Notación gran O()

* Permite evaluar que implementación es mejor
* Se usa para medir cuanto tiempo y cuanto espacio se utiliza.
* F es de orden g si
* Para todo n>=n0
* Se busca que aumente lo menos.
* Hay algunos como n! que incrementan demasiado rápido
* Calcular la complejidad
  + Tiempo
    - Resta de valores de filas y columnas
    - C1+c2+c3 =C el tiempo en hacer operación
    - NC+D= n veces la constante C más el tiempo de imprimir a la siguiente
    - N(Nc+d)=hace n veces lo anterior.
    - En tiempo O(n2)
  + Espacio
    - Se necesitan sólo 5 datos
    - Es constante O(5(1))

Ordenamiento

* Es el proceso de ordenar los elementos de una lista
* Está basado en valores específicos
* Hay muchos algoritmos que varían en eficiencia
* Dos diferentes:
  + Selection Sort
    - Seleccionoa el más pequeño y lo pone al principio
    - Cambia el que estaba en la primera a donde estaba el más pequeño
    - Se repite para todos los valores
    - 3 9 6 1 2
    - 1 9 6 3 2
    - 1 2 6 3 9
    - 1 2 3 6 9
    - Complejidad tiempo:
      * Se tarda n en encontrar el más pequeño
      * Hace lo anterior n veces
      * .: la complejidad O(n^2)
    - Espacio:
      * N para almacenar la cadena
      * .: la complejidad es O(n)
  + Insection Sort
    - Se selecciona uno y se pone donde corresponde
    - Si es necesario corre a los demás
    - 3 9 6 1 2
    - 3 9 6 1 2
    - 3 6 9 1 2
    - 1 3 6 9 2
    - 1 2 3 6 9
    - Complejidad en tiempo:
      * Para insertar algo en el peor de los casos se necesita n
      * Lo anterior se hace n veces
      * .: O(n^2)
    - Espacio
      * O(n)

Polimorfismo en ordenamiento

* Una clase que implementa la interfaz Comparable define un método compareTo
* compareTo compara dos objetos.
  + Retorna + si es mayor, 0 si es igual, - si es menor
* Se puede usar el polimosrfismo para hacer un sort genérico
* Es punto es usar objetos que usen Comparable