Ordinamento di un vettore

Supponiamo di avere un vettore, ad es. di numeri interi, e di volerlo ordinare in ordine crescente, cioè dall'elemento più piccolo al più grande. Questo significa che, se per esempio il contenuto iniziale del vettore è:

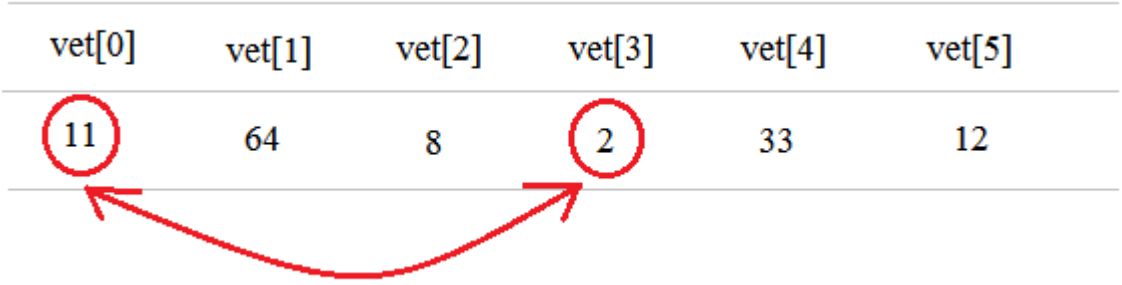
vet[0]	vet[1]	vet[2]	vet[3]	vet[4]	vet[5]
11	64	8	2	33	12

dopo l'ordinamento il nuovo contenuto sarà:

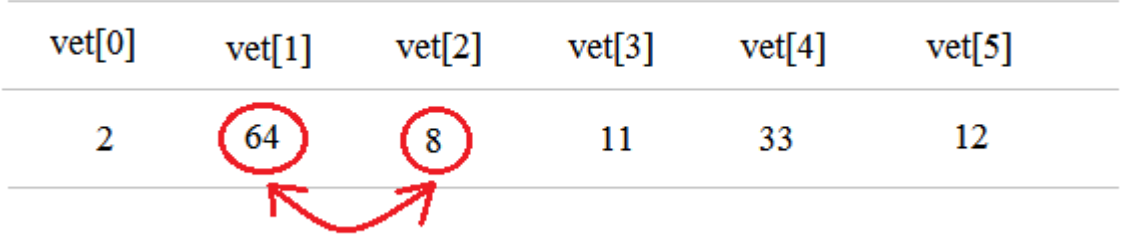
vet[0]	vet[1]	vet[2]	vet[3]	vet[4]	vet[5]
2	8	11	12	33	64

Selection Sort (chiamato anche metodo dello scambio)

L'idea è la seguente: si trova l'elemento minimo (quello col valore più piccolo di tutti) e lo si sposta nell'elemento zero (il primo elemento) del vettore, scambiandolo di posto con quest'ultimo:



A questo punto l'elemento zero è sistemato (contiene già il valore più piccolo). Si procede quindi a una nuova ricerca del minimo sugli elementi restanti (dall'1 alla fine del vettore). Analogamente a quanto fatto prima, il nuovo minimo viene scambiato di posto con l'elemento di posizione 1 (il secondo del vettore):



Il procedimento prosegue allo stesso modo fino alla fine del vettore.

Per implementare l'algoritmo occorrono due cicli annidati (uno dentro l'altro) così (*n* rappresenta il numero di elementi del vettore):

```
for(i=0; i<n-1; i++)
   {
   for(j=i+1; j<n; j++)
      {
       if(vet[j] < vet[i]) //cambiare questa condizione per invertire l'ordine
          {temp=vet[j];
           vet[j]=vet[i];
           vet[i]=temp;
          }
       }
    }
```

Si noti che il ciclo interno inizia da i+1 (cioè dall'elemento successivo all'ultimo elemento ordinato) e si limita a memorizzare la posizione dell'elemento minimo. Lo scambio viene fatto solo una volta che questa posizione è stata individuata (nel ciclo esterno).
Il tempo di esecuzione di questo algoritmo cresce ovviamente al crescere delle dimensioni del vettore da ordinare.

Cambiando la diseguaglianza indicata nel codice, l'ordinamento può essere facilmente invertito (dal più grande al più piccolo, decrescente).

controllo dei confronti // i rappresenta la prima casella, j la seconda

0-1x	1-2x	2-3x	3-4x	4-5x
0-2x	1-3x	2-4x	3-5x	(1if)
0-3x	1-4x	2-5x	(2if)	
0-4x	1-5x	(3if)		
0-5x	(4 if)			
(5 if)				

15 giri(confronti/if)

```
// i rappresenta la prima casella, j la seconda
for(i=0;i<5;i++)
   {for(j=i+1;j<6;j++)
     {if.....
     }
   }
```

il programmatore "distratto" fa eseguire 36 giri (confronti/if)

```
for(i=0;i<6;i++)6giri [0-1-2-3-4-5]
   {for(j=0;j<6;j++) 6giri [0-1-2-3-4-5]
```