Problem sheet 1.

Please submit the question marked * to your tutor.

E1.1* (Catastrophic cancellation)

The following bit of code defines the polynomial $p(x) = ax^2 + bx + c$ and evaluates it at $x = 2$

In [1]:
```
a,b,c=1,3,2
p = lambda x: a*x**2+b*x+c
x = 2
print('The polynomial evaluated at x=2 is',p(x))
```

```
The polynomial evaluated at x=2 is 12
```

Complete the following to write a function to return the roots of a quadratic polynomial given coefficients a,b,c.

In [2]:
```
import numpy as np
def quad_roots(a,b,c):
    #fill in
    # xp = ...
    # xm = ...
    return xp, xm
```

Try your code with $a = 10^{-3}$, $b = 10^6$ and $c = 10^{-3}$.

Does your function suffer from cancellation errors? If so, modify your function to avoid this.

By using the Taylor expansion for $\sqrt{1-x}$, work out what the actual roots are to 16 significant figures.

Note: Evaluate your polynomial at the computed roots, the value should be close to 0 (although not exactly 0 due to machine precision).

Hint: Which root would suffer from cancellation errors? If you know one root to high accuracy, can you work out the other?

In [3]:
```
a,b,c = 1e-3,1e6,1e-3
# test out your function!
```

E1.2 (Conditioning and the Hilbert matrix)

Run the following command. It defines the Hilbert matrix which has (i, j) entry as $1/(i + j - 1)$ for $i, j = 1, \dots, d$.

In [4]:
```
from scipy.linalg import hilbert
d=3
A = hilbert(d)
print(A)
```

```
[[1.         0.5        0.33333333]
 [0.5        0.33333333 0.25      ]
 [0.33333333 0.25       0.2       ]]
```

Let $b \in \mathbb{R}^d$ have random entries defined as follows.

In [5]:
```
b = np.random.randn(d,)
```

We can solve the linear system $Ax = b$ and compute the residue $r = Ax - b$ as follow:

In [6]:
```
x = np.linalg.solve(A,b)
r = A@x - b
print('Should be small:', np.linalg.norm(r))
```

```
Should be small: 1.8251269632034268e-15
```

In the above, the norm computes $\|r\| := \sqrt{r^\top r}$

Repeat the above for $d = 10$ and $d = 20$, what do you observe?