⑦ Factors: Donada $N \in \mathbb{Z}^+$ $\quad x_i \in Primes$

retornar $x_1^{a_1}, \dots, x_n^{a_n}$ tq $x_1^{a_1} * \dots * x_n^{a_n} = N$

Factorization: Donada $N \in \mathbb{Z}^+$

retornar p tq $1 < p < N \wedge$ primer amb N $\wedge$ $N \bmod p = 0$

Factorization-D: Donades $N \in \mathbb{Z}^+$ i $D < N$

retornar $\{\exists p$ tq $1 < p < D \wedge N \bmod p = 0$?

a) Factorization-D is NP and co-NP

En ambdos casos podem donar un algorisme verificador que donada una possible solució (testimoni) retorni si és correcte o no:

NP:

```
input N, D, p {
    si N % p = 0 and p < D
        return TRUE
    else
        return FALSE
}
```

La correctesa és trivial

co-NP: El problema complementari ha de pertànyer a NP:

Fact-D: Donats N i D

retornar si $\forall p \leq D, N \% p \neq 0$

ho podem comprovar amb el següent verificador:

Aquesta quina relació té amb |N| ?

```
input x_1, ..., x_n, a_1, ..., a_n, D, N {
    for i = 1 to n {
        if N % x_i = 0 and x_i < D
            return false
        if N % x_i ≠ 0
            return false
    }
    return TRUE
}
```

El verificador retornarà fals si troba algun divisor de N més petit que D.

$x_1, \dots, x_n$ representen tots els divisors primers de N?

b) Si P=NP, demostrar que factorization $\in FP$

Com hem demostrat en l'apartat a) que Fact-D és NP, i P=NP, existeix un algoritme que decideix Fact-D en temps polinòmic, per tant, es pot fer una cerca dicotòmica per trobar un divisor de N:

```
input N{
    D=N/2
    while Fact-D(N,D){ ∈P=NP
        D=D/2
    }
    for i=2 to D{ ← Compte!
        if N%i=0 return i
    }
    return PRIME
}
```

nombre d'iteracions $O(D)$ " $O(2^{|N|})$!!

Aquest algoritme fa cerca dicotòmica per a trobar el primer més petit que divideix N, el cost, al fer cerca binària és: $O(\log_2(N)) \rightarrow O(\log_2(2^{|N|})) \rightarrow \rightarrow O(|N|)$.

El bucle del final fa una quantitat logarítmica de passos respecte a N, ja que es fan un nombre primer de passos. Relativament

? No ho entenc..

c) Si P=NP, Factors $\in FP$:

Similar a l'apartat anterior, si P=NP, tenim un algoritme que resol Factorization en $\in P$, i podem construir:

```
input N{
    Lps = []
    while N≠1{ ∈P=NP
        np = Factorization(N)
        Lps = Lps ∪ np
        N = N/np → petar dona error (np=PRIME)
    }
    return Lps
}
```

Aquest algoritme extreu una llista de tots els factors primers de N, fa un nombre relativament primer d'iteracions, per tant el logaritme respecte N: $O(\log_2(N)) \rightarrow O(\log_2(2^{|N|})) \rightarrow \rightarrow O(|N|)$

Si N és primer, ~~l'algoritme sortirà~~ l'algoritme donarà error en la primera iteració.