(Cryptographie à clé publique : RSA)

Exercice 0 :

On considère l'ensemble $Z_{30} = \{0, 1, 2, \ldots, 29\}$ des entiers modulo 30. Rappelons qu'un élément $a \in Z_{30}$ est inversible si, et seulement si, pgcd(a, 30) = 1.

1) Énumérer tous les éléments de Z_{30}^*(les éléments de Z_{30} inversibles).
2) Calculer l'inverse dans Z_{30}^* des éléments trouvés à la question précédente.

Exercice 1 : Factorisation

1) En admettant que l'entier ***14803*** est le produit de deux nombres premiers, pouvez-vous facilement le factoriser à la main? Estimer le nombre maximal de tests pour trouver la dite factorisation.
2) Si en outre, on révèle que $\phi(14803) = 14560$, la factorisation est-elle possible ? Donnez les deux facteurs.
3) Écrire une fonction ***int premier(int n)*** qui permet de renvoyer **1** si n est premier est **0** sinon.
4) Écrire une fonction ***void factorisation(int n)*** qui permet de factoriser **n** en produit de deux nombres premiers.

Exercice 2 : Théorème de chinois

Alice change sa clé RSA tous les *25 jours*. Bob lui change sa clé tous les *31 jours*. Sachant qu'Alice change sa clé aujourd'hui et que Bob a changera sa clé dans les trois jours qui arrivent, déterminer le nombre de jours quand sera la prochaine fois qu'Alice et Bob changeront leur clé le même jour.

N.B : Notons *d* le nombre de jours jusqu'à ce que Alice et Bob changent leur clé le même jour.

Exercice 3 :

Bob utilise le protocole RSA et publie sa clé publique N = 187 et e = 3.

1) Encoder le message m = 15 avec la clé publique de Bob.
2) En utilisant le fait que $\phi(N) = 160$, retrouver la factorisation de N, puis la clé privée de Bob.

Exercice 4

Bob_1 et Bob_2 ont pour clé publique RSA respectivement (N, e_1) et (N, e_2) avec e_1 et e_2 premiers entre eux. Alice envoie le même message m crypté par les clés publiques RSA de Bob_1 et Bob_2 en c_1 et c_2.

Expliquer comment Oscar, qui intercepte les deux messages cryptés et qui connait les clés publiques de Bob_1 et Bob_2, peut retrouver le message clair m.

Exercice 5 :

1) Écrire une fonction ***void two_primes(int n, int prime[])*** qui permet de générer aléatoirement deux nombres premiers < n dans le tableau prime.
2) Écrire une fonction ***void keys(int prime[],int Pk[],int Sk[])*** qui permet de générer aléatoirement deux clés l'une publique (Pk) et l'autre privée (Sk).