Laboratoire 16

- **Buts**
  - Implanter la méthode de cryptographie à clé publique proposée en 1973 par Clifford Christopher Cocks, connue sous le nom de RSA.
- **Travail à réaliser (individuellement)**
  - Implanter un programme qui demande à l'utilisateur deux nombres premiers $p \neq q$ ainsi qu'un nombre e premier avec $(p-1) \cdot (q-1)$ (on se limitera à des valeurs $p \cdot q$ et $e < 2^{31} - 1$)
  - Le programme devra vérifier que p et q sont très probablement premier avec un test rapide.
  - Le programme devra ensuite calculer d, l'inverse de e modulo $(p-1) \cdot (q-1)$ avec l'algorithme d'Euclide étendu.
  - Le programme devra afficher la clé publique (n, e) ainsi que la clé secrète d.
  - Finalement, le programme devra vérifier que $(m^e \mod n)^d \mod n \equiv m \quad \forall m < n$, autrement dit, que l'on arrive bien à retrouver univoquement le message d'origine m à partir du message crypté $c = m^e \mod n$.
- **Délai**
  - Mercredi 3 novembre, avant minuit. Le laboratoire sera noté.

Méthode de cryptographie à clé publique

Processus d'échange de message secret avec un canal non sécurisé

- Bob veut transmettre un message secret à Alice. Il demande à cette dernière une clé publique.
- Alice génère deux nombres premiers distincts p, q et e premier avec $(p-1)\cdot(q-1)$, plus petit que cette valeur.
- Elle transmet à Bob la valeur de e et celle de $n = p \cdot q$ (qui forment la clé publique, sans révéler p et q qui doivent rester secrets)
- Bob chiffre son message $m < n$ en calculant $c = m^e \mod n$ et transmet c à Alice.
- Alice calcule d, l'inverse de e modulo $(p-1)\cdot(q-1)$.
- Alice retrouve le message secret m de Bob en calculant $c^d \mod n$.

Cette méthode se base sur le fait que l'on ne sait pas factoriser efficacement un nombre comportant plusieurs centaines de chiffres. Pour ce labo, on se contentera de nombres avec moins de 9 chiffres !

Test rapide de primalité (gratuit et sans rendez-vous)

Propriétés d'un nombre p premier :

- $\forall a < p, a^{p-1} \equiv 1 \mod p$
- 1 n'a que 2 racines carrées modulo p : (1 et $p-1$)

Input: $p \in \mathbb{N}$
Output: `false` si p non premier ; `true` si p probablement premier

```
1  if p < 2 then return false;
2  if p = 2 then return true;
3  begin Répéter 10 fois
4  |   Générer un nombre aléatoire a < p
5  |   if a^{p-1} ≠ 1 mod p then return false;
6  |   q = 1; u = p − 1
7  |   while u pair and q = 1 do
8  |   |   u ← u/2
9  |   |   q ← a^u mod p
10 |   |   if q ≠ 1 and q ≠ p − 1 then return false;
11 |   end
12 end
13 return true
```

Algorithme d'Euclide étendu

Input: $a, b \in \mathbb{N}$ avec $a > b$
Output: d, inverse de $b \mod a$ si a et b sont premiers entre eux, plus grand diviseur commun r entre a et b

```
1 r ← a ; r' ← b ; d ← 0 ; d' ← 1
2 while r' ≠ 0 do
3 |   q ← ⌊r/r'⌋
4 |   rs ← r ; ds ← d
5 |   r ← r' ; d ← d'
6 |   r' ← rs − q · r' ; d' ← ds − q · d'
7 end
8 if d < 0 then d ← d + a;
9 return d, r
```