EX.2.1.8, Sauer3

If a system of 3000 equations in 3000 unknowns can be solved by Gaussian elimination in 5 seconds on a given computer, how many back substitutions of the same size can be done per second?

EX.2.1.8, Sauer3, solution, Langou

Colab link: `https://colab.research.google.com/drive/1pE9tW7RnnzFXp132H_jWZfSbNK-ZvdBH`

Gaussian elimination take $\frac{2}{3}n$ more time than backsubstitution, so with $n = 3000$, 1 Gaussian elimination take 2000 more time than 1 backsubstitution, So if we can do 1 Gaussian elimination in 5 seconds, then we can 2000 backsubstitutions in 5 seconds. So we can do 400 backsubstitutions in 1 second.

$$\boxed{\text{400 backsubstitutions per second}}$$

```
n = 3000.
gigaflops_per_second = ( 2./3. * n * n * n ) / 5 * 1e-9
print( gigaflops_per_second, 'GigaFLOP per second' )
```

```
3.6 GigaFLOP per second
```

```
time_backsubstitution = \
 ( n * n ) / ( gigaflops_per_second * 1e9 )
print( f"{time_backsubstitution:10.5f}", 'seconds' )
```

```
0.00250 seconds
```

```
print( f"{1./time_backsubstitution:10.1f}", 'backsubstitution second' )
```

```
400.0 backsubstitution second
```

Backsubstitution requires n^2 (floating-point) operations (FLOPs).
If our computer can complete backsubstitution in 0.002 seconds for $n = 4000$, then it performs

```
n = 4000.
gigaflops_per_second = ( n * n ) / 0.002 * 1e-9
print( gigaflops_per_second, 'GigaFLOP per seconds' )
```

```
8.0 GigaFLOP per seconds
```

which means 8 billions operations per seconds. (We say 8 GigaFLOPs/sec.)

Gaussian elimination requires $\frac{2}{3}n^3$ operations. The time, in second, needed for $n = 9000$ is therefore

```
n = 9000.
time_GaussianElimination = \
 ( 2./3. * ( n * n * n ) ) / ( gigaflops_per_second * 1e9 )
print( time_GaussianElimination, 'seconds' )
```

```
60.75 seconds
```

Rouding to the nearest second, we get $\boxed{\text{61 seconds}}$. So about 1 minute.

Not needed for full credit.

We can check that the time for backsubstiution is negligible

```
n = 9000.
time_backsubstitution = ( n * n ) / ( gigaflops_per_second * 1e9 )
print( time_backsubstitution, 'seconds' )
```

```
0.010125 seconds
```

Indeed, the time for backsubstiution is negligible compared to the time for Gaussian elimination. And this makes sense, since backsubstiution is order n^2, and Gaussian elimination is order n^3, and n is 9000. More precisely, the time for backsubstiution is $\frac{2}{3}n$ times smaller than the time for Gaussian elimination, so, with $n = 9000$, it is 6000 times smaller.

Not needed for full credit.

```
n = 9000.
print( f"{2./3. * ( n * n * n ) * 1e-9:4.0f}",\
      'billions of floating-point operations' )
```

```
486 billions of floating-point operations
```

We see that to solve a 9,000 $\times$ 9,000 system of linear of equations, we need to perform (about) 486 billions floating-point operations (FLOPS).