# Apunte teórico Taller de Programación

*Ordenación y matriz*

#### ¿Que es un algoritmo de ordenación?¿Por qué es importante en arreglos?¿Qué métodos de ordenación conoce?

un **algoritmo de ordenamiento** es un algoritmo que pone elementos de una lista o un vector en una secuencia dada por una relación de orden, es decir, el resultado de salida ha de ser una permutación —o reordenamiento— de la entrada que satisfaga la relación de orden dada. Ordenamientos eficientes son importantes para optimizar el uso de otros algoritmos (como los de búsqueda y fusión) que requieren listas ordenadas para una ejecución rápida. También es útil para poner datos en forma canónica y para generar resultados legibles por humanos.

Hay diferentes métodos de ordenación, por ejemplo

**Selección:** Se busca el mínimo elemento de la lista, se intercambia con el primero, se busca el mínimo en el resto de la lista y se intercambia con el segundo, etc

**Inserción:** Para cada elemento de la lista o vector se toma el siguiente y se lo inserta de forma ordenada entre los elementos ya ordenados

**Intercambio:** Funciona revisando cada elemento de la lista que va a ser ordenada con el siguiente, intercambiandolos de posición si están en el orden equivocado. Es necesario revisar varias veces toda la lista hasta que no se necesiten más intercambios, lo cual significa que la lista está ordenada.

#### ¿Cómo funciona el método de intercambio?¿Cuantas pasadas sobre el vector requiere?

Se comparan e intercambian datos adyacentes según corresponda.

Requiere n-1 pasadas.

El mayor elemento queda al final del arreglo, por lo que no es necesario volver a compararlo.

#### ¿Que es una Matriz?¿Cuales son sus características importantes?¿Por medio de qué tipo de datos se pueden indexar sus elementos?

Un tipo de dato Matriz es una **colección ordenada e indexada** de elementos.

Esta estructura de datos compuesta permite acceder a cada componente utilizando **dos índices** (fila y columna) que permiten ubicar un elemento dentro de la estructura.

Homogénea: Todos sus elementos son del mismo tipo.

Estática: Su espacio en memoria no cambia durante la ejecución del programa.

Acceso indexado: Accede a sus elementos a través de un índice.

Lineal: Sus elementos tienen una relación 1 a 1.

Sus elementos pueden ser de cualquier tipo de datos y ser indexados unicamente por cualquier tipo ordinal: Entero, Carácter, Subrango.

*Merge*

#### ¿En qué consiste la operación merge?¿Como se procede para realizarlo?

La operación de Merge consiste en generar una nueva estructura de datos ordenada a partir de la mezcla de dos o más estructuras de datos previamente ordenadas.

Las estructuras que se combinan guardan el mismo orden lógico interno (por ejemplo datos ordenados alfabéticamente)

Se procede de la siguiente manera:

1. Se comparan los elementos de cada una de las estructuras que se combinan y se guarda el más pequeño en la estructura resultante.
2. Se avanza al siguiente elemento en aquella estructura de la cual se obtuvo el valor más pequeño

*Recursión*

#### ¿En qué consiste una solución recursiva?¿Cómo se logra?¿Que es el Caso Base o Caso Degenerado?¿Qué condiciones debe cumplir un algoritmo recursivo?

Una solución recursiva resuelve un problema por resolución de instancias más pequeñas del mismo problema

Para lograrlo:

1. El módulo realiza invocaciones a sí mismo, en cada una el problema se reduce
2. Existe un caso distinto que se resuelve de manera particular o directa. Llamado Caso Base o Caso Degenerado

Un algoritmo recursivo debe cumplir:

* Debe tener alguna condición de terminación
* El llamado recursivo, la acción debe acercar a la condición de terminación

#### ¿Que es una búsqueda dicotómica?¿Cuántas comparaciones debe realizar en el peor de los casos?

Es un algoritmo de búsqueda que encuentra la posición de un valor en un array ordenado. Compara el valor con el elemento en el medio del array, si no son iguales, la mitad en la cual el valor no puede estar es eliminada y la búsqueda continúa en la mitad restante hasta que el valor se encuentre. La búsqueda binaria es computada en el peor de los casos en un tiempo logarítmico, realizando O ( log ⁡ n ) comparaciones.

*Árbol*

#### ¿Que es un Árbol?¿Cuales son sus características principales?¿Qué elementos posee?

Es una estructura de datos que satisface 3 propiedades:

* Cada elemento del árbol se relaciona con 0 o más elementos (hijos)
* Si el árbol no está vacío hay un único elemento raíz, el cual no tiene padre (predecesor)
* Todo otro elemento posee un único padre y es un descendiente de la raíz

Homogéneo: Todos sus elementos son del mismo tipo

Dinámico: Puede aumentar o disminuir durante la ejecución del programa

No lineal: Cada elemento puede tener 0,1 o más sucesores

Acceso secuencial: Se debe recorrer la estructura para acceder a sus elementos

Raíz: El nodo superior del árbol

**Hijo:** Un nodo conectado directamente con otro cuando se aleja de la raíz.

**Nivel:** El nivel de un nodo se define por 1 + (el número de conexiones entre el nodo y la raíz).

Subárbol: Conjunto de nodos de un árbol con un padre que no es la raíz

#### ¿Cómo se relacionan los nodos de un árbol binario?¿Por qué es útil?

Los ABB son estructuras de tipo árbol organizadas de forma eficiente para la búsqueda, cada nodo tiene un valor que es más grande que todos los nodos del subárbol izquierdo y más pequeño que todos los nodos del subárbol derecho

Su utilidad más importante son las búsquedas cuyo tiempo medio es O(log n)

#### ¿Cómo se construye un árbol binario?

Se insertan nodos de forma recursiva, si el valor es nil se inserta el nodo, si el valor es más grande que el nodo actual se intenta con el subárbol izquierdo, y si es más pequeño con el derecho.

#### ¿Para qué sirven los recorridos?¿Que tipos conoce?¿En qué se diferencian?¿Qué es un recorrido Acotado?

Los recorridos permiten desplazarse a través de todos los nodos del árbol de tal forma que cada nodo sea visitado una única vez.

Existen varios métodos, se diferencian en el orden en que se visitan:

En Inorden, Preorden y Postorden

Un recorrido de árbol acotado implica no recorrer nodos cuyo valor está fuera de un rango dado

#### ¿Qué escenarios pueden darse al momento de borrar un nodo de un árbol y como se resuelven?

**El nodo es una hoja**: Se puede borrar inmediatamente (actualizando direcciones)

**El nodo tiene un único hijo:** Puede ser borrado después de que el padre del nodo actualize el puntero al hijo del nodo a borrar

**El nodo tiene 2 hijos:** Se busca y selecciona el nodo más a la izquierda del subárbol derecho, se intercambia el valor del nodo encontrado por el que se quiere borrar, Se llama al borrar a partir del hijo derecho con el valor del nodo encontrado. El nodo ahora es una hoja y puede ser borrado inmediatamente.