Problema 1:

Parte A:

La idea básica de 2-Color puede reducirse a la idea de que entre nodos padres e hijos de un grafo, los colores que se asignaran a esto siempre serán diferentes y se utilizaran no más de 2 colores. Tomando en cuenta este sencillo argumento, podemos recorrer el grafico en anchura mediante el algoritmo BFS (Bread-First-Seach), estableciendo colores diferentes a padres e hijos. Si no nos fuera posible ejecutar esta tarea, sería correcto afirmar que un grafo no puede colorearse con 2 colores. Dado que el algoritmo BFS se ejecuta en tiempo P, y 2-Color puede representarse en términos de este algoritmo, entonces es correcto asumir que 2-Color es un algoritmo capaz de resolverse en tiempo P.

Parte B:

El problema de 3-COLOR consiste en revisar cada nodo de un grafo revisando sus vecinos para verificar que no existan vértices adyacentes del mismo color. Es posible comprobar que cada uno de los vértices tiene color diferente a sus vecinos en un tiempo O(N^2), por tanto, es posible asumir que para trabajar este problema, se requiere un tiempo NP.

"3-COLORING = {, c1, c2, c3: there exists a function from nodes in G to {c1, c2, c3} such that no two adjacent nodes have the same color.}
An answer to the 3-COLORING problem is a function f from nodes in G = (V, E) to {c1, c2, c3}. In O(|V|^2) steps we can check every edge (u,v) in E to make sure that f(u) is not equal to f(v). Thus 3-COLORING is in NP."

-Fuente

Ahora bien, el problema de 3-COLOR también se puede considerar que está en la clase NP-Completo porque se puede verificar en tiempo polinomio y, además, puede ser reducido en tiempo polinomial a 3-SAT de 3-COLOR tomando como vértices las variables que son tomadas de las clausulas. Tomase en cuenta que si un problema NP-COMPLETO puede ser reducible polinomialmente a un NP, el problema NP en cuestión es NP-COMPLETO.

Cualquier K > 3 para 3 Color puede ser llevado a 3-Color haciendo, por cada 3 < i < K, agregar un vértice y conectarlo adyacente a todos los vértices originales (Fuente). Asumiendo que este enunciado es correcto, como 3-COLOR es NP y reducible 3-SAT, y 7-COLOR es reducible a 3-COLOR, es correcto afirmar que 7-COLOR pertenece a NP.

Problema 2:

Parte A:

Análisis Agregado

Inicialmente se pudo conseguir que el costo de las operaciones que se realizan en un árbol de Fenwick son O(nlogn). Esto sería el costo de búsqueda y modificación. Al aplicar el método agregado el costo promedio conseguido para la operación de búsqueda es O(logn). Teniendo en cuenta que n es el tamaño del arreglo dado inicialmente.

Costo de Búsqueda

$$\sum_{i=1}^{n} Ci$$

$$\sum_{i=1}^{n} log_2 n$$

$$T(n) = nlogn$$

$$\frac{T(n)}{n} = \frac{O(nlogn)}{n}$$

Ĉi= $O(\log n)$

Método Contable

Para el método contable se agregó la operación de crear el arreglo, y las operaciones restantes se quedaron con costo Log(n). De igual manera se le asignaron valores (entre paréntesis) que pensamos que pueden representar el costo real o amortizado de la operación.

Operations	Ci	Ĉi
Creation	n (1)	2n
Search	Log n (1)	0
Update	Log n (1)	0

Métodos Potencial

En el método potencial se pudo llegar a la conclusión de que la búsqueda y actualización, como fue planteado inicialmente. Cabe destacar que se necesita saber estado actual y estado anterior, pero en este caso no aplican por el hecho de que se hace una actualización en el mismo nodo que se está trabajando y no se toma en cuenta lo sucedido anteriormente.

$$\hat{c}_i = c_i + \Phi(D_i) - \Phi(D_{i-1})$$

Ĉi = log (n) + n-n

Ĉi = log(n)

Problema 3:

Parte A:

Asumamos a cada elemento como un nodo perteneciente a un grafo propio. Las restricciones pretenderán conectar los elementos en grafos donde una restricción dada por (A B), quiere decir que A se conecta a B. Ahora, con los elementos que han de estar separados en un grafo, se puede intentar colorear cada uno de los grafos mediante 2-Color. Debido a que este problema es reducible a 2-Color, y 2-Color es, como mencionado en otra parte del trabajo, reducible a BFS, es posible deducir que este problema puede ser resuelto en tiempo P mediante el algoritmo BFS.