In [1]:
```
import mpmath
import numpy as np
```

In [2]:
```
# Initialize the pi for each kind
mpmath.mp.dps = 60
piMathe = mpmath.pi
piEgypt = mpmath.mpf(22/7)
piChina = mpmath.mpf(355/113)
piIndia = mpmath.mpf(339/108)
piGreec = mpmath.mpf(0.5 * ((223/71) + (22/7)))

print("piMathe = {}".format(piMathe))
print("piEgypy = {}".format(piEgypt))
print("piChina = {}".format(piChina))
print("piIndia = {}".format(piIndia))
print("piGreec = {}".format(piGreec))
print('{:.60f}'.format(22/7))
```

```
piMathe = 3.14159265358979323846264338327950288419716939937510582097494
piEgypy = 3.14285714285714279370154144999105483293533325195312
piChina = 3.14159292035398252096456417348235845565795898437
piIndia = 3.13888888888888888395456433499930426478385925292986875
piGreec = 3.14185110663983913070751441409813775253295898434
3.142857142857142793701544499910548329353325195312500000000
```

In [3]:
```
# Extract the data after the decimal point
piMathe = str(piMathe)
piEgypt = str(piEgypt)
piChina = str(piChina) + '00'
piIndia = str(piIndia) + '0'
piGreec = str(piGreec) + '0'
print("piMathe {} {}".format(piMathe, len(piMathe)))
print("piEgypt {} {}".format(piEgypt, len(piEgypt)))
print("piChina {} {}".format(piChina, len(piChina)))
print("piIndia {} {}".format(piIndia, len(piIndia)))
print("piGreec {} {}".format(piGreec, len(piGreec)))
print()

piMathe = piMathe[2:52]
piEgypt = piEgypt[2:52]
piChina = piChina[2:52]
piIndia = piIndia[2:52]
piGreec = piGreec[2:52]
print("piMathe {} {}".format(piMathe, len(piMathe)))
print("piEgypt {} {}".format(piEgypt, len(piEgypt)))
print("piChina {} {}".format(piChina, len(piChina)))
print("piIndia {} {}".format(piIndia, len(piIndia)))
print("piGreec {} {}".format(piGreec, len(piGreec)))
```

```
piMathe 3.14159265358979323846264338327950288419716939937510582097494 61
piEgypt 3.14285714285714279370154144999105483293533325195312 53
piChina 3.14159292035398252096456417348235845565795898437500 52
piIndia 3.13888888888888888395456433499930426478385925292986 53
piGreec 3.14185110663983913070751441409813775253295898434750 52

piMathe 14159265358979323846264338327950288419716939937510 50
piEgypt 14285714285714279370154144999105483293533325195312 50
piChina 14159292035398252096456417348235845565795898437500 50
piIndia 13888888888888888395456433499930426478385925292687 50
piGreec 14185110663983913070751441409813775253295898434750 50
```

In [4]:
```
# Error between different error
def error(true, test):
    true = int(true)
    test = int(test)
    temp = abs(true-test) / true
    return temp*100

t = error(piMathe, piEgypt)
print("Egype method erroe is {}%".format(t))

t = error(piMathe, piChina)
print("Chian method erroe is {}%".format(t))

t = error(piMathe, piIndia)
print("India method erroe is {}%".format(t))

t = error(piMathe, piGreec)
print("Greece method erroe is {}%".format(t))
```

```
Egype method erroe is 0.8930472275863234%
Chian method erroe is 0.00018840256363535378%
India method erroe is 1.9095374176243989%
Greece method erroe is 0.18253281049075765%
```

Question 1

In [5]:
```
# How many first decimal digits are correct when compaing with piMathe
def sameLetter(test, answer):
    n = 0
    for (t, a) in zip(test, answer):
        if t == a:
            n = n+1
        else:
            return n

if __name__ == "__main__":
    n = sameLetter(piEgypt, piMathe)
    print('For piEgypt, n = {}'.format(n))
    n = sameLetter(piChina, piMathe)
    print('For piChina, n = {}'.format(n))
    n = sameLetter(piIndia, piMathe)
    print('For piIndia, n = {}'.format(n))
    n = sameLetter(piGreec, piMathe)
    print('For piGreec, n = {}'.format(n))

    print('China method gave the highest precison')
```

```
For piEgypt, n = 2
For piChina, n = 6
For piIndia, n = 1
For piGreec, n = 3
China method gave the highest precison
```

In [6]:
```
# Compute the frequency
def digitFrequency(inputVector):
    n = len(inputVector)
    ans = [ 0 for i in range(10)]
    for d in inputVector:
        d = int(d)
        ans[d] = ans[d] + 1

    ans = np.array(ans, dtype = 'f')
    ans = (ans * 100) / len(inputVector)

    return ans

if __name__ == "__main__":
    f = digitFrequency(piMathe)
    print("Frequency of piMathe = {}, sum = {}, max = {}, min = {}".format(f, sum(f), max(f), min(f)))
    f = digitFrequency(piEgypt)
    print("Frequency of piEgype is {}, sum = {}, max = {}, min = {}".format(f, sum(f), max(f), min(f)))
    f = digitFrequency(piChina)
    print("Frequency of piChina is {}, sum = {}, max = {}, min = {}".format(f, sum(f), max(f), min(f)))
    f = digitFrequency(piIndia)
    print("Frequency of piIndia is {}, sum = {}, max = {}, min = {}".format(f, sum(f), max(f), min(f)))
    f = digitFrequency(piGreec)
    print("Frequency of piGreec is {}, sum = {}, max = {}, min = {}".format(f, sum(f), max(f), min(f)))
```

```
Frequency of piMathe = [ 4. 10. 10. 16.  8. 10.  8.  8. 10. 16.], sum = 100.0, max = 16.0, min = 4.0
Frequency of piEgype is [ 4. 16. 12. 14. 14. 14.  0.  8.  6. 12.], sum = 100.0, max = 16.0, min = 0.0
Frequency of piChina is [ 8.  6. 10. 10. 12. 20.  6.  6. 10. 12.], sum = 100.0, max = 20.0, min = 6.0
Frequency of piIndia is [ 2.  2.  8. 12. 10.  8.  6.  4. 34. 14.], sum = 100.0, max = 34.0, min = 2.0
Frequency of piGreec is [10. 18.  4. 12. 10. 12.  4. 10. 10. 10.], sum = 100.0, max = 18.0, min = 4.0
```

Quesiton 2

In [7]:
```
piMathe = digitFrequency(piMathe)
piEgypt = digitFrequency(piEgypt)
piChina = digitFrequency(piChina)
piIndia = digitFrequency(piIndia)
piGreec = digitFrequency(piGreec)

print(piMathe)
print(piEgypt)
print(piChina)
print(piIndia)
print(piGreec)
```

```
[ 4. 10. 10. 16.  8. 10.  8.  8. 10. 16.]
[ 4. 16. 12. 14. 14. 14.  0.  8.  6. 12.]
[ 8.  6. 10. 10. 12. 20.  6.  6. 10. 12.]
[ 2.  2.  8. 12. 10.  8.  6.  4. 34. 14.]
[10. 18.  4. 12. 10. 12.  4. 10. 10. 10.]
```

In [8]:
```
import statistics
def maxAbs(test, ans):
    errorList = []
    for (t, a) in zip(test, ans):
        t = int(t)
        a = int(a)
        error = abs(t - a)
        errorList.append(error)
    return max(errorList)

def medianAbs(test, ans):
    errorList = []
    for (t, a) in zip(test, ans):
        t = int(t)
        a = int(a)
        error = abs(t - a)
        errorList.append(error)
    return statistics.median(errorList)
def meanAbs(test, ans):
    errorList = []
    for (t, a) in zip(test, ans):
        t = int(t)
        a = int(a)
        error = abs(t - a)
        errorList.append(error)
    return sum(errorList) / len(errorList)

def rootSquError(test, ans):
    errorList = []
    for (t, a) in zip(test, ans):
        t = int(t)
        a = int(a)
        error = abs(t - a)
        errorList.append(error * error)
    return(sum(errorList) / len(errorList))**0.5

if __name__ == "__main__":

    # Max Absolute
    e = maxAbs(piEgypt, piMathe)
    print("piEgypt, max absolute is {}".format(e))
    e = maxAbs(piChina, piMathe)
    print("piChina, max absolute is {}".format(e))
    e = maxAbs(piIndia, piMathe)
    print("piIndia, max absolute is {}".format(e))
    e = maxAbs(piGreec, piMathe)
    print("piGreec, max absolute is {}".format(e))
    print()

    # Median Absolute
    e = medianAbs(piEgypt, piMathe)
    print("piEgypt, median absolute is {}".format(e))
    e = medianAbs(piChina, piMathe)
    print("piChina, median absolute is {}".format(e))
    e = medianAbs(piIndia, piMathe)
    print("piIndia, median absolute is {}".format(e))
    e = medianAbs(piGreec, piMathe)
    print("piGreec, median absolute is {}".format(e))
    print()

    # Mean Absolute
    e = meanAbs(piEgypt, piMathe)
    print("piEgypt, mean absolute is {}".format(e))
    e = meanAbs(piChina, piMathe)
    print("piChina, mean absolute is {}".format(e))
    e = meanAbs(piIndia, piMathe)
    print("piIndia, mean absolute is {}".format(e))
    e = meanAbs(piGreec, piMathe)
    print("piGreec, mean absolute is {}".format(e))
    print()

    # RMSE
    e = rootSquError(piEgypt, piMathe)
    print("piEgypt, RMSE is {:.1f}".format(e))
    e = rootSquError(piChina, piMathe)
    print("piChina, RMSE is {:.1f}".format(e))
    e = rootSquError(piIndia, piMathe)
    print("piIndia, RMSE is {:.1f}".format(e))
    e = rootSquError(piGreec, piMathe)
    print("piGreec, RMSE is {:.1f}".format(e))
    print()
```

```
piEgypt, max absolute is 8
piChina, max absolute is 10
piIndia, max absolute is 24
piGreec, max absolute is 8

piEgypt, median absolute is 4.0
piChina, median absolute is 4.0
piIndia, median absolute is 2.0
piGreec, median absolute is 4.0

piEgypt, mean absolute is 3.6
piChina, mean absolute is 3.6
piIndia, mean absolute is 5.2
piGreec, mean absolute is 4.0

piEgypt, RMSE is 4.4
piChina, RMSE is 4.6
piIndia, RMSE is 8.3
piGreec, RMSE is 4.6
```