Apuntes de Análisis de Algoritmos y Estructuras de Datos

PREGUNTAS GENERALES

1.- ¿Qué es un TAD?

Un TAD es un conjunto de valores y operaciones que se definen mediante una especificación independiente de cualquier representación.

El calificativo de abstracto expresa la cualidad de independencia de la representación.

2.- ¿Qué importancia tiene la especificación en la creación y utilización de un TAD?

La especificación define el dominio que tomarán los valores de los objetos pertenecientes al TAD.

Define como usarlos mediante una especificación sintáctica que nos indique las reglas a seguir para realizar una operación y una especificación semántica que exprese el significado de cada operación.

La especificación es la única parte que conoce el usuario del mismo y contiene el nombre del tipo y la especificación de las operaciones.

Resumiendo, es la parte más importante de cada TAD.

3.- ¿Qué es la especificación sintáctica?

Indica cómo usar las operaciones, es decir, la forma en que se ha de escribir cada una, dando su nombre junto al orden y tipo de operandos y resultado.

4.- ¿Qué es la especificación semántica?

Expresa el significado de las operaciones, o sea, que hace y que propiedades tiene cada una.

5.- ¿Qué es la implementación de un TAD?

La implementación de un TAD es la parte conocida solamente por el diseñador del TAD y contiene su representación (estructura de datos) y la implementación de las operaciones.

6.- ¿Cómo se realiza la ocultación en C?

Mediante la abstracción de las implementaciones de las operaciones.

Debería ser también la ocultación de los tipos de datos, pero esto en C no es posible ya que no podemos ocultar el archivo de cabecera .h

Las implementaciones se introducen en un archivo .c oculto (que sería el archivo .o de ese fichero) y se le proporcionaría al usuario del TAD especificándole las operaciones que puede utilizar.

7.- Principio fundamental de los TAD: independencia de la representación

El principio de independencia de la representación nos dice que no importa como esté implementado un TAD mientras se cumpla la especificación del mismo.

Esto permite que un mismo TAD se pueda representar mediante diferentes implementaciones y que el usuario del mismo lo pueda utilizar sin importarle como esté implementado.

8.- ¿Existe alguna relación entre la ocultación de la información y la independencia de la representación?

Si, ya que gracias a la ocultación de la información conseguimos que la especificación de un TAD sea independiente de su implementación.

9.- Ante la posibilidad de que no se verifiquen las precondiciones de una operación de un TAD, ¿Qué decisiones de diseño puede implementar la operación?

Lo ideal sería que diese un error, llegando incluso a detener la ejecución del programa. Es muy importante la verificación de las precondiciones de la especificación para una implementación independiente en el TAD.

10.- ¿Qué sucede si después de la ejecución de una determinada operación, no se cumplen las precondiciones de las mismas?

Esto no suele suceder ya que una determinada operación que pertenezca a la función se va a poder realizar siempre y cuando se cumplan las precondiciones de dicha función. En el caso en el que no se cumplan las precondiciones, podría dar un resultado erróneo o indeterminado.

PILAS

11.- ¿Qué es una pila?

Secuencia de elementos de un tipo determinado en la cual se pueden añadir y eliminar elementos sólo por uno de sus extremos, llamado tope o cima.

En una pila el último elemento añadido es el primero en salir de ella, por lo que se también se les conoce como estructuras LIFO: Last input First Output.

12.- ¿Cuándo utilizaremos una representación estática del TAD pila? Ventajas e inconvenientes.

Ventajas: Especialmente indicada cuando utilizamos durante todo el programa pilas del mismo tamaño. Ocupa menos espacio que la implementación del TAD Pila mediante celdas enlazadas ya que no requiere un puntero por elemento.

Inconvenientes: Exige fijar el tamaño máximo de la pila en tiempo de compilación, por lo que el tamaño tendrá que ser una constante y siempre el mismo para todas las pilas que creemos.

13.- ¿Cuándo usaremos una representación dinámica del TAD Pila? Ventajas e inconvenientes.

Ventajas: El tamaño de las pilas es variable, con lo que podemos incrementarlo o decrementarlo cuando queramos en tiempo de ejecución.

Siempre ocupará el espacio justo para almacenar los elementos que contenga la pila en cada momento.

Desventajas: Alto consumo de memoria debido a la utilización de un puntero por cada elemento.

14.- ¿Cuándo utilizaremos una representación pseudoestática del TAD Pila? Ventajas e inconvenientes.

Ventajas: Nos permite fijar distintos tamaños en tiempo de ejecución (este tamaño será fijo durante el tiempo de vida de la pila), por lo que un tamaño inicial no determinará el tamaño de todas. Ocupa menos espacio que el TAD Pila mediante celdas enlazadas.

Inconvenientes: Tamaño constante durante la vida de cada pila, tamaño que tenemos que ir controlando para no sobrepasarlo.

15.- ¿Por qué en la representación estática y pseudoestática de un TAD Pila, el registro que representa a la pila se pasa por referencia sin ser necesario?

No es necesario pasarlo por referencia. Sin embargo, en virtud del principio de independencia de la representación, los parámetros del tipo pila tienen que ser transferidos por referencia, en las diferentes representaciones.

16.- ¿Tiene algún sentido representar una pila con una lista doble enlazada?

No, ya que en una lista doblemente enlazada se usan 2 punteros (uno para el elemento siguiente y otro para el anterior) y en las pilas solo se inserta y se elimina por el tope, con lo que es un gasto innecesario al valer con un único puntero.

17.- ¿Cuándo utilizaremos la representación circular del TAD Pila?

Esta representación no existe para el TAD Pila

Tal y como se define el TAD pila, una pila es una secuencia de elementos de un tipo determinado en la cual se pueden añadir y eliminar objetos sólo por uno de sus extremos, llamado tope o cima, por lo tanto, carecería de sentido utilizar un puntero que se utilizara para apuntar a la base de la pila (primer elemento que se introduciría en ella).

18.- ¿Cómo se podría eliminar un elemento cualquiera de una cola/pila?

No se podría hacer, ya que tanto para pilas como para colas tenemos una especificación que ha de cumplirse, y en ninguna de las dos, se nos dice cómo hacerlo. Una forma de hacerlo sin saltarnos la especificación sería ayudarnos de una pila/cola auxiliar en la cual volcamos la que tenemos, hacer pop en el elemento que queramos eliminar y después hacer otro volcado.

19.- ¿Por qué en las implementaciones mediante celdas enlazadas, tanto de pilas como de colas, no existe un método de llena?

No existe el método de llena en estas implementaciones debido a que el tamaño es dinámico y no está definido, y, por tanto, el tamaño de la estructura variará en tiempo de ejecución según requiera el usuario.

20.- ¿En qué programa utilizarías el TAD Pila?

En aquellos casos en los que el usuario desee extraer los elementos en el orden inverso en que se insertaron y viceversa. También si se desea que todas las operaciones del TAD tengan orden constante.

Algunos programas comunes en los que se usa el TAD pila con por ejemplo líneas de texto.

COLAS

21.- ¿Qué es una cola?

Una cola es una secuencia de elementos en las que las operaciones se realizan por los extremos:

- Las eliminaciones se realizan por el extremo llamado inicio, frente o principio de la cola.

- Los nuevos elementos son añadidos por el otro extremo, llamado final de la cola

En una cola el primer elemento añadido es el primero en salir de ella, por lo que también se le conoce como estructura FIFO: First Input First Output.

22.- Ventajas e inconvenientes de una representación vectorial del TAD cola.

Ventajas: Ocupa un espacio fijo de memoria, el cual está representado por una estructura que contiene un vector que almacena todos los elementos, un entero que almacena el tamaño máximo del vector y otro entero que almacena el fin de la cola

Inconvenientes: Hay que conocer el número aproximado de datos que se van a introducir en la cola. Y, además, en la operación Pop(), habría que mover todos los elementos de la cola de una posición anterior, por lo que sería muy costoso.

23.- Ventajas e inconvenientes de una representación circular del TAD cola.

Ventajas: Esta representación soluciona el inconveniente de la representación vectorial. Y, además, como el principio y el fin de la cola son consecutivos, el orden de las operaciones de eliminación e inserción se reducen.

Inconvenientes: Distinguir entre una cola llena y una vacía, ya que la posición de los índices de inicio y fin serían la misma en ambos casos. Las soluciones propuestas serían añadir un indicador de si está o no vacía la cola o añadir una posición más al vector la cual siempre estaría vacía. Si inicio=fin, cola vacía.

24.- Eres un diseñador del TAD Cola y un usuario te solicita una operación de acceso al n-ésimo elemento de la misma. ¿Incluirías esta operación en el TAD?

No, ya que esta operación no es una operación del TAD Cola (No está definida en la especificación del TAD Cola y esta no se puede modificar). Se tendría que crear otro TAD correspondiente a las necesidades del usuario.

25.- ¿Por qué en la implementación vectorial de las colas, existe una posición que siempre está vacía en el vector?

Para distinguir cuando la cola está vacía o llena.

26.- La representación de una cola mediante estructuras enlazadas, ¿Por qué se representa usando dos punteros a cada extremo de la misma?

La representación de una cola mediante estructuras enlazadas podría realizarse utilizando un solo puntero a uno de los extremos de la misma.

Una posible representación usando solo un puntero sería apuntar al fin de la misma y enlazar el último elemento con el primero, de forma que se accedería al fin de la cola en coste O(1) y al inicio en coste O(2).

Otra posibilidad acorde con los accesos al TAD Cola, es representarla mediante punteros que apunten al inicio y final de la cola. Su ventaja es que a ambos extremos se accede con coste O(1). Su inconveniente es que requiere dos punteros = más memoria.

27.- ¿Para qué se utiliza el nodo cabecera en el TAD Cola, en su implementación con celdas enlazadas?

El nodo cabecera no se utiliza en el TAD Cola, es innecesario dado que ésta surge para resolver un problema de independencia de la representación en el TAD Lista, cuando definimos la posición en esta.

28.- ¿Con la implementación de colas mediante un vector circular de tamaño n, cuantos elementos pueden almacenarse?

Se pueden almacenar n-1 elementos, ya que una de las posiciones del vector se debe reservar para poder distinguir cuando está llena y cuando vacía la cola.

29.- Con la representación de colas mediante una estructura enlazada, con puntero al final y circular ¿Cuántos elementos pueden almacenarse?

Pueden almacenarse todos los elementos que se quieran, ya que la representación es dinámica y por tanto no hay un límite de elementos a almacenar.

30.- Comente la siguiente afirmación: El TAD Cola Circular es un TAD representado mediante un vector circular en el que el número de elementos a insertar como máximo es n-1

El TAD Cola Circular no existe, pero si existe una representación circular del TAD Cola, en el que se pueden almacenar n-1 elementos, debido al indicador de cola llena/vacía.

31.- ¿En qué programas usarías el TAD Cola?

En aquellos casos en los que el usuario desee extraer los elementos en el mismo orden en el que se insertaron. También si se desea que todas las operaciones del TAD tengan orden constante.

LISTAS

32.- ¿Qué es una lista?

Una lista es una secuencia de elementos de un tipo determinado. Se representa de la forma L= (a1, a2….an), donde n es mayor que 0 y longitud igual a n. Si n es igual a 0, es una lista vacía.

Posición: Lugar que ocupa un elemento en la lista. Los elementos están ordenados de forma lineal según las posiciones que ocupan. Todos los elementos salvo el primero tienen un único predecesor, y todos los elementos salvo el ultimo tienen un único sucesor.

Posición Fin: Posición siguiente a la del último elemento. Esta posición no está ocupada nunca por ningún elemento.

33.- ¿En una representación vectorial de una lista, como representamos la posición fin?

La posición fin en una lista está representada mediante la longitud

34.- Ventajas e inconvenientes de la implementación vectorial del TAD Lista

Ventajas: Las posiciones podrían representarse por enteros y, además, se puede permitir el acceso directo a un elemento en concreto por dicho entero.

Inconvenientes: Las operaciones insertar() y eliminar() pueden ser muy costosas y otro inconveniente sería especificar el tamaño máximo de la lista.

35.- Ventajas e inconvenientes de la implementación dinámica del TAD Lista

Ventajas: No se desaprovecharía memoria con el uso de punteros, ya que usan el espacio necesario.

Inconvenientes: Si la lista es demasiado pequeña, puede ser que se "desaproveche" más espacio para albergar al puntero de cada nodo que a la propia lista

36.- ¿Por qué surge el nodo cabecera en el TAD Lista?

Porque si no existiese el nodo cabecera, nos estaríamos saltando el principio de la independencia de la representación, así pues, utilizando el nodo cabecera estaremos cumpliendo la especificación del TAD en las funciones insertar y eliminar para que todos los elementos tengan predecesor y así ninguno tenga que tratarse de forma diferente.

Con el nodo cabecera se consigue reducir el orden de complejidad de las operaciones insertar y eliminar, de O(n) a O(1), ya que con la cabecera no es necesario recorrer la lista en busca de la posición anterior a la que se desea insertar o eliminar.

Las operaciones buscar() anterior () y fin() siguen siendo de orden O(n).

37.- ¿Cuándo se utiliza un TAD Lista Circular?

Cuando no hay definido un primer y un último elemento de la lista, es decir, cuando no se sabe cuáles son los extremos de la lista, por lo que se puede acceder a todos los nodos a partir de uno cualquiera. También hay que tener en cuenta que es innecesario el nodo cabecera y la operación fin().

38.- Diferencia entre TAD Lista Circular y TAD Lista

La diferencia principal entre estos dos TADS es que, en la Lista Circular, a diferencia de la Lista, todos los elementos tienen estrictamente un sucesor y un predecesor. Una Lista Circular no posee posición primera() ni posición fin(), solo posee una posición llamada inipos() que devuelve una posición para tomarla como referencia. También cabe decir que el TAD Lista Circular, a diferencia del TAD Lista, no posee cabecera ya que es innecesaria.

39.- ¿Para qué se utiliza una implementación mediante estructura enlazada doble?

Se utiliza para reducir el orden de las operaciones anterior y fin, de orden n a 1, ya que, en esta implementación, existe, además del puntero a la posición siguiente, un puntero a la anterior. La operación buscar () sigue siendo la única operación con orden n que no se puede mejorar.

40.- ¿Cuándo es conveniente utilizar una implementación mediante estructura enlazada doble?

Es conveniente utilizar esta implementación cuando se vayan a realizar operaciones como anterior() y fin(), para evitar el mayor coste de éstas.

41.- En la lista Circular ¿es posible implementar alguna operación de orden logarítmico?

No, todas las operaciones son de orden 1 o de orden n, ya que se referencia al elemento al cual apunta el puntero o al contiguo, o para las operaciones de orden n, recorre toda la lista circular hasta hallar el elemento que desea.

42.- ¿Qué condición tiene que cumplir una lista para que la búsqueda sea logarítmica? ¿Y orden cuadrático?

Una lista no podrá tener orden logarítmico en la búsqueda. Y una búsqueda de orden cuadrática no tendría sentido, ya que el orden normal de las búsquedas es orden n, el orden cuadrático lo empeoraría.

43.- ¿Por qué la especificación del TAD Lista en la operación anterior, las precondiciones son L= (a1...an) 2 menor o igual que p, menor igual que n+1?

El hecho de que p deba ser mayor o igual que 2 y menor o igual que n más 1, viene dado porque la primera posición no tiene anterior y la posición fin, aunque no tenga elementos, si posee una posición anterior.

44.- ¿Qué pasaría si insertaras o eliminaras en la primera posición en una implementación de la lista mediante estructura enlazada sin cabecera?

Se perdería la asignación de la posición al no tener el primer elemento un predecesor definido, hecho que con la cabecera no pasaría

45.- Representando el TAD Lista mediante un vector, ¿Es posible evitar el coste n para inserciones y borrados en los extremos?

Si, implementando mediante un vector circular

46.- ¿En qué programas usarías el TAD Lista?

Lo usaremos en aquellos casos en los que se desee insertar y eliminar elementos en posiciones concretas de la estructura.