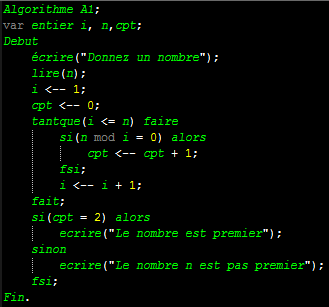
***Rapport TP n°1***

**Algorithme 1 :**

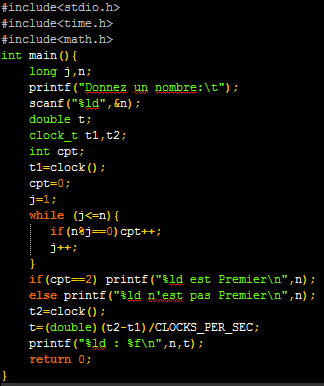
1. Algorithme de calcul de primalité (n de 1 à n) :



1. Calcul de la complexité :

1+1+1+1+n+1+2n+2(n/2 + 1) +2n+2=6n+9=O(n)

1. Algorithme en C :

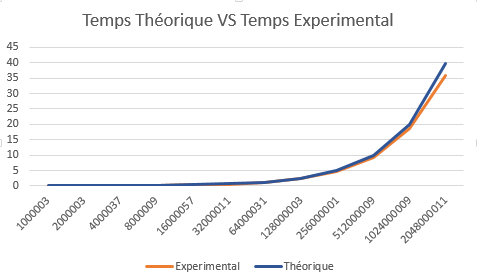


1. Comparaison de la complexité théorique et expérimentale :

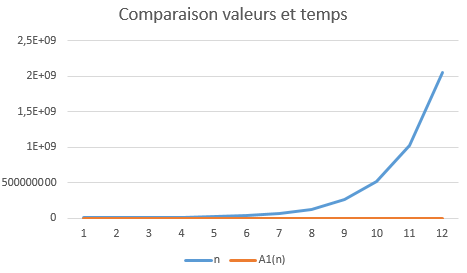
On a : F(n) = (6n+9) Δt

On calcule Δt : On a F(4000037) = 0,078

Donc : (6\*4000037+9) Δt = 0,078 alors Δt = 0,078 / (6\*4000037+9)

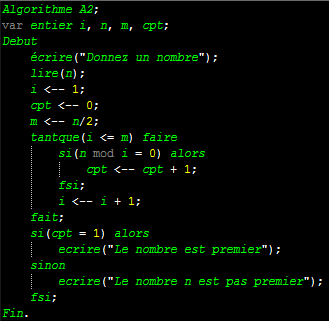


1. Représentation Graphique :



**Algorithme 2 :**

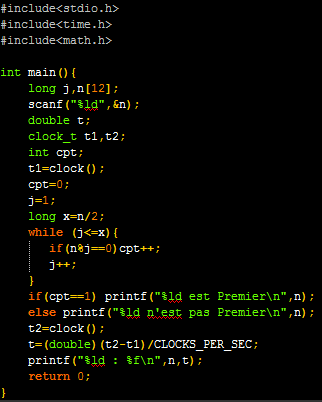
1. Algorithme de calcul de primalité (n de 1 à n/2) :



1. Calcul de la complexité :

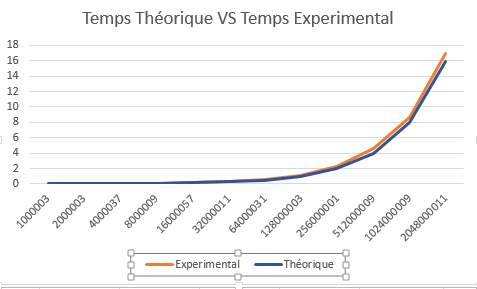
1+1+1+1+1+n/2+1+2n/2+2(n/2) +2n/2+1+1=7n/2+8=O(n)

1. Algorithme en C :



1. Comparaison de la complexité théorique et expérimentale :

On calcule le Delta de la même façon, puis compare les résultats.



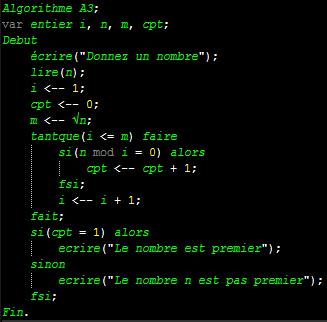
1. Comparaison Graphique :



On voit clairement que l’algorithme 2 s’exécute en moins de temps que l’algorithme 1 donc c’est le plus performant.

**Algorithme 3 :**

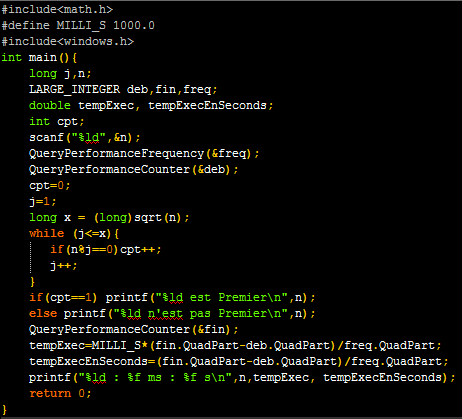
1. Algorithme de calcul de primalité (n de 1 à √n) :



1. Calcul de la complexité :

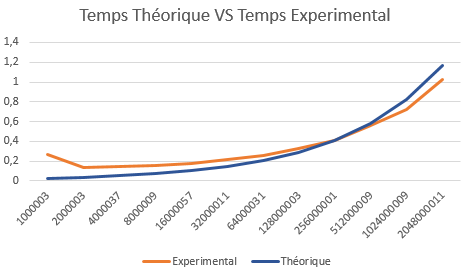
1+1+1+1+1+√n+1+2√n+2√n+2√n+1+1=7√n+8=O(n1/2)

1. Algorithme en C :



1. Comparaison de la complexité théorique et expérimentale :

On calcule le Delta de la même façon, puis compare les résultats.



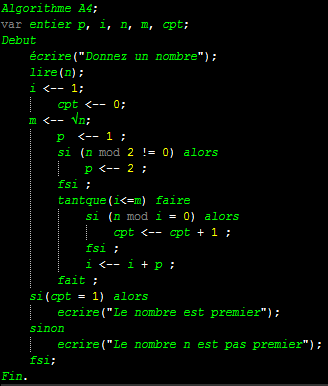
1. Représentation Graphique :



On voit à peine la courbe de l’algorithme 3 comparé en 2 premiers algorithmes, c’est évident que l’algorithme 3 est plus performant.

**Algorithme 4 :**

1. Algorithme de calcul de primalité (n de 1 à √n) :



1. Calcul de la complexité :

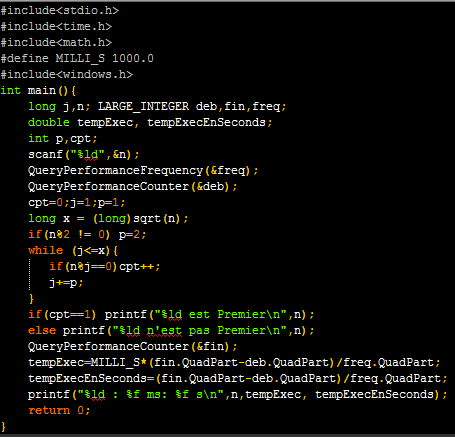
Si Paire : La boucle (√n+1+2√n+2√n+2√n = 7√n+1)

Si Impaire : La boucle (√n/2 +1+2√n/2+2√n/2+2√n/2 = 7√n/2+1)

On prend le Max.

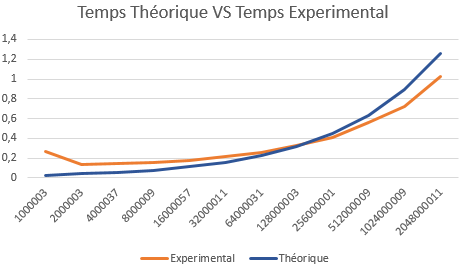
1+1+1+1+1+1+2+1+7√n+1+1+1=7√n+12=O(n1/2)

1. Algorithme en C :

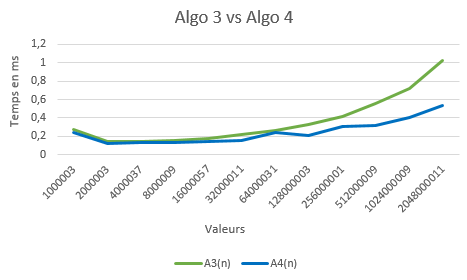


1. Comparaison de la complexité théorique et expérimentale :

On calcule le Delta de la même façon, puis compare les résultats.



1. Représentation Graphique :



En comparant, les algorithmes 3 et 4, on voit qu’on gagne du temps en utilisant l’algorithme 4, donc l’algorithme le plus efficace dans ces 4 algorithmes est le 4ème puisqu’il est mieux que le 3ème et le 3ème est mieux que les 2 premiers.

Les Donnés obtenus à l’exécution des programmes :

A1-A2 en secondes

A3-A4 en millisecondes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n | A1(n) | A2(n) | A3(n) | A4(n) |
| 1000003 | 0,031 | 0,015 | 0,268446 | 0,288296 |
| 2000003 | 0,031 | 0,016 | 0,139894 | 0,17534 |
| 4000037 | 0,078 | 0,031 | 0,144148 | 0,170614 |
| 8000009 | 0,141 | 0,063 | 0,156908 | 0,181012 |
| 16000057 | 0,312 | 0,234 | 0,180539 | 0,203697 |
| 32000011 | 0,547 | 0,266 | 0,220712 | 0,234417 |
| 64000031 | 1,078 | 0,531 | 0,258048 | 0,276953 |
| 128000003 | 2,313 | 1,074 | 0,323742 | 0,342174 |
| 256000001 | 4,61 | 2,203 | 0,411176 | 0,445204 |
| 512000009 | 9,125 | 4,656 | 0,557214 | 0,60164 |
| 1024000009 | 18,501 | 8,626 | 0,724047 | 0,785015 |
| 2048000011 | 35,861 | 16,954 | 1,023213 | 1,053933 |