The keys for the RSA algorithm are generated in the following way:

```
SIMPLE EXAMPLE:
```

1. Choose two distinct prime numbers p and q.

- For security purposes, the integers p and q should be chosen at random, and should be similar in magnitude but differ in length by a few digits to make factoring harder. Prime integers can be efficiently found using a primality test.
- p and q are kept secret.

```
p=7 and q=11
```

2. Compute n = pq.

- n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
- n is released as part of the public key.

```
n = p*q = 7*11 = 77
```

3. Compute $\lambda(n)$, where λ is Carmichael's totient function. Since $n = pq$, $\lambda(n) = \mathrm{lcm}(\lambda(p),\lambda(q))$, and since p and q are prime, $\lambda(p) = \varphi(p) = p - 1$ and likewise $\lambda(q) = q - 1$. Hence $\lambda(n) = \mathrm{lcm}(p - 1, q - 1)$.

- $\lambda(n)$ is kept secret.
- The lcm may be calculated through the Euclidean algorithm, since $\mathrm{lcm}(a,b) = |ab|/\gcd(a,b)$.

```
λ(p) = φ(p) = p-1 = 7-1 = 6
λ(q) = φ(q) = q-1 = 11-1 = 10
λ(n) = lcm(λ(p),λ(q)) = lcm(6,10) = 30
```

4. Choose an integer e such that $1 < e < \lambda(n)$ and $\gcd(e, \lambda(n)) = 1$; that is, e and $\lambda(n)$ are coprime.

- e having a short bit-length and small Hamming weight results in more efficient encryption – the most commonly chosen value for e is $2^{16} + 1 = 65{,}537$. The smallest (and fastest) possible value for e is 3, but such a small value for e has been shown to be less secure in some settings.[14]
- e is released as part of the public key.

```
e | 1<e<λ(n) and gcd(e,λ(n))=1 => e | 1<e<30 and gcd(e,30)=1 => i.e. e=13
```

5. Determine d as $d \equiv e^{-1} \pmod{\lambda(n)}$; that is, d is the modular multiplicative inverse of e modulo $\lambda(n)$.

- This means: solve for d the equation $d \cdot e \equiv 1 \pmod{\lambda(n)}$; d can be computed efficiently by using the Extended Euclidean algorithm, since, thanks to e and $\lambda(n)$ being coprime, said equation is a form of Bézout's identity, where d is one of the coefficients.
- d is kept secret as the private key exponent.

```
d*e = 1 mod λ(n) => d*13 = 1 mod 30 => d=7  (7*13=91=30+30+30+1)
```

ENCRYPTION AND DECRYPTION:

```
m = 40 (message, converted to integer smaller than n=77)
c = m^e mod n = 40^13 mod 77 = 68
d = c^d mod n = 68^7 mod 77 = 40
```

```
#!/bin/bash
#  @FBM2020

EXPECTED_ARGS=3


if [ $# -eq $EXPECTED_ARGS ]
then

      targetCnt=$(($1 + 0))
      if [ "$targetCnt" -le "1" ];
      then
            echo "First argument \"$1\" is not an integer > 1"
            exit
      fi

      targetCnt=$(($2 + 0))
      if [ "$targetCnt" -le "0" ];
      then
            echo "Second argument \"$2\" is not an integer > 0"
            exit
      fi

      targetCnt=$(($3 + 0))
      if [ "$targetCnt" -le "1" ];
      then
            echo "Third argument \"$3\" is not an integer > 1"
            exit
      fi

      a=1;
      for (( i = 1; i <= $2; ++i ));
      do
#           echo "[+] Round: $i. Base=$1 Exponent=$2 Modulus=$3"
#           echo "[+] Modular multiplication: a=($a*$1)%$3=$(( $a * $1 ))%$3=$((
$(( $a * $1 )) % $3 ))"
            a=$(( $(( $a * $1 )) % $3 ))
#           echo -e "\n"
      done

#     echo "[+] Done!. $1 raised to $2 module $3 is $a"
      echo "$a"

else
        printf  "Usage: $0 Base Exponent Modulus\n"

fi;
```

Wikipedia example

1. Choose two distinct prime numbers, such as
   $p = 61$ and $q = 53$
2. Compute $n = pq$ giving
   $n = 61 \times 53 = 3233$
3. Compute the Carmichael's totient function of the product as $\lambda(n) = \operatorname{lcm}(p - 1, q - 1)$ giving
   $\lambda(3233) = \operatorname{lcm}(60, 52) = 780$
4. Choose any number $1 < e < 780$ that is coprime to 780. Choosing a prime number for e leaves us only to check that e is not a divisor of 780.
   Let $e = 17$
5. Compute d, the modular multiplicative inverse of e (mod $\lambda(n)$) yielding,
   $d = 413$
   Worked example for the modular multiplicative inverse:
   $d \times e = 1 \bmod \lambda(n)$
   $413 \times 17 = 1 \bmod 780$

The **public key** is (n = 3233, e = 17). For a padded plaintext message m, the encryption function is

$$c(m) = m^{17} \bmod 3233$$

The **private key** is (n = 3233, d = 413). For an encrypted ciphertext c, the decryption function is

$$m(c) = c^{413} \bmod 3233$$

For instance, in order to encrypt m = 65, we calculate

$$c = 65^{17} \bmod 3233 = 2790$$

To decrypt c = 2790, we calculate

$$m = 2790^{413} \bmod 3233 = 65$$