

Générer la clé publique

1 Descriptif

L'objectif de ce défi est d'implémenter la méthode permettant de générer la clé publique de l'algorithme de RSA. Pour ce défi, on pourra ajouter des getters aux attributs P , Q , N , ϕ et e de la classe `GenerateurDeClesRSA`. Pour générer une clé publique RSA, il faut :

- Choisir deux grands nombres premiers distincts P et Q .
- Calculer $N = PQ$.
- Calculer $\phi = (P - 1)(Q - 1)$.
- Choisir un nombre e premier avec ϕ .

2 Protocole

1. Une fois la connexion établie, le serveur commence par envoyer un premier message annonçant le début du défi :

-- Début du défi : Generer cle publique --

Ce message n'attend pas de réponse.

2. Vous devez envoyer une série de quintuplet (P, Q, N, ϕ, e) de nombres binaires.
3. Pour chaque quintuplet, le serveur enverra un message commençant par "OK" ou "NOK" suivant si celui-ci convient ou non. Sauf pour le dernier quintuplet pour lequel le serveur enverra "FIN" si celui-ci est correcte.
4. Le serveur terminera la communication par le message "FIN", votre client devra alors fermer la socket. Aucune réponse n'est attendue.

3 Exemple de communication

Voici un exemple (incomplet) d'une communication pour ce défi. Dans cet exemple les "<" et ">" indiquent le sens de transfert de chaque message et ne doivent pas être présents dans la communication.

```

< -- Début du défi : Generer cle publique --
> 1001000001010101101101101011110110010011001000000010001110010011
> 1110110101011011100000000011101000110000100011011000101101100011
> 1000010111010010111100001111000100100000110000100001110111101110...
> 1000010111010010111100001111000100100000110000100001110111101101...
> 1010101101011111010110011000001110010100001101101001101100100101...
< OK
> 1011111000000011001000010110101100101001011111000011101110110011
> 1100011110010010001010000011011011010101000100011010101001110111
> 1001010000100000111010101001110010001100011001001011110111100010...
> 1001010000100000111010101001110010001100011001001011110111100000...
> 1101111010101011101110101011010010000000100110010101101110111000...
< OK
> 1111010010110101100101100011110110110110110011100110000101011111
> 1100101101001110100101100111010000111000100000010010111010001011
> 1100001001010111000111010100100000110101000101101100110000000111...
> 1100001001010111000111010100100000110101000101101100110000000101...
> 1110100010111000010011011110110101110101010110000001100110111000...
< OK

```