Coding Back-Substitution

In [1]:

```
#keep
import numpy as np
```

Here's an upper-triangular matrix A and two vectors x and b so that $Ax = b$.

See if you can find x by computation.

In [11]:

```
#keep
n = 5

A = np.random.randn(n, n) * np.tri(n).T
print(A)

x = np.random.randn(n)
print(x)

b = np.dot(A, x)
```

```
[[-1.26236737 -0.8644523   1.55110419 -0.94165954 -0.71166821]
 [-0.         -1.89991829 -1.12215066  0.16162471 -0.5094088 ]
 [-0.         -0.         -0.52611369  1.03649351 -1.03046035]
 [-0.         -0.          0.          0.22869562 -0.45786146]
 [-0.         -0.          0.         -0.          0.19889282]]
[ 1.35615426 -0.7539793  -0.04295377  0.12033124 -1.9996183 ]
```

In [16]:

```
xcomp = np.zeros(n)

for i in range(n-1, -1, -1):
    tmp = b[i]
    for j in range(n-1, i, -1):
        tmp -= xcomp[j]*A[i,j]

    xcomp[i] = tmp/A[i,i]
```

Now compare the computed x against the reference solution.

In [19]:

```
#keep
print(x)
print(xcomp)
```

```
[ 1.35615426 -0.7539793  -0.04295377  0.12033124 -1.9996183 ]
[ 1.35615426 -0.7539793  -0.04295377  0.12033124 -1.9996183 ]
```

Questions/comments:

- Can this fail?
- What's the operation count?