PROBLEM - A

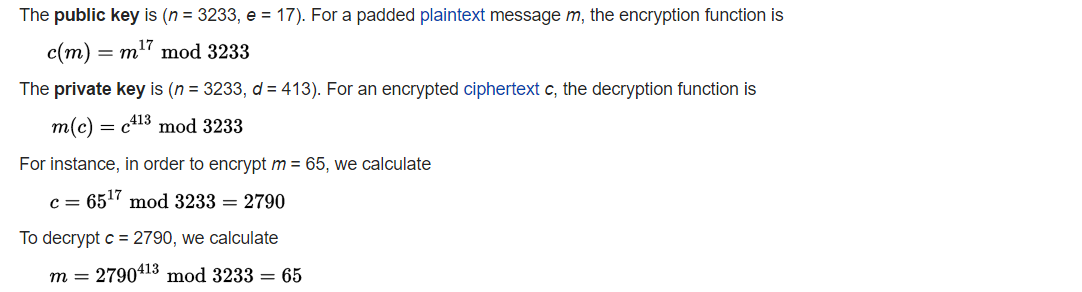
# Prime importance

The RSA cryptosystem, used ubiquitously in almost all forms of modern cryptography, functions on the basis of large prime numbers.  Here is what has to happen in order to generate secure RSA keys:

1. **Large Prime Number Generation**: Two large prime numbers ***p*** and ***q*** need to be generated. These numbers are very large: At least 512 digits, but 1024 digits is considered safe.
2. **Modulus**: From the two large numbers, a modulus ***n*** is generated by multiplying ***p*** and ***q***.
3. **Totient**: The totient of ***n, ϕ(n)*** is calculated.
4. **Public Key**: A *prime number* **e** is calculated from the range ***[3,ϕ(n))*** that has a greatest common divisor of 1 with ***ϕ(n)***. The public key is the pair of numbers **(n, e)**
5. **Private Key**: Because the prime in step 4 has a gcd of 1 with ***ϕ(n)***, we are able to determine it's inverse **d** with respect to ***mod ϕ(n)***. The private key is the pair of numbers **(n, d)**

***ϕ(n)*** represents [**Euler's Totient function**](https://en.wikipedia.org/wiki/Euler%27s_totient_function)**.** For given definition, ***ϕ(n) = lcm(p - 1, q - 1)***

**Modular inverse is calculated using the Euclidean algorithm.**



A naïve cryptographer was careless enough to generate small prime numbers ***p, q.*** For given public key and ciphertext, find out the plaintext.

Public key = (551,265)

Ciphertext = 338

PROBLEM - B

# The bomb

To intercept Nazi army messages to the frontlines, the Polish began development on a device, called the **bombe.** It was intended to break the encryption provided by German devices called the Enigma.

The bombe was designed to discover some of the daily settings of the Enigma machines on the various German military networks: specifically, the set of rotors in use and their positions in the machine; the rotor core start positions for the message—the message key—and one of the wirings of the plugboard.

The cryptanalysis of the Enigma machine was made possible by weaknesses in the design of the Enigma and the carelessness of the machine operators in properly following the procedures of use.

One of the key problems with the Enigma was that a letter could never be encrypted to itself, thus ruling out many possible plaintexts once a partial decryption of a message was available.

Given an encrypted Enigma message and a partial decryption, determine the list of possible offsets from the beginning of the message and the decryption.

Partial ciphertext : …ANQHFTQHLR….

Partial plaintext : HIELHITLER

The answer to this problem lies in the polyalphabetic shift of the letters “AKQ” using values from the above list, sorted in non-decreasing order.

PROBLEM - C

# The decision tree

A lot of ancient cryptography was based on pre-shared sequences of data, which both the parties could exchange secretly and then use a hidden and unique scheme. Given a series of numbers A, having 6 digits each, determine the number of numbers which could be valid polyalphabetic shifts for a given ciphertext, provided that the answers are to be computed from a dictionary D.

The required input files are named ‘seq.txt’ and ‘dictionary.txt’. First line of each file has the number of entries in the file.

PROBLEM - D

# Pseudoshift

Find the sum of all possible shifts of the ciphertext ‘GRJMBA’, provided that the plaintext exists in the dictionary D.

The required input file is named `dictionary.txt`.