vector by a scalar
v = SVC(-3, 5, 6)
s = 2
v // s
```

Out[47]:
```
VC(I=-2, J=2, K=3)
```

In [48]:
```python
# In-place floor division of a vector and a scalar
v = SVC(-3, 5, 6)
s = 2
v //= s
v
```

Out[48]:
```
VC(I=-2, J=2, K=3)
```

In [49]:
```python
# Create a vector by applying modulus to a vector and another
u = SVC(-3, 5, 6)
v = SVC(2, -2, 2)
u % v
```

Out[49]:
```
VC(I=1, J=-1, K=0)
```

In [50]:
```python
# In-place application of modulus to a vector and another
u = SVC(-3, 5, 6)
v = SVC(2, -2, 2)
u %= v
u
```

Out[50]:
```
VC(I=1, J=-1, K=0)
```

In [51]:
```python
# Create a vector by applying modulus to a vector and a scalar
v = SVC(-3, 5, 6)
s = 2
v % s
```

Out[51]:
```
VC(I=1, J=1, K=0)
```

In [52]:
```python
# In-place application of modulus to a vector and a scalar
v = SVC(-3, 5, 6)
s = 2
v %= s
v
```

Out[52]:
```
VC(I=1, J=1, K=0)
```

In [ ]:
```python

```