**Nombres:**

* Pedro Alejandro Valderrama Tapias **Código:** 2879195
* Brayan Alexander Riascos Ruiz **Código:** 2879762
* Edgar Rafael Cruz Rodriguez **Código:** 2879720

**TALLER 1**

**1.** Desarrolle los siguientes ejercicios del libro [Cormen]

**a)** Ejercicio 3.1-2. Muestre que para cualquier real y , donde ,

**Solución:**

, donde y , donde.

Entonces

Podemos escribir,

Elevamos a la b todo:

Al comparar esto con lo anterior:

, tenemos:

y

por lo tanto

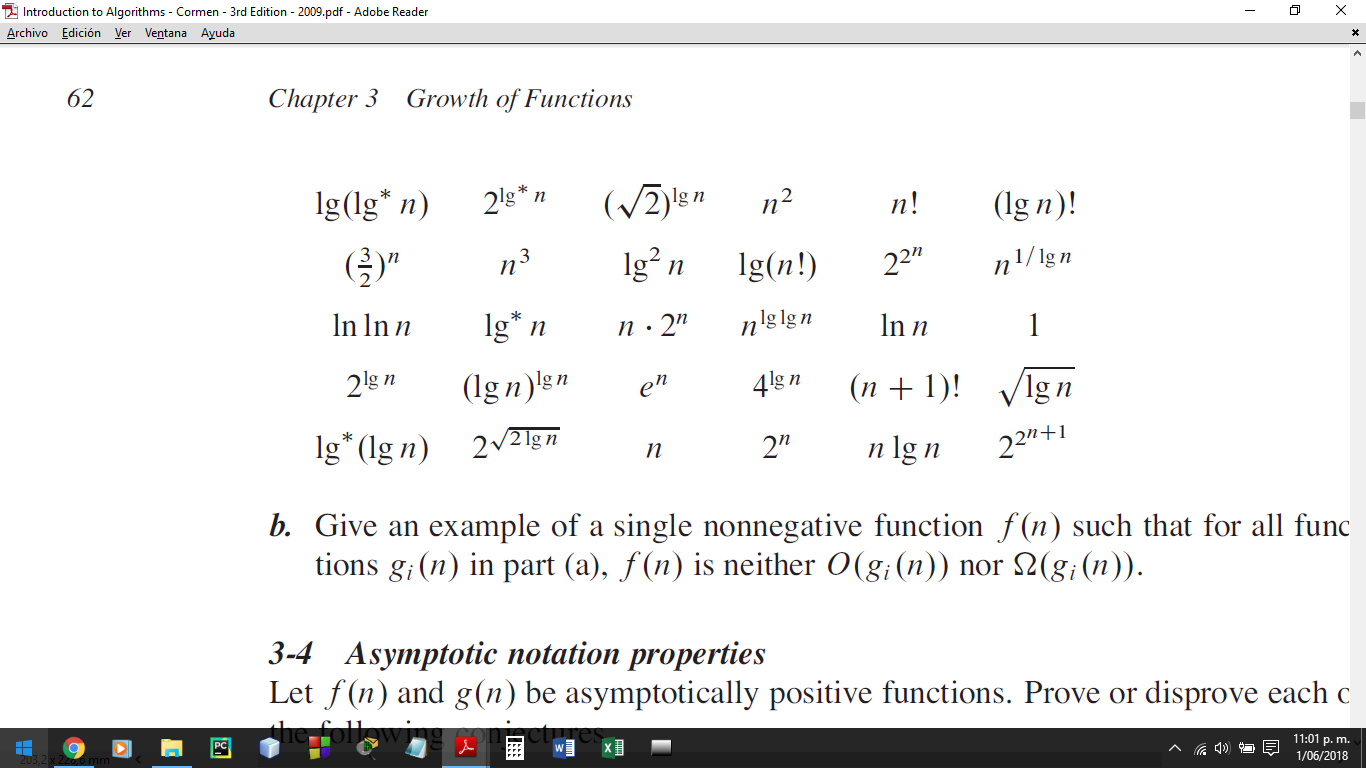
**b)** Ejercicio 3.1-7. Pruebe que es un conjunto vacío.

**Solución:**

Suponiendo que no es un conjunto vacío, es equivalente a decir que para todo c1, c2 > 0 tiene 0 <= c1 < <c2 donde n >= max (n1, n2).  
  
Hacemos c1 = c2, entonces llegamos a una contradicción, por lo que la suposición no es verdadera, luego se demuestra que es un conjunto vacío.

**c)** Problema 3.3.

**a.** Clasifique las siguientes funciones por orden de crecimiento, es decir, encuentre un arreglo de las funciones que satisfacen Particione su lista en clases de equivalencia de modo que las funciones y están en la misma clase si y sólo si



**Solución:**

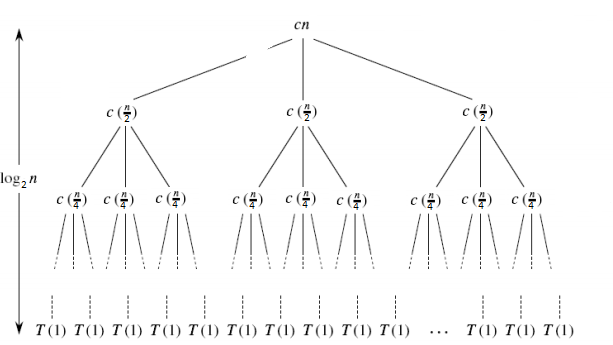
Orden de las funciones con su clase de equivalencia

**b.** De un ejemplo de una única función no negativa tal que para todas las funciones  
 en la parte (a), no es ni ni .

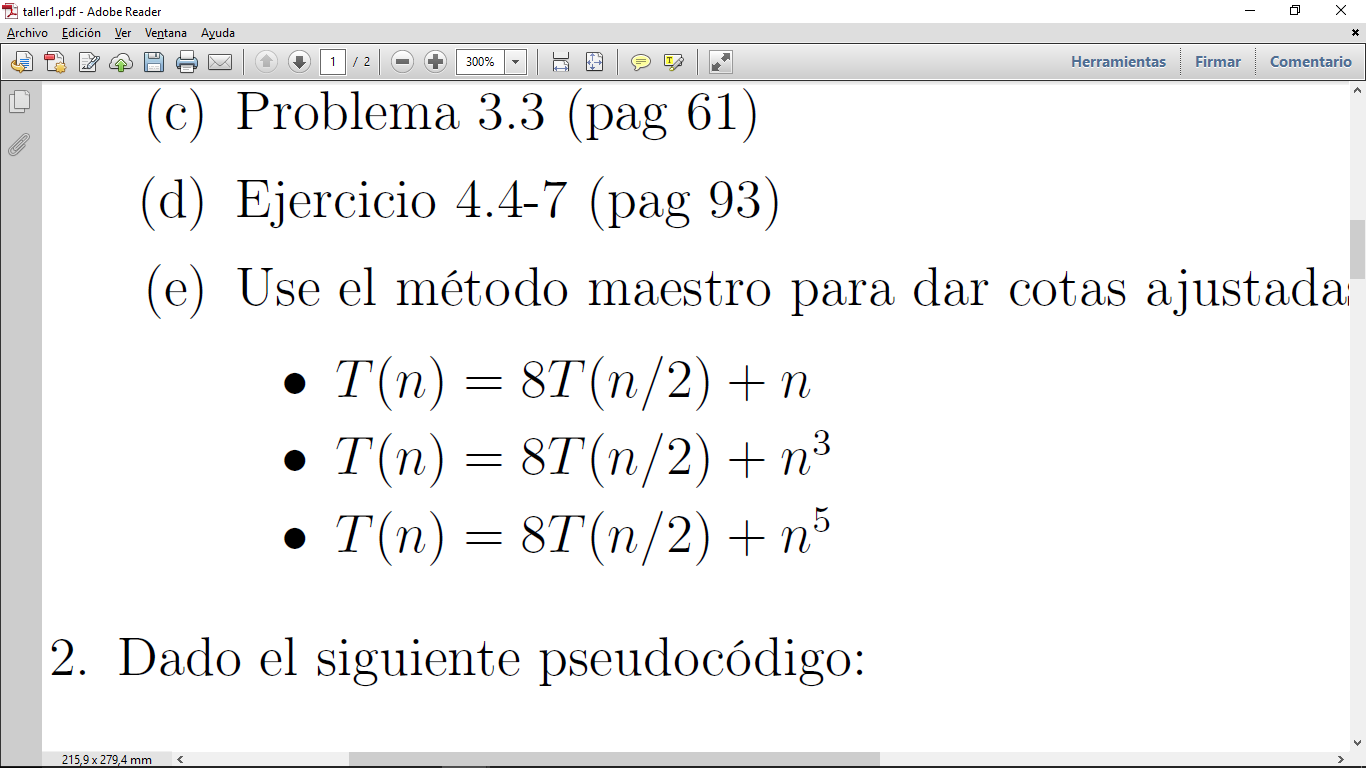
**Solución:**

**d)** Ejercicio 4.4-7. Dibuje el árbol de recursión para , donde es una constante, y proporciona un límite asintótico apretado en su solución. Verificar su límite por el metodo de sustitucion.

**Solución:**



**e)** Use el método maestro para dar cotas ajustadas para las siguientes recurrencias:



**Solución:**

Para tenemos que:

entonces como , la cota ajustada es

Para tenemos que:

entonces como , la cota ajustada es

Para tenemos que:

entonces como , la cota ajustada es

**2.** Dado el siguiente pseudocódigo:

def misterio(n):

if n <= 1:

return 1

else:

r = misterio(n / 2)

i = 1

while n > i\*i:

i = i + 1

r = r + misterio(n / 2)

return r

**a)** Plantee una ecuación de recurrencia para T(n), el tiempo que toma la función misterio(n).

**Solución:**

**b)** Dibuje el árbol de recursión y calcule:

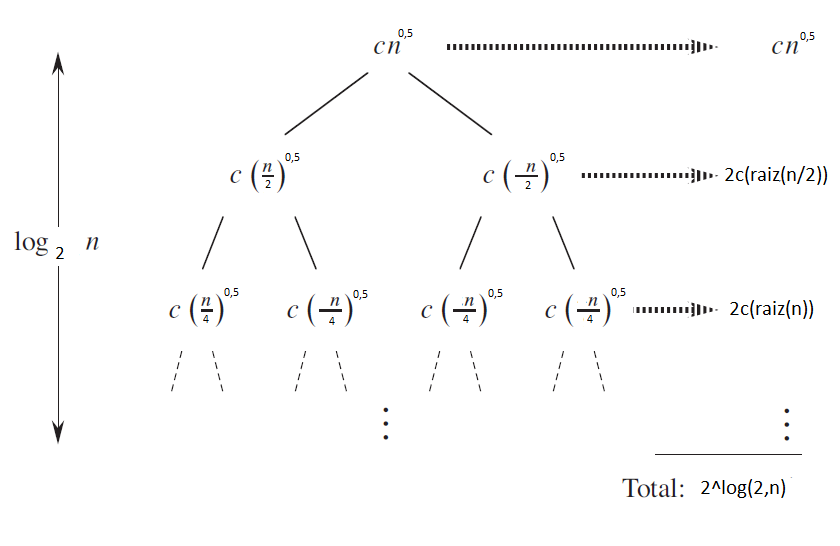
i. La altura del mismo

ii. El número de nodos por cada nivel

iii. La suma de los nodos de cada nivel

iv. La suma total

**Solución:**

****

i. La altura del árbol es .

ii. El número de nodos por nivel es.

iii. La suma de nodos de cada nivel es

iv. La suma total es

**c)** Determine el comportamiento asintótico de T(n) justificándolo de manera detallada.

**Solución:**

Por teorema maestro tenemos que =

=

**3.** Ejercicio 22.3-1. Haga un cuadro de 3 por 3 con etiquetas de fila y columna BLANCO, GRIS y NEGRO. En cada celda (i, j), indicar si, en cualquier punto durante una búsqueda en profundidad de un gráfico dirigido, puede haber un borde desde un vértice de color hasta un vértice de color . Para cada borde posible, indique qué tipos de borde puede ser . Haga una segunda tabla de este tipo para la búsqueda en profundidad de un grafo no dirigido.

**Solución:**

Dirigido

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Blanco** | **Gris** | **Negro** |
| **Blanco** | TBFC | BC | C |
| **Gris** | TF | TFB | TFC |
| **Negro** |  | B | TFBC |

No dirigido

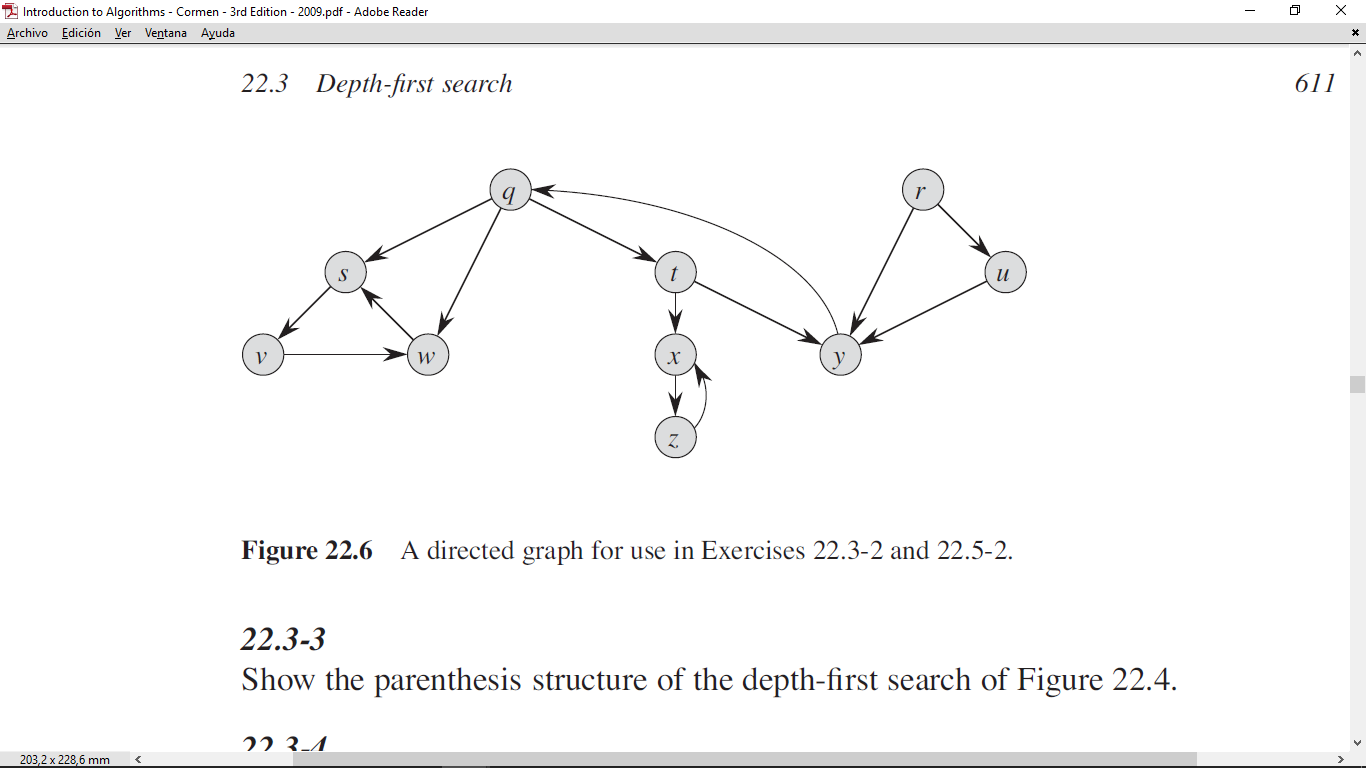
|  |  |  |  |
| --- | --- | --- | --- |
|  | **Blanco** | **Gris** | **Negro** |
| **Blanco** | TB | TB |  |
| **Gris** | TB | TB | TB |
| **Negro** |  | TB | TB |

Convenciones:

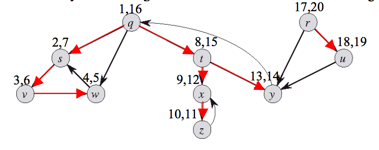
* **T** : Los bordes de los árboles son bordes en el bosque de profundidad . Borde es un borde de árbol si fue descubierto por primera vez explorando el borde
* **B** : Los bordes posteriores son esos bordes conectando un vértice a un antecesor en un árbol en profundidad. Consideramos que los bucles automáticos, que pueden aparecer en los gráficos dirigidos, tienen bordes posteriores.
* **F** : Los bordes delanteros son aquellos que no son bordes conectando un vértice a un descendiente en un árbol en profundidad.
* **C** : Los bordes cruzados son todos los otros bordes. Pueden ir entre vértices en el mismo árbol de profundidad, siempre que un vértice no sea un antepasado del otro, o pueden ir entre vértices en diferentes árboles de profundidad.

**4.** Ejercicio 22.3-2. Muestre cómo funciona la búsqueda de profundidad en el gráfico de la figura 22.6. Supongamos que el bucle for de las líneas 5-7 del procedimiento DFS considera los vértices en orden alfabético y supone que cada lista de adyacencia se ordena alfabéticamente. Muestre los tiempos de descubrimiento y finalización para cada vértice y muestre la clasificación de cada borde.

Figura 22.6



**Solución**

****

Hojas del árbol: (q, s), (s, v), (v, w), (q, t), (t, x), (x, z), (t, y), (r, u)

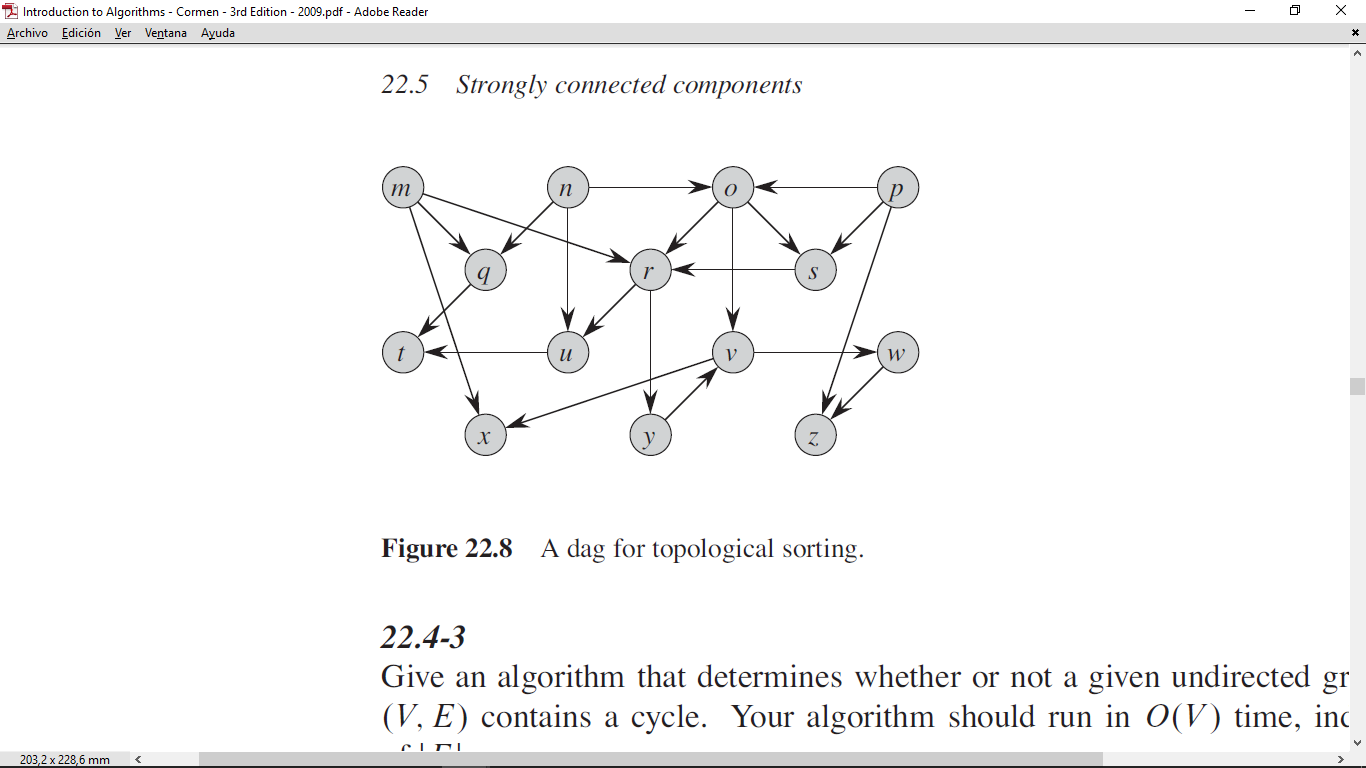
hojas traseras: (w, s), (z, x), (y, q)

Hojas delanteras: (q, w)

Hojas cruzadas: (r, y), (u, y)

**5.** Ejercicio 22.4-2. Proporcione un algoritmo de tiempo lineal que tome como entrada un gráfico acíclico dirigido y dos vértices y , y devuelve el número de caminos simples de a en . Por ejemplo, el gráfico acíclico dirigido de la Figura 22.8 contiene exactamente cuatro caminos simples desde el vértice al vértice : y . (Su algoritmo solo necesita contar las rutas simples, no las debe enumerar).

Figura 22.8



**Solución**

Agregue un campo a la representación de vértices para mantener un recuento de enteros. Inicialmente, establezca el recuento de vértices t en 1 y el conteo de otros vértices en 0. Comience a ejecutar DFS con s como el vértice de inicio. Cuando se descubre t, debe marcarse inmediatamente como terminado (NEGRO), sin que se inicie el procesamiento posterior. Posteriormente, cada vez que DFS finalice un vértice v, establezca el recuento de v en la suma de los recuentos de todos los vértices adyacentes a v. Cuando DFS finalice los vértices, detenga y devuelva el conteo calculado para s.