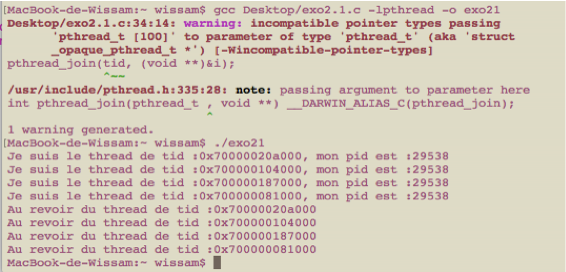
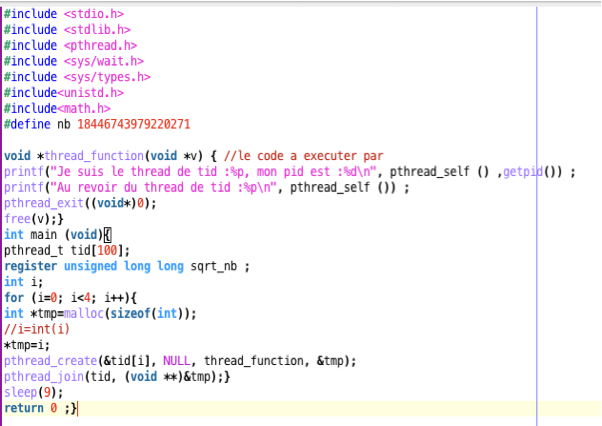
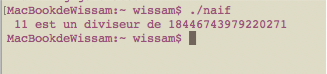
**EXERCICE 2 :**

1. Exécution de code naïf qui vérifie si le nombre nb est premier : après avoir compiler ce code avec la commande  
   gcc nomprgm.c -lpthread -o executable  
   Et l’exécuter nous avons obtenu le résultat suivant :
2. La construction des P threads étant choisir P=4 :



3

3. Tester la primalité de nombre nb en utilisant P threads : a/ La corps de la fonction de thread :

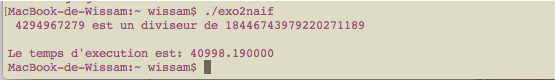


b/ Le corps de la fonction main :

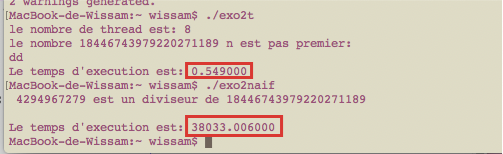


4

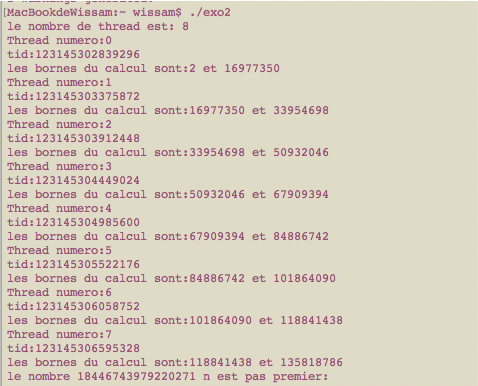
c/ Compilation et exécution de l’algorithme naïf :



Nous voyons bien que le temps d’exécution est très grand(en seconde) nous allons par suite le comparer par celle d’un programme multithreadé.



Le processeur avait utiliser 8 threads pour tester la primalité de ce nombre en subdivisant un intervalle allant de 2 à racine(nb).



5

**EXERCICE 3 :**

Compilation et l’exécution du code :

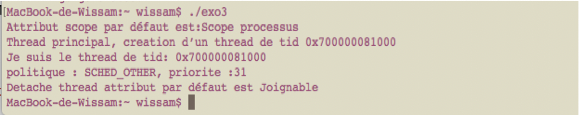
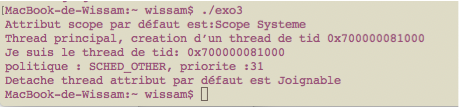
1.

a)

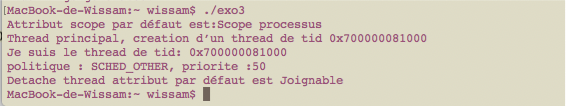
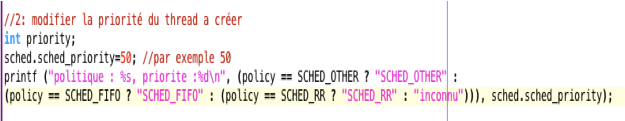
La différence entre un scope système et un scope processus est que le scope système sera en concurrence pour le processeur avec d’autres thread système. Par contre les scope processus partagent un seul noyau en concurrence pour le processeur seulement dans le processus en cours d’exécution.

b)

Modification de la valeur de l’attribut scope :

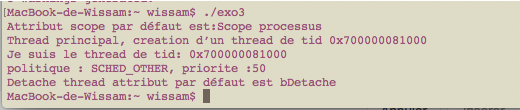
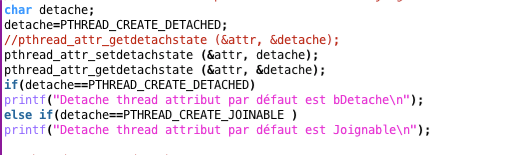


2. Modification de la priorité de thread à créer :

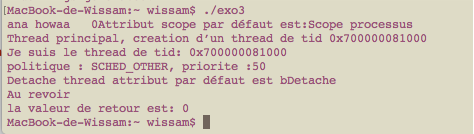
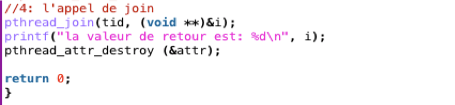


6

3. Modifier la valeur de l'attribut detachstate pour créer un thread injoignable



4.  
a) Ajout de l’appel de la fonction pthread\_join :



b) On peut définir l’utilité de l’utilisation de threads détachés lorsque le programme principale n’aura pas de besoin de valeur de retour de tel thread pour continuer son exécution. Par contre les threads joignables vont servir à fournir des valeurs de retour en sortie au programme père qui seront ensuite nécessaires dans un calcul ou de testes etc.