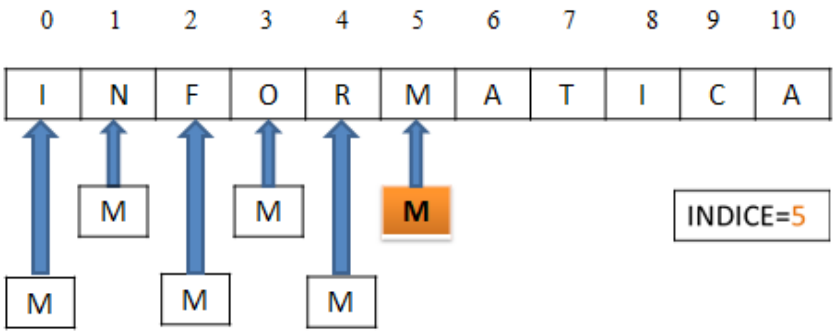
Annex I.- Arrays cerca i ordenació

Mètodes de cerca

1. Cerca lineal en un array no ordenat

Buscar, de forma seqüencial, un element en un array fins que el trobem. Si l'element no es troba, s'ha d'indicar d'alguna forma.

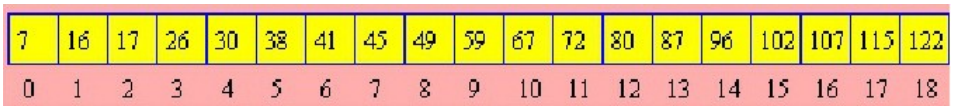


La seua complexitat temporal és de:

- En el millor dels casos: 1 (l'element es troba al principi).
- En el pitjor dels casos: n (l'element no està al vector).

2. Cerca lineal en un array ordenat

La diferència respecte a buscar en un array no ordenat és que el procés es pot parar abans (no sempre). Si busquem un valor en un array ordenat ascendentment, en el moment que trobem un valor major que l'element que estem buscant, sabem que l'element que busquem ja no estarà a partir d'eixa posició.



Si busquem l'element 35 només necessitarem fer 6 comparacions (fins la posició 5) ja que conté el número 38 i per tant els valors que queden són majors.

La seua complexitat temporal és de:

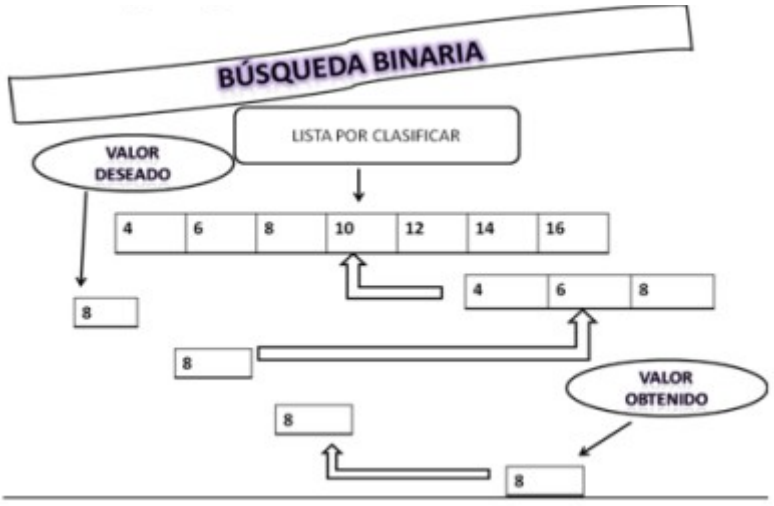
- En el millor dels casos: 1 (l'element es troba al principi).
- En el pitjor dels casos: n (l'element no està al vector).

3. Cerca binària o dicotòmica en un array ordenat

Consisteix en utilitzar la tècnica "Divide y vencerás". Este mètode consisteix en calcular la posició central de l'array i repetir el procés de la següent manera:

- Si l'element buscat és igual al contingut de la posició central: s'ha trobat.
- Si l'element buscat és menor que el contingut de la posició central: es repetirà el procés en la part esquerre.

- Si l'element buscat és major que el contingut de la posició central: es repetirà el procés en la part dreta.



Si busquem el número 8 només s'han de fer 3 comparacions.

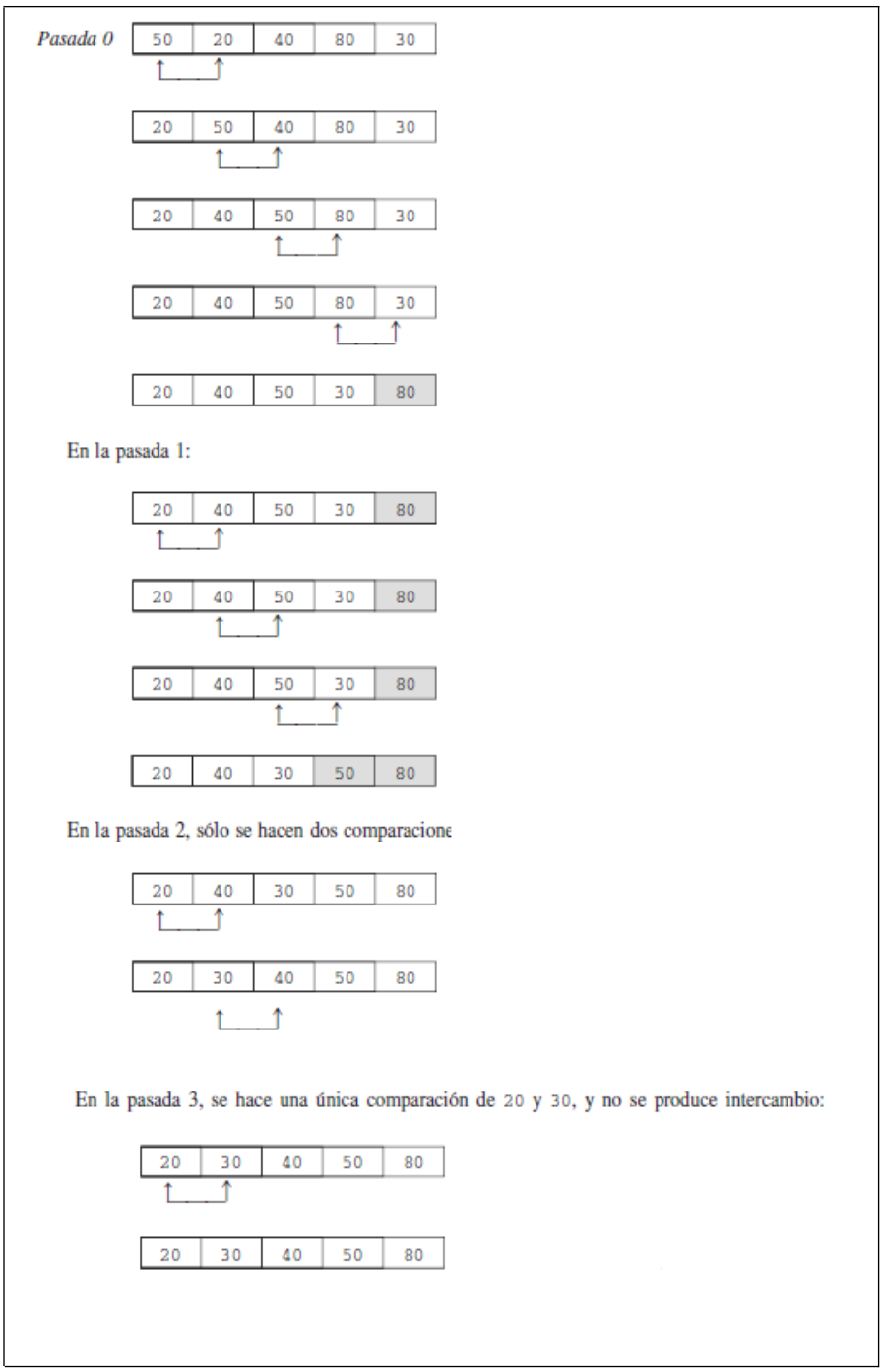
La seua complexitat temporal és de:

- En el millor dels casos: 1 (l'element es troba a la posició central).
- En el pitjor dels casos: log n (l'element no està al vector).

Mètodes d'ordenació

1. Mètode de la bambolla

Es tracta de comparar el primer element amb el segon i si el segon és menor que el primer, s'intercanvien els valors. Després el segon amb el tercer i així succesivament. Si no n'hi han intercanvis, és que l'array està ordenat (això és una millor immediata: si en una passada del bucle no es produix cap intercanvi, està ordenat i es pot aturar el procés). Té una complexitat (cost temporal) de $\mathbf{n^2}$, és a dir, que augmenta de forma exponencial (potència de dos) quan major siga el nombre d'elements a ordenar. Se es fa la millora el seu cost temporal seria de **n**.



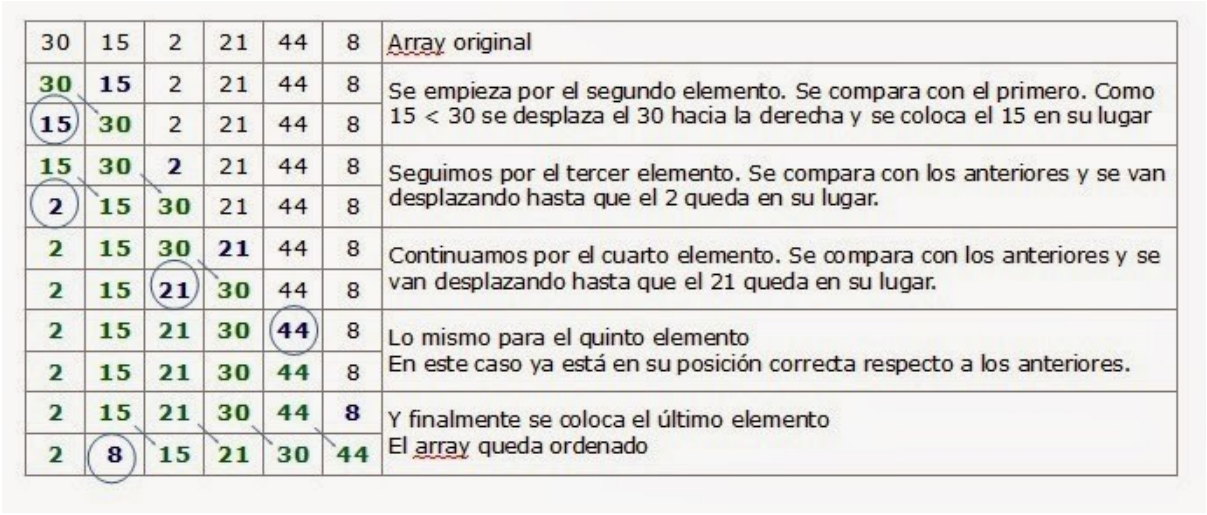
2. Inserció

Consisteix a recórrer tot l'array començant des del segon element fins al final. Per a cada element, es tracta de col·locar-lo en el lloc correcte entre tots els elements anteriors a ell, es a dir, entre els elements a la seva esquerra en l'array.

Donada una posició actual p, l'algorisme es basa en que els elements A[0], A[1], ..., A[p-1] ja estan ordenats.

Té una complexitat (cost temporal) de $\mathbf{n^2}$ (en el pitjor dels casos) i en el millor dels casos (quan l'array ja està ordenat) la complexitat és de **n**.

De forma gràfica el procés que segueix el mètode d'inserció directa és el següent:



30	15	2	21	44	8	Array original
30	15	2	21	44	8	Se empieza por el segundo elemento. Se compara con el primero. Como 15 < 30 se desplaza el 30 hacia la derecha y se coloca el 15 en su lugar
15	30	2	21	44	8	
15	30	2	21	44	8	Seguimos por el tercer elemento. Se compara con los anteriores y se van desplazando hasta que el 2 queda en su lugar.
2	15	30	21	44	8	
2	15	30	21	44	8	Continuamos por el cuarto elemento. Se compara con los anteriores y se van desplazando hasta que el 21 queda en su lugar.
2	15	21	30	44	8	
2	15	21	30	44	8	Lo mismo para el quinto elemento En este caso ya está en su posición correcta respecto a los anteriores.
2	15	21	30	44	8	
2	15	21	30	44	8	Y finalmente se coloca el último elemento El array queda ordenado
2	8	15	21	30	44	

3. Selecció

El mètode d'ordenació per selecció consisteix a repetir els següents passos:

1. Es busca l'element més xicotet de l'array i es col·loca en la primera posició.
2. Entre els restants, es busca l'element més xicotet i es col·loca en la segona posició.
3. Entre els restants es busca l'element més petit i es col·loca en la tercera posició.
4. ……..
5. Aquest procés es repeteix fins a col·locar l'últim element.

Té una complexitat (cost temporal) de $\mathbf{n^2}$ (en el pitjor dels casos) i en el millor dels casos (quan l'array ja està ordenat) la complexitat és de **n**.

De forma gràfica el procés seria el següent:

Array original a ordenar: [50, 26, 7, 9, 15, 27]

50	26	7	9	15	27	Array original
7	26	50	9	15	27	Se coloca el 7 en primera posición. Se intercambian 7 y 50
7	9	50	26	15	27	Se coloca el 9 en segunda posición. Se intercambian 9 y 26
7	9	15	26	50	27	Se coloca el 15 en tercera posición. Se intercambian 15 y 50
7	9	15	26	50	27	El menor de los que quedan es el 26. Se deja en su posición
7	9	15	26	27	50	Se coloca el 27 en quinta posición. Se intercambian 27 y 50

4. Quicksort

Consisteix en ordenar un array mitjançant un pivot, que és un punt intermedi en l'array, fent que a l'esquerre del pivot queden els menors i a la dreta els majors. S'ha de repetir el procés en troços de l'array. Utilitza recursivitat (per tant s'haurà de cridar a ell mateix). Té de mitjana una complexitat (cost temporal) de **n log n**, en el pitjor dels casos **n^2**.