In [59]:
```
from sympy.abc import x
from sympy import sqrt,diff,plot,solve,evalf
```

In [60]:
```
eq = x**2-x-1
eq
```

Out[60]: $x^2 - x - 1$

In [61]:
```
solve(eq)
```

Out[61]:
```
[1/2 - sqrt(5)/2, 1/2 + sqrt(5)/2]
```

In [62]:
```
r1 = 1/2 - sqrt(5)/2
r1
```

Out[62]: $0.5 - \frac{\sqrt{5}}{2}$

In [63]:
```
r2 = 1/2 + sqrt(5)/2
r2
```

Out[63]: $0.5 + \frac{\sqrt{5}}{2}$

In [6]:
```
r1.evalf(10)
```

Out[6]: -0.6180339888

In [28]:
```
r2.evalf(60)
```

Out[28]: $1.61803398874989484820458683436563811772030917980576286213545$

In [1]:
```
for i in range(2,1_000_000,100_000):
    import time
    from sympy import sqrt
    r2 = 1/2 + sqrt(5)/2
    startTime = time.time()

    digits = i
    result = r2.evalf(digits)

    executionTime = (time.time() - startTime)
    print('Execution time to calculate',digits,'Digits of Phi (in seconds): ' + str(executionTime))
    print("Last digit is:", str(result)[-1])
    print("                              ")
```

```
Execution time to calculate 2 Digits of Phi (in seconds): 0.0001571178436279297
Last digit is: 6

Execution time to calculate 100002 Digits of Phi (in seconds): 0.06370139122009277
Last digit is: 8

Execution time to calculate 200002 Digits of Phi (in seconds): 0.18637681007385254
Last digit is: 0

Execution time to calculate 300002 Digits of Phi (in seconds): 0.3826889991760254
Last digit is: 8

Execution time to calculate 400002 Digits of Phi (in seconds): 0.4409317970275879
Last digit is: 2

Execution time to calculate 500002 Digits of Phi (in seconds): 0.77689528465271
Last digit is: 3

Execution time to calculate 600002 Digits of Phi (in seconds): 1.0968341827392578
Last digit is: 6

Execution time to calculate 700002 Digits of Phi (in seconds): 1.3132565021514893
Last digit is: 5

Execution time to calculate 800002 Digits of Phi (in seconds): 1.5213916301727295
Last digit is: 3

Execution time to calculate 900002 Digits of Phi (in seconds): 1.693979263305664
Last digit is: 2
```

In [2]:
```
# Execution Times
t0  = 0.0001571178436279297
t1  = 0.06370139122009277
t2 = 0.18637681007385254
t3 = 0.3826889991760254
t4 = 0.4409317970275879
t5 = 0.77689528465271
t6 = 1.0968341827392578
t7 = 1.3132565021514893
t8 = 1.5213916301727295
t9 = 1.693979263305664
```

In [42]:
```
# Digits calculated per second
d1 = (10_002-2)/(t1-t0)
d2 = (20_002-10_002)/(t2-t1)
d3 = (30_002-20_002)/(t3-t2)
d4 = (40_002-30_002)/(t4-t3)
d5 = (50_002-40_002)/(t5-t4)
d6 = (60_002-50_002)/(t6-t5)
d7 = (70_002-60_002)/(t7-t6)
d8 = (80_002-70_002)/(t8-t7)
d9 = (90_002-80_002)/(t9-t8)
```

In [58]:
```
# Average Digits calculated per second
(d1+d2+d3+d4+d5+d6+d7+d8+d9)/9
```

Out[58]:
```
74970.5764998911
```

In [64]:
```
# 75,000 digits of Phi calculated per second.
```

In [47]:
```
# A Record divided by my rate
(10**13)/75_000  # 10^13 is the amount of accurate digits Phi has been calculated to
                 #  - Yee, Alexander J. (13 March 2021). "Records Set by y-cruncher". numberword.org.
                 #    Two independent computations done by Clifford Spielman.
```

Out[47]:
```
133333333.33333333
```

In [65]:
```
# 133_333_333 seconds
```

In [55]:
```
133_333_333 / 60
```

Out[55]:
```
2222222.216666667
```

In [52]:
```
# 2_222_222 minutes
```

In [56]:
```
2_222_222 / 60
```

Out[56]:
```
37037.03333333333
```

In [54]:
```
#37_037 hours
```

In [57]:
```
37_037 / 8_766   # (8_766 hours in a year)
```

Out[57]:
```
4.225074150125485
```

In [ ]:
```
# 4.225 years to compute 10^13 digits of Phi with my computer.
```