Computer Exercise 2.1.8

The following program will use the *naive_gauss* algorithm to solve $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{b}$ is defined such that $\mathbf{x} = (1, 2, \ldots, n)^T$ where n is the dimension of the square matrix $\mathbf{A}$. We will first define a preliminary array $\mathbf{x}_0 = (1, 2, \ldots, n)^T$ then use matrix multiplication $\mathbf{Ax}_0$ to acquire $\mathbf{b}$. *naive_gauss* will then be applied to $\mathbf{A}$ and $\mathbf{b}$ to determine if we end up having $\mathbf{x} = \mathbf{x}_0$.

```
%seed for random number generator
format default
rng('default')
s = rng;

n = 5;
A = randi([4, 20]).*rand(n,n);
x0 = (1:n)';
b = A*x0;
x = naive_gauss(A, b);
x
```

```
x = 5×1
    1.0000
    2.0000
    3.0000
    4.0000
    5.0000
```

We see that indeed $\mathbf{x} = \mathbf{x}_0$.

```
function x = naive_gauss(A, b)
    n = length(b);
%forward elimination
    for k = 1:(n-1)
        for i = (k+1):n
            xmult = (A(i, k))/(A(k, k));
            A(i, k) = 0;
            for j = (k+1):n
                A(i, j) = A(i, j) - xmult*A(k, j);
                if abs(A(i, j)) < 10^(-12)
                    A(i, j) = 0;
                end
            end
            b(i) = b(i) - xmult*b(k);
        end
    end
%backwards substitution
    x = zeros(n, 1);
    x(n) = (b(n))/(A(n,n));
    for u = (n-1):-1:1
        sum = 0;
        for v = (u+1):n
```

```
            sum = sum + (A(u, v)*x(v));
        end
        x(u) = (b(u) - sum)/(A(u, u));
    end
end
```