Valentin THOUVENIN

Clément BELLANGER

Rapport Miller-Rabin

Dépôt Git : https://github.com/Kleyment/Miller-Rabin

(Explication de la compilation dans le README github)

Implémentation des fonctions élémentaires

Question 1

Nous avons choisi d'opter pour le langage java.
Celui-ci dispose de la librairie BigInteger. Il gère les entiers de très grande taille.
Il possède la plupart des opérations élémentaires dont :
add(BigInteger val) :
subtract(BigInteger val)
multiply(BigInteger val)
divide(BigInteger val)
mod(BigInteger val)

Pour plus de détails :
https://docs.oracle.com/javase/8/docs/api/java/math/BigInteger.html

Question 2

Un nombre aléatoire est un nombre qui considérant des bornes ou une précision fixé, a autant de chance d'être choisi qu'un autre nombre délimité par ces bornes ou cette précision.
On utilisera la bibliothèque Random (BigInteger a un constructeur prenant en paramètres Random et un nombre de bits fixé).

Question 3

Test de la fonction decomp sur 10000 valeurs de 1024 bits dans le fichier testDecomp.txt (dans ressources)

Question 4

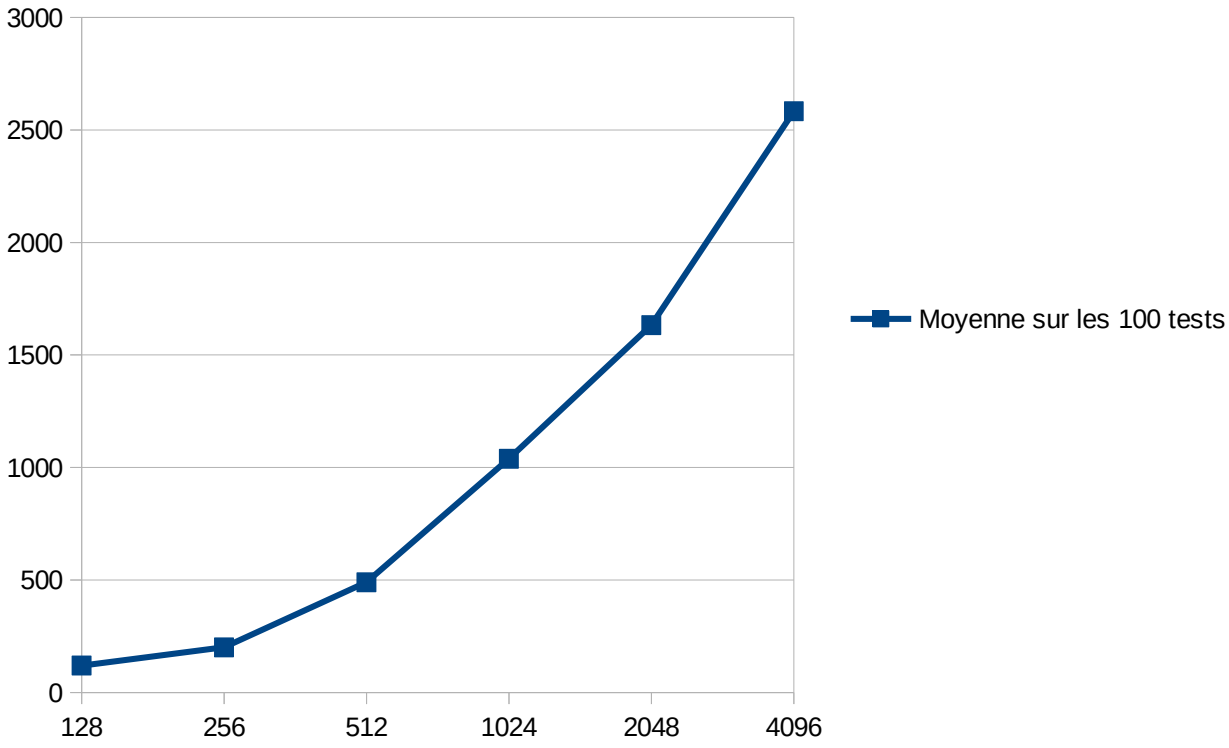
Test de la fonction expMod sur 10000 valeurs de 1024 bits dans le fichier testExpMod.txt (dans ressources)

Question 6

Test de n1 : probablement premier
Test de n2 : composé
Test de n3 : probablement premier

Question 8

Nombre de bits	Moyenne sur les 100 tests
128	120
256	201
512	490
1024	1039
2048	1633
4096	2583



Question 9

La moyenne de la valeur de Compteur est proportionnelle au nombre de bits.
On constate que le courbe ne croit pas exponentiellement, on peut donc en déduire que la complexité est en temps polynomial.

Question 10

Le test déterministe qui prouve de manière sûre la primalité d'un nombre est le test ECCP (Elliptic curve primality proving).
Il implique des courbes elliptique. Sa complexité en pratique est polynomiale (environ O(log(n)^5)).