1. Explique que quiere decir la propiedad de *"Transitividad"* de $O()$.
2. ¿Cuál es la Regla del producto"para $O()$?
3. ¿Cuándo dos tasas de crecimiento del orden del tiempo de ejecución son equivalentes si?
4. ¿Cuál es la signatura de la función `insert()` en listas STL? Diga qué es cada una las variables y cuál es el tipo de retorno. En caso de que haya varias signaturas explique una de ellas.
5. ¿Por qué no es útil definir a un "iterator"sobre un árbol ordenado orientado como `typedef *cell` como se hace en listas simplemente enlazadas?
6. Explique con un ejemplo la condición de "prefijo"de un código.
7. ¿Cuál es el costo de inserción en tablas de dispersión abiertas con listas desordenadas en el peor caso?. Diga cual es ese caso.
8. ¿Cuál es al condición de *"Árbol Binario de Búsqueda"*?
9. ¿Qué quiere decir que un algoritmo de ordenamiento sea *"estable"*?.
10. ¿Por qué se les dice *"lentos"* a ciertos algoritmos de ordenamiento? Enumere los algortimos lentos que conoce.
11. ¿Puede tener una correspondencia varios valores iguales del dominio, o sea varias claves iguales? (Por ejemplo `M1={(1,2),(1,3)}`) ¿Y varios valores iguales del contradominio? (Por ejemplo `M2={(1,2),(3,2)}`)
12. ¿Qué retorna la dereferenciación de un iterator sobre correspondencias (clase `map`)?
13. ¿Qué ocurre si se invoca el `operator[]` sobre una correspondencia con una clave que no tiene asignación? Por ejemplo `M={(1,2),(3,4)}` y hacemos `x = M[5]`. ¿Da un error? ¿Qué valor toma `x`?
14. ¿Cuál es la condición de Arbol Binario de Búsqueda?
15. ¿Cómo se encuentra el mínimo y el máximo de los valores en un árbol binario de búsqueda? ¿Cuál es la complejidad algorítmica de esta operación (caso promedio, mejor y peor)?
16. ¿Porqué decimos que $(n+1)^2 = O(n^2)$ si siempre es $(n+1)^2 > n^2$?
17. Discuta si es posible insertar en una posición dereferenciable en árboles binarios.
18. Discuta el valor de retorno de `insert(x)` para conjuntos.
19. Defina árbol binario completo y árbol binario lleno. ¿Es verdad que todo árbol binario completo es lleno? ¿Y viceversa?
20. Discuta el número de inserciones que requieren los algoritmos de ordenamiento lentos en el peor caso.
21. Si la la correspondencia `M` `M={(1->2),(5->8)}` y ejecutamos el código `int x= M[5]`. ¿Que ocurre? ¿Que valores toman `x` y `M`? ¿Y si hacemos `x = M[3]`?
22. ¿Que retorna el constructor por defecto de lista?

23. ¿Cuál es el tipo de dereferenciación de un iterator a lista, para listas simplemente enlazadas?

24. ¿Cómo es la clase `cell` en listas simplemente enlazadas? ¿Y para listas doblemente enlazadas?

25. ¿Qué es la condición de prefijo? El código `(a->0)`, `(b->01)`, `(c->1)`, `(d->10)` lo satisface? ¿Para qué necesitamos que el código satisfaga esta condición?

26. ¿Para qué sirve el algoritmo de Huffmann para generar códigos? ¿En qué casos es bueno? ¿Da una solución *"heurística"* o da el código óptimo.

27. ¿Cómo se traduce la *condición de prefijos* para códigos representados por árboles? El árbol `(. (. a b) (c . (` ¿lo satisface?

28. El siguiente código: `(. (. a b) (. . (. c d)))`, ¿es óptimo? ¿Qué es un código redundante?

29. Hufmann: ¿De qué dependen las profundidades de los nodos en el árbol?

30. ¿Qué pasa si hago una copia espejo del árbol de Hufmann?

31. ¿Cuándo es *"bueno"* el algoritmo de Hufmann? Es decir, ¿Cuándo da un código sensiblemente mejor que, por ejemplo, un código de longitud fija?

32. Como se verifica si un código de Huffmann no es redundante viendo el árbol correspondiente?

33. Nombre diversas relaciones de orden que se pueden usar con los algoritmos de ordenamiento.

34. ¿Qué condiciones debe cumplir un predicado binario para ser una relación de orden?

35. ¿Cuál es el costo de inserción en tablas de dispersión?

36. ¿Cómo es el costo de `find()` para las posibles implementaciones de correspondencias?

37. ¿Cuál es el tiempo de ejecución del algoritmo de búsqueda binaria?

38. ¿De qué depende el costo de las operaciones en tablas de dispersión cerrada? ¿Cómo se define el parámetro de tasa de llenado?

39. Supongamos que tenemos una función recursiva `fun(tree<int> T,tree<int>::iterator n )`, y se aplica al árbol `(1 (2 3 4) (5 6 7))`, ¿Cómo es el estado de llamadas cuando se está procesando el nodo 4? ¿Y para 5, 6 y 7?

40. ¿Cuál es la signatura de las funciones que se pasan como relaciones de orden?

41. ¿Cuál es la condición para que un árbol sea árbol binario de búsqueda?

42. ¿Qué retorna la dereferenciación de un iterator?

43. Suponiendo que se tienen conjuntos representados por vectores de bits, siendo el conjunto universal los enteros en $[0, 10)$. ¿Cómo se representa el conjunto $S = \{4, 6, 7\}$?

((document-version "aed-2.0.2-21-g95eccb0 'clean")
(document-date "Tue Mar 4 20:58:03 2008 -0200")
(processed-date "Tue Mar 4 20:58:05 2008 -0200"))