TP02 - Exercício 2

No exercicio 2 é nos pedido para implementar o NTT-CRT tal como é descrito nos apontamentos fornecidos pelo docente.

O 1º passo para isto é descobrir um N ,que seja potencia de 2, e um primo que verifique $q \equiv 1 \mod 2N$. É de importantancia relembrar que o N escolhido tem de ser sufientemente grande para que $\mathcal{R}_{q,N}$ contenha todos os polinómios que, previsivelmente, são relevantes à aplicação desta tecnica.

In [1]:
```
n = 2048 #2^32
q = next_prime(2 * n)
i = 1
while q % (2*n) != 1:
    q = next_prime(n + i)
    i += 1
```

NTT:

A NTT tranforma um polinomio do dominio do tempo (coeficientes do polinomio) para o dominio da frequencia (valores do polinomio em pontos especifico), onde as etapas basicas da NTT são:

1. Escolha de um numero primo q, feito anteriormente. Isso normalemnte garante a existencia de uma raiz primitiva n-esima de unidade.
2. Um elemento ξ no campo **GF(q)** tal que suas potencias geram um ciclo de ordem n. esse elemento é usado para avaliar o polinomio em pontos especificos, facilitando a transformação.
3. A NTT transforma o vetor de coeficientes de um polinomio em um vetor de valores avaliados em potencias da raiz de unidade. A transformação inversa reconstroi o vetor de coeficientes original a partir do vetor transformado.

CRT:

É empregado na reconstrução do polinomio após a palicação da NTT, onde primeiramente é calculado o conjunto base para a reconstrução do polinomio original. Cada elemento da base corresponde a um polinomio que é 0 em todos os pontos exceto um, onde o valor é 1.

O vetor **ff** representa os valores do polinomio transformado pela NTT. Cada elemento de ff corresponde a um valor do polinomio em um ponto especifico. O **fff** reconstrói o polinomio original no dominio do tempo a partir de seus valores no dominio da frequencia. Isto é feito multiplicando cada valor transformado pelo seu correspondente polinomio na base do CRT e somando todos os produtos.

Por fim se verifica a igualdade entre **f** e **fff** visto que foi efetuada a inversa da NTT (facilitada pelo uso do CRT) que reconstrói o vetor de entrada original.

In [2]:
```
F = GF(q)
R = PolynomialRing(F, name="w")
w = R.gen()
g = (w^n + 1)
xi = g.roots(multiplicities=False)[-1]
rs = [xi^(2*i+1)  for i in range(n)]
base = crt_basis([(w - r) for r in rs])

f = R.random_element(1023)

u = f.list()
preenchido = u + [0]*(n-len(u))

def ntt(xi,N,f):
    if N==1:
        return f
    N_ = N/2 ; xi2 =  xi^2
    f_plus = [f[2*i]   for i in range(N_)] ; f_minus = [f[2*i+1] fo
    ff_plus = ntt(xi2,N_,f_plus) ; ff_minus = ntt(xi2,N_,f_minus)
    s  = xi ; ff = [F for i in range(N)]
    for i in range(N_):
        a = ff_plus[i] ; b = s*ff_minus[i]
        ff[i] = a + b ; ff[i + N_] = a - b
        s = s * xi2
    return ff

ff = ntt(xi,n,preenchido)
fff = sum([ff[i]*base[i] for i in range(n)])
print(f"f=fff:{f == fff}")
```

```
f=fff:True
```