Implementation of RSA Algorithm

In [3]:

```
import random
import numpy as np
import math
from mpmath import mp
```

In [4]:

```
def mod(num,a):
    res = 0
    for i in range(0, len(num)):
        res = (res * 10 + int(num[i])) % a;
    return res
```

In [5]:

```
def gcd(a,h):
    temp=""
    while (1):
        temp = a%h;
        if (temp == 0):
            return h
        a = h
        h = temp
```

In [6]:

```
def selecte(n,P,Q):
    max=(P-1)*(Q-1)
    e=2

    while e < max:
        if (gcd(e, max)==1):
            break
        else:
            e=e+1
    return e
```

In [8]:

```
#Encrypt with public key
def encrypt(data,publickey):
    n=publickey[0]
    e=publickey[1]
    return pow(data,e)%n
```

In [9]:

```
def decrypt(cipher,privatekey,n):
    orig= pow(cipher,privatekey)
    return orig%n
```

In [10]:

```
P=3
Q=7
firstpart=P*Q

e=selecte(firstpart,P,Q)
```

In [11]:

```
k=2
```

```
phi=(P-1)*(Q-1)
privatekey = (1 + (k*phi))/e;
print("Private Key",privatekey)
```

```
Private Key 5.0
```

In [12]:

```
publickey=(firstpart,e)
print("Public Key:",publickey)
```

```
Public Key: (21, 5)
```

In [13]:

```
data=12
cipher=encrypt(data,publickey)
originaltext=decrypt(cipher,privatekey,firstpart)
```

In [14]:

```
print("Data:",data)
print("Cipher:(Data after encryption)",cipher)
print("Original Data:(After decryption of cipher)",originaltext)
```

```
Data: 12
Cipher:(Data after encryption) 3
Original Data:(After decryption of cipher) 12.0
```