SOLUCIÓN PROBLEMA 1

El problema fue resuelto en lenguaje C++ con los siguientes razonamientos.

Para la resolución del problema se programó la función de fibonacci en sus dos versiones, el método recursivo dado por el enunciado y como ecuación dependiendo solo de n:

$$F_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right]$$

Dada la longitud de los números a calcular se utilizaron arrays de enteros para almacenarlos. La respuesta del inciso B fue dada por la longitud del array utilizado, que fue de 309 cifras. Esto en principio se pensó en almacenarse en “long double” que almacena hasta 308 cifras, a con una precisión de 18 cifras. Esto fue descartado obviamente.

La mayor complicación vino en calcular la respuesta al inciso A, en la cual pedía listar todos los n (menores que 90) para los cuales fib de n es un número primo. Si observamos fib(90) es un número de 19 cifras, lo cual es posible almacenarlo en “unsigned long long”. Esto es importante para poder usar la librería matemática (math.h) y sus operadores. Aún así atacar este problema por fuerza bruta es un poco exagerado, con lo cual se prosiguió a analizar matemáticamente el problema.

Siguiendo este libro: “Estructuras de matemáticas discretas para la computación” de Kolman, se escoge este algoritmo para determinar cuándo un número es primo:

Algoritmo para probar si un entero $N > 1$ es primo:

PASO 1. Verifique si N es 2. Si lo es, N es primo. Si no lo es, prosiga con el
PASO 2. Verifique si $2 \mid N$. Si esto es afirmativo, N no es primo; de lo contrario, prosiga al
PASO 3. Calcule el entero más pequeño $K \leq \sqrt{N}$. Luego
PASO 4. Verifique si $D \mid N$, en donde D es cualquier número impar tal que $1 < D \leq K$. Si $D \mid N$, entonces N no es primo; de lo contrario, N es primo.

La prueba para saber si un entero es primo es una tarea común para computadoras. El algoritmo que se acaba de ver es demasiado ineficiente para números muy grandes, pero hay muchos otros algoritmos para probar si un entero es o no primo.

Es simple de entender, pero poco eficiente computacionalmente. Actualmente existen Test de Primalidad estadísticos o el famoso método planteado en 2002 “AKS”, que son mucho más eficientes (ver https://es.wikipedia.org/wiki/Test_de_primalidad#Test_de_primalidad_AKS). Por cuestiones obvias de tiempo no se desarrolló ninguno de estos.

Bajo este panorama tenemos una forma de determinar si un número es primo. Esto en fib(90) determina dividir por todos los primos menores que su raíz cuadrada, lo cual nos da un número de 10 cifras. Pero, ¿No hay alguna relación entre los números primos y los números de fibonacci? ¿Todos los números de fibonacci pueden ser primos o algunos son obvio que no son?

En busca de responder esta respuesta surge la siguiente demostración matemática:

Observemos los primeros números de la sucesión:

$$F_3 = 2, F_4 = 3, F_5 = 5, F_7 = 13, F_{11} = 89, F_{13} = 233 \ldots$$

Si prestamos atención en los índices, vemos los números n tales que F_n es un número primo. Los que tenemos arriba son $3, 4, 5, 7, 11 \text{ y } 13$.
Si nos olvidamos del 4, todos los demás son números primos.
¿Se tendrá que sí p es un número primo entonces F_p también lo es?
Bueno, el 2 fallaría, pero ¿qué pasa con los siguientes? Después del 13, los siguientes números primos son 17, 19, 23 y 29:

$$F_{17} = 1597, F_{19} = 4181, F_{23} = 28657, F_{29} = 514229.$$

¿Son números primos? Pues no, 3 de ellos sí, pero nos falla $4181 = 37 \times 113$.
Hemos visto que salvo $F_4 = 3$, los primos de Fibonacci que nos han ido saliendo tienen todos un índice primo. ¿Será esto verdad?
(ver http://www.zurditorium.com/la-sucesion-de-fibonacci-y-los-numeros-primos)

Teorema 1: Todo primo de Fibonacci distinto a 3 tiene como índice un número primo.
Teorema 2: Si un número m divide a un número n entonces F_m divide a F_n.

Tomemos un primo de Fibonacci F_n. Si n no es un número primo, existirá un número natural $m \neq 1$ y menor que n que divida a n, pero entonces por nuestra afirmación tendremos que F_m divide a F_n por lo que F_n no puede ser un número primo. Por tanto no puede pasar que n no sea un número primo...
¿Ha sido sencillo? Pues ojo, algún error tiene que tener la demostración ya que si estuviese bien, al ser $F_4 = 3$ un número primo deduciríamos que 4 es un número primo.
Bien, el error estar en decir que si F_m divide a F_n entonces F_n no es un número primo.
Para poder afirmar esto, primero tenemos que asegurarnos de que F_m no es ni 1 ni el propio F_n. Está claro que si n>2 entonces F_m no es F_n ya que la sucesión va creciendo y $m < n$. Pero la pega está en que podríamos tener que m=2 y por tanto $F_m = 1$. Y eso es lo que pasa con el número 4, su único divisor distinto a él mismo y a 1 es el número 2 y por eso nos falla en dicha demostración. Sin embargo para cualquier otro número mayor que 4

y que no sea primo, podremos encontrar un divisor distinto a sí mismo y mayor que 2. ¿Cómo podemos afirmar esto? Pues si escribimos n como $n = p \cdot q$, con p un número distinto a 1 y distinto a n, si p es distinto a 2 habríamos terminado, y si no lo es, pues q tiene que ser distinto a 2 puesto que n no es 4.
Ahora sí, teniendo en cuenta este pequeño detalle, hemos probado que salvo el $F_4 = 3$, todo primo de Fibonacci tiene que tener índice primo.

Nos quedaría solo probar que si m divide a n entonces F_m divide a F_n. Si buscáis por internet, se puede ver que se deduce a partir de la siguiente curiosa propiedad de la sucesión de Fibonacci. Para n, m dos números, denotemos por $MCD(n, m)$ el máximo común divisor de dichos números, es decir, el mayor número entero que divide a ambos números, se tiene entonces que:

$$MCD(F_n, F_m) = F_{MCD(n,m)}.$$

A partir de esta propiedad sería inmediato probar lo que queremos. Pero claro, tendríamos que probar esta propiedad también. No es excesivamente complicado, pero como sale demasiado larga voy a probar por otro camino más corto. Aún así merecía la pena comentaros la relación de la sucesión de Fibonacci con el máximo común divisor. Voy a probar primero la siguiente propiedad:

$$F_{m+k} = F_{m-1}F_k + F_m F_{k+1},$$

donde m, k son números enteros positivos. Para ver esto, voy a fijar un número entero positivo k y voy a hacer inducción sobre el elemento m.

Primer paso: comprobamos que dicha igualdad es cierta para m=1,2. Y efectivamente, para m=1 se tiene que $F_{m-1} = F_0 = 0$ y $F_m = F_1 = 1$ y por tanto:

$$F_{m-1}F_k + F_m F_{k+1} = 0F_k + 1F_{k+1} = F_{k+1} = F_{m+k}.$$

Y para m=2 se tiene que $F_{m-1} = F_1 = 1$ y $F_m = F_2 = 1$ así que

$$F_{m-1}F_k + F_m F_{k+1} = 1F_k + 1F_{k+1} = F_{k+2} = F_{m+k}.$$

Segundo paso: sabemos que es válido hasta m=2. Supongamos que es válida hasta cierto n con n al menos 2.
Tercer paso: sabiendo que la igualdad se cumple para m=n, veamos si se cumple para m=n+1. Pues bien, tenemos que demostrar que la igualdad se cumple al cambiar m por n+1, es decir

$$F_{n+1+k} = F_n F_k + F_{n+1}F_{k+1}.$$

Por hipótesis de inducción (segundo paso) estamos suponiendo que la propiedad se cumple al cambiar m por n o por n-1, es decir, sabemos que:

$$F_{n+k} = F_{n-1}F_k + F_n F_{k+1}.$$
$$F_{n-1+k} = F_{n-2}F_k + F_{n-1}F_{k+1}.$$

Ahora bien, por definición tenemos que $F_{n+1+k} = F_{n+k} + F_{n-1+k}$ por lo que sumando las dos igualdades anteriores tendremos que F_{n+1+k} es igual a

$$F_{n-1}F_k + F_n F_{k+1} + F_{n-2}F_k + F_{n-1}F_{k+1}.$$

Sacando factor común por un lado F_k y por otro F_{k+1} se tiene que

$$F_{n+1+k} = (F_{n-1} + F_{n-2})F_k + (F_n + F_{n-1})F_{k+1}.$$

Por último, como sabemos que $F_n = (F_{n-1} + F_{n-2})$ y $F_{n+1} (F_n + F_{n-1})$ la igualdad anterior se convierte en

$$F_{n+1+k} = F_n F_k + F_{n+1} F_{k+1},$$

y con esto queda completado el tercer paso y con ello la prueba por inducción.

Bien, en el Paso 1 hemos visto que era cierto para 1 y 2. Como es cierto para 1 y 2, el Paso 3 nos permite deducir que es cierto para 3. Una vez que sabemos que es cierto también para 3, el Paso 3 nos permite deducir que también lo es para 4, luego para 5, luego para 6 y así con cualquier número natural.

Ya podemos ver que si m divide a n entonces F_m divide a F_n. Esta misma afirmación la reescribimos de la siguiente forma:
Si $n = d \cdot m$ entonces F_m divide a F_n.
Y esto lo vamos a demostrar de nuevo por inducción, en este caso sobre d:
Paso 1: Claramente es cierto para $d = 1$ ya que en ese caso $n = m$ y por ello $F_m = F_n$.
Paso 2: Supongamos que la afirmación es cierta para cierto d, es decir F_m divide a $F_{d \cdot m}$.
Paso 3: Veamos qué pasa para $n = (d+1) \cdot m$. Utilizando la igualdad

$$F_{m+k} = F_{m-1} F_k + F_m F_{k+1}$$

con $k = d \cdot m$ se tiene que

$$F_n = F_{m+d \cdot m} = F_{m-1} F_{d \cdot m} + F_m F_{d \cdot m+1}.$$

Ahora bien, por hipótesis de inducción (Paso 2) se tiene que existe un número entero q tal que $F_{d \cdot m} = q F_m$ y por tanto

$$F_n = F_{m-1} \cdot q \cdot F_m + F_m F_{d \cdot m+1} = F_m (F_{m-1} \cdot q + F_{d \cdot m+1})$$

y por tanto tenemos que F_m divide a F_n.

Bajo este teorema, hemos podado nuestro problema. En vez de tener que probar para todos los n < 90 si fib(n) es un número primo, basta probarlo para los n primos, y agregar el 4, que ya vimos que es primo.

Como solución se entrega el código, con el makefile correspondiente.