Exercices
Semaine 14

Cours Turing

1 Chercher efficacement des grands nombres premiers

Dans cet exercice, on vous propose de chercher des grands nombres premiers en utilisant tout ce que vous avez vu jusqu'à maintenant, à savoir :

i) un générateur de nombres aléatoires pour tirer au hasard nombre N à 100 chiffres. Pour cela, vous pouvez utiliser celui implémenté en Python, qui fait appel au package random. Voici par exemple un moyen simple de générer un nombre aléatoire à 100 chiffres :

```
N = random.randint(10**99,10**100-1)
```

ii) le test de Fermat qui teste si, pour un nombre donné N, et un nombre tiré au hasard A, $A^{N-1} \pmod N = 1$ (à répéter avec 30 valeurs différentes de A pour minimiser la probabilité de faire une erreur).

NB : En Python, $A^B \pmod N$ s'écrit pow(A, B, N).

iii) et si vous voulez encore optimiser votre code, vous pouvez d'abord tester si le nombre N n'est multiple ni de 2, ni de 3, ni de 5 (nous vous laissons réfléchir comment tester ceci rapidement).

2 Exponentiation rapide

Ecrire un programme qui calcule efficacement $A^B \pmod N$, en vous basant sur ce que vous avez vu en cours. Il s'agit donc :

- d'utiliser la décomposition binaire du nombre B ;

- de calculer successivement $A^2 \pmod N$, $(A^2)^2 \pmod N$, etc. (en stockant éventuellement ces calculs préliminaires dans un vecteur), puis d'effectuer les multiplications appropriées (toujours $\pmod N$) pour trouver le résultat final.

Bonus : Vous pouvez aussi écrire une version récursive de ce programme !