```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <sys/wait.h>
#include <sys/types.h>
#include <math.h>
#include <unistd.h>

//#define nb 18446743979220271
unsigned long long nb;
pthread_t tids[10];
int nb_threads;

void * isprime(void * var) { //le code a executer par un thread
int * num=var;

printf("Thread num +++++\n ");
int numthread;
numthread=*num;
register unsigned long long sqrt_nb=sqrt(nb);
register unsigned long long debut=2+ (numthread * sqrt_nb)/
nb_threads;
register unsigned long long fin=2+((numthread+1)* sqrt_nb)/
nb_threads;

printf("Thread num:%d, tid:%p, les bornes du calcul:%llu,%llu \n",
numthread, pthread_self (), debut, fin) ;

 register unsigned long long i;
 int j;
 for(i=debut;i<fin; i++)
  if (!(nb % i))
  {
  printf("Thread %p: %llu est un diviseur de %llu
\n",pthread_self(),i,nb) ;
  //il faut annuler les autres threads
  for(j=0;j<nb_threads;j++){
          if (!pthread_equal (pthread_self (), tids[j]))
                  pthread_cancel(tids[j]);
  }
  free(var);
  pthread_exit((void*)0);
  }

  pthread_exit((void*)1);
}

int main(){

nb=18446743979220271;
nb_threads=sysconf(_SC_NPROCESSORS_ONLN);


int i;
```

```
int * a;


for(i=0;i<nb_threads;i++){
a=malloc(sizeof(int));
*a=i;
pthread_create(&tids[i], NULL,isprime,(void*)a);
printf("Thread principal, creation d'un thread de tid  %p
\n",tids[i]);
}

//attendre la fin du thread tid et récupérer son retour dans la
variable i
printf("Thread principal, attente terminaison threads: \n");

unsigned int r,rt;

for(i=0;i<nb_threads;i++){
pthread_join(tids[i], (void **)&r);
rt+=r;
}

if(rt==0)  printf("Le chiffre %llu n'est pas un nombre premier
\n",nb);
else  printf("Le chiffre %llu est un nombre premier\n",nb);

return 0;
}
```