MA 203: TD6 - Test de primalité
Prise en main de la GMP

27 mars 2017

Le but de ce TP est de programmer le test de primalité de Fermat. Le langage de programmation est C. Étant donné que nous aurons à manipuler des grands entiers, nous utiliserons la bibliothèque multi-précision GMP :
http ://gmplib.org/
installée sur vos machines, dont la documentation se trouve à http ://gmplib.org/manual/

Dans l'en-tête des fichiers utilisant cette bibliothèque, il faut mettre (dans cet ordre) :

```
#include <stdio.h>
#include <gmp.h>
```

Pour compiler, on doit faire le lien avec la bibliothèque :

gcc monprog.c -lgmp

Un entier GMP a le type mpz_t. Il doit au préalable être initialisé par la fonction mpz_init :

```
{
      mpz_t integ;
      mpz_init (integ);
      ...
      mpz_add (integ, ...);
      ...
      mpz_sub (integ, ...);
      ...
      mpz_clear (integ);
    }
```

Lorsque l'on n'utilise plus un entier, on libère l'espace mémoire occupé par cet entier en appelant la fonction mpz_clear, comme dans l'exemple ci-dessus.

Les opérations de base sur les entiers mpz_t sont décrites dans http ://gmplib.org/manual/Integer-Functions.html

Pour générer un nombre premier, la plupart des méthodes consistent à tirer un entier aléatoirement et à tester sa primalité. Un moyen simple de tester la primalité d'entiers est le *test de Fermat*, basé sur le théorème (de Fermat) : si n est un nombre premier, alors $a^{n-1} \equiv 1 \bmod n \ \forall a \in \mathbb{Z},\ (a, n) = 1$. Soit n l'entier dont on veut tester la primalité. Une conséquence du théorème de Fermat est que si, pour un entier a premier avec n, $a^{n-1} \not\equiv 1 \bmod n$, alors cela prouve que n est composé.
D'autre part, l'existence d'entiers a tels que $a^{n-1} \equiv 1 \bmod n$ ne garantit évidemment pas la primalité de n. On a même la définition suivante : soit n un entier impair composé, et a un entier, $1 \leq a \leq n-1$. Si $a^{n-1} \equiv 1 \bmod n$, n est dit *pseudo-premier en base a*. Un nombre *pseudo-premier en base a* n'est pas nécessairement premier. Toutefois, pour une base a donnée, les pseudo-premiers composés sont peu nombreux.

1. Proposer un test de primalité utilisant le théorème de Fermat. Expliciter le résultat renvoyé par un tel algorithme.

Étapes de programmation :

2. Écrire une fonction "Fermat" qui prend en entrée un entier n et teste sa primalité selon le test ci-dessus.
3. Écrire une fonction "GenPremier" qui génère un entier premier congru à 3 modulo 4 d'une taille donnée en argument.