Algoritmi definiti sugli array

Per utilizzare in maniera efficiente gli array, è necessario poter eseguire alcune operazioni “tipiche”:

- inizializzazione
- lettura
- stampa
- ricerca del minimo e del massimo
- ricerca di un valore
- eliminazione di un valore
- inserimento di un valore
- ordinamento del vettore

```
PROGRAM CercaMaxReal
  ! Lettura di un array di valori reali
  ! e ricerca del valore massimo.
  ! Il programma fornisce in uscita
  ! valore e posizione dell'elemento massimo.

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,posmax
  REAL :: vet, max

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  ! Ricerca del valore massimo
  posmax=1
  max=vet(1)

  DO i=2,riemp
    IF(vet(i)>max) THEN
      posmax=i
      max=vet(i)
    END IF

    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)
    WRITE(*,*)"Max corrente- pos: ",posmax,"  valore:
",max
  END DO
```

```
  ! Stampa risultati
  WRITE(*,*)"Valori letti: "
  DO i=1,riemp
    WRITE(*,*) vet(i)
  END DO
  WRITE(*,*)
   WRITE(*,*)"Il valore max e' ",max," e si trova in
posizione ",posmax

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
  WRITE(*,*)
  IF(riemp>0) THEN
    WRITE(*,*)"Il valore max e' ",max," e si trova in
posizione ",posmax
  ELSE
    WRITE(*,*)"Nessun valore fornito in input"
  END IF
```

```
Quanti elementi ?
5
 Valore dell'elemento  1 :
2.0
 Valore dell'elemento  2 :
1.3
 Valore dell'elemento  3 :
-1.8
 Valore dell'elemento  4 :
3.2
 Valore dell'elemento  5 :
2.9

 Elemento corrente vet( 2):  1.29999995
 Max corrente- pos:  1  valore:   2.

 Elemento corrente vet( 3): -1.79999995
 Max corrente- pos:  1  valore:   2.

 Elemento corrente vet( 4):  3.20000005
 Max corrente- pos:  4  valore:   3.20000005

 Elemento corrente vet( 5):  2.9000001
 Max corrente- pos:  4  valore:   3.20000005
 Valori letti:
  2.
  1.29999995
 -1.79999995
  3.20000005
  2.9000001

Il valore max e' 3.20000005 e si trova in posizione 4
 Premere ENTER per terminare
```

```
Quanti elementi ?
0
 Valori letti:

Il valore max e' 6.7021E-40 e si trova in posizione 1
 Premere ENTER per terminare



Quanti elementi ?
0
 Valori letti:

 Nessun valore fornito in input
 Premere ENTER per terminare
```

```
PROGRAM CercaMaxMinReal
  ! Lettura di un array di valori reali
  ! e ricerca del valore massimo e del valore minimo.
  ! Il programma fornisce in uscita
  ! valore e posizione dell'elemento massimo e
  ! dell'elemento minimo.

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,posmax,posmin
  REAL :: vet,max,min

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  ! Ricerca del valore massimo e del valore minimo
  posmax=1
  max=vet(1)

  posmin=1
  min=vet(1)

  DO i=2,riemp
    IF(vet(i)>max) THEN
      posmax=i
      max=vet(i)
    ELSE IF(vet(i)<min) THEN
      posmin=i
      min=vet(i)
    END IF
```

```
    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)
    WRITE(*,*)"Max corrente- pos: ",posmax,"  valore:
",max
    WRITE(*,*)"Min corrente- pos: ",posmin,"  valore:
",min
  END DO

  ! Stampa risultati
  WRITE(*,*)"Valori letti: "
  DO i=1,riemp
    WRITE(*,*) vet(i)
  END DO
  WRITE(*,*)

  IF(riemp>0) THEN
    WRITE(*,*)"Il valore max e' ",max," e si trova in
posizione ",posmax
    WRITE(*,*)"Il valore min e' ",min," e si trova in
posizione ",posmin
  ELSE
    WRITE(*,*)"Nessun valore fornito in input"
  END IF

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
Quanti elementi ?
5
 Valore dell'elemento  1 :
2.0
 Valore dell'elemento  2 :
1.3
 Valore dell'elemento  3 :
-1.8
 Valore dell'elemento  4 :
3.2
 Valore dell'elemento  5 :
2.9

 Elemento corrente vet( 2):  1.29999995
 Max corrente- pos:  1  valore:   2.
 Min corrente- pos:  2  valore:   1.29999995

 Elemento corrente vet( 3): -1.79999995
 Max corrente- pos:  1  valore:   2.
 Min corrente- pos:  3  valore:  -1.79999995

 Elemento corrente vet( 4):  3.20000005
 Max corrente- pos:  4  valore:   3.20000005
 Min corrente- pos:  3  valore:  -1.79999995

 Elemento corrente vet( 5):  2.9000001
 Max corrente- pos:  4  valore:   3.20000005
 Min corrente- pos:  3  valore:  -1.79999995
 Valori letti:
  2.
  1.29999995
 -1.79999995
  3.20000005
  2.9000001

 Il  valore  max  e'      3.20000005  e  si  trova  in
posizione  4
 Il  valore  min  e'    -1.79999995  e  si  trova  in
posizione  3
 Premere ENTER per terminare
```

```
PROGRAM ContaOccorrenze
  ! Lettura di un array di valori reali
  ! e conteggio delle occorrenze di un valore

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,cont
  REAL :: vet,val

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  ! Lettura del valore da ricercare
  WRITE(*,*)"Valore da cercare:"
  READ(*,*) val

  ! Conteggio delle occorrenze del valore
  cont=0
  DO i=1,riemp
    IF(vet(i)==val) THEN
      cont=cont+1
    END IF

    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)
     WRITE(*,*)"Occorrenze di ",val," trovate finora:
",cont
  END DO
```

```
  ! Stampa risultati
  WRITE(*,*)"Valori letti: "
  DO i=1,riemp
    WRITE(*,*) vet(i)
  END DO

  WRITE(*,*)
  IF(riemp>0) THEN
    WRITE(*,*)"Il valore ",val," e' presente ",cont,"
volte"
  ELSE
    WRITE(*,*)"Nessun valore fornito in input"
  END IF

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
Quanti elementi ?
 6
 Valore dell'elemento  1 :
 1.0
 Valore dell'elemento  2 :
 2.5
 Valore dell'elemento  3 :
 3.2
 Valore dell'elemento  4 :
 2.5
 Valore dell'elemento  5 :
 2.5
 Valore dell'elemento  6 :
 1.7

 Valore da cercare:
 2.5
```

```
Elemento corrente vet( 1):  1.
Occorrenze di   2.5 trovate finora:  0

Elemento corrente vet( 2):  2.5
Occorrenze di   2.5 trovate finora:  1

Elemento corrente vet( 3):  3.20000005
Occorrenze di   2.5 trovate finora:  1

Elemento corrente vet( 4):  2.5
Occorrenze di   2.5 trovate finora:  2

Elemento corrente vet( 5):  2.5
Occorrenze di   2.5 trovate finora:  3

Elemento corrente vet( 6):  1.70000005
Occorrenze di   2.5 trovate finora:  3

Valori letti:
 1.
 2.5
 3.20000005
 2.5
 2.5
 1.70000005

Il valore   2.5 e' presente  3 volte
Premere ENTER per terminare
```

```
PROGRAM ContaOccorrenze2
  ! Lettura di un array di valori reali
  ! e conteggio delle occorrenze di un valore
  ! Il programma fornisce in uscita il numero di
  ! occorrenze di un valore e le sue posizioni

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,k,cont,pos
  REAL :: vet,val

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)
  DIMENSION pos(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

! Lettura del valore da cercare
  WRITE(*,*)"Valore da cercare:"
  READ(*,*) val
```

```
  ! Conteggio delle occorrenze del valore
  cont=0
  DO i=1,riemp
    IF(vet(i)==val) THEN
      cont=cont+1
      pos(cont)=i
    END IF

    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)
     WRITE(*,*)"Occorrenze di ",val," trovate finora:
",cont
    WRITE(*,*)"Posizioni: ",(pos(k),k=1,cont)
  END DO

  ! Stampa risultati
  WRITE(*,*)"Valori letti: "
  DO i=1,riemp
    WRITE(*,*) vet(i)
  END DO

  WRITE(*,*)
  IF(riemp>0) THEN
    WRITE(*,*)"Il valore ",val," e' presente ",cont,"
volte"
    WRITE(*,*)"Posizioni: ",(pos(i),i=1,cont)
  ELSE
    WRITE(*,*)"Nessun valore fornito in input"
  END IF

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
Quanti elementi ?
 6
 Valore dell'elemento  1 :
 1.0
 Valore dell'elemento  2 :
 2.5
 Valore dell'elemento  3 :
 3.2
 Valore dell'elemento  4 :
 2.5
 Valore dell'elemento  5 :
 2.5
 Valore dell'elemento  6 :
 1.7

 Valore da cercare:
 2.5
```

```
Elemento corrente vet( 1):  1.
 Occorrenze di   2.5 trovate finora:  0
 Posizioni:

 Elemento corrente vet( 2):  2.5
 Occorrenze di   2.5 trovate finora:  1
 Posizioni:  2

 Elemento corrente vet( 3):  3.20000005
 Occorrenze di   2.5 trovate finora:  1
 Posizioni:  2

 Elemento corrente vet( 4):  2.5
 Occorrenze di   2.5 trovate finora:  2
 Posizioni:  2 4

 Elemento corrente vet( 5):  2.5
 Occorrenze di   2.5 trovate finora:  3
 Posizioni:  2 4 5

 Elemento corrente vet( 6):  1.70000005
 Occorrenze di   2.5 trovate finora:  3
 Posizioni:  2 4 5
 Valori letti:
  1.
  2.5
  3.20000005
  2.5
  2.5
  1.70000005

 Il valore   2.5 e' presente  3 volte
 Posizioni:  2 4 5
 Premere ENTER per terminare
```

```
PROGRAM CercaValore1
  ! Lettura di un array di valori reali
  ! e ricerca di un valore nell'array.
  ! Il programma fornisce in uscita il risultato
  ! della ricerca e la posizione del valore
  ! (se presente)

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,pos
  REAL :: vet,val
  LOGICAL :: trovato

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  ! Lettura del valore da cercare
  WRITE(*,*)"Valore da cercare:"
  READ(*,*) val
```

```
  ! Ricerca del valore
  trovato=.FALSE.
  pos=0

  DO i=1,riemp
    IF(vet(i)==val) THEN
      trovato=.TRUE.
      pos=i
    END IF

    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)
    WRITE(*,*)"Trovato valore ",val," : ", trovato
    WRITE(*,*)"Posizione: ",pos

  END DO

  ! Stampa risultati
  WRITE(*,*)"Valori letti: "
  DO i=1,riemp
    WRITE(*,*) vet(i)
  END DO

  WRITE(*,*)

  IF(riemp>0) THEN
    IF(trovato) THEN
      WRITE(*,*)"Il  valore  ",val,"  e'  presente  in
posizione ",pos
    ELSE
      WRITE(*,*)"Il valore ",val," non e' presente"
    END IF
  ELSE
    WRITE(*,*)"Nessun valore fornito in input"
  END IF

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
Quanti elementi ?
 6
 Valore dell'elemento  1 :
 1.0
 Valore dell'elemento  2 :
 2.5
 Valore dell'elemento  3 :
 3.2
 Valore dell'elemento  4 :
 2.5
 Valore dell'elemento  5 :
 2.5
 Valore dell'elemento  6 :
 1.7

 Valore da cercare:
 2.5
```

```
Elemento corrente vet( 1):  1.
 Trovato valore   2.5 :  F
 Posizione:  0

 Elemento corrente vet( 2):  2.5
 Trovato valore   2.5 :  T
 Posizione:  2

 Elemento corrente vet( 3):  3.20000005
 Trovato valore   2.5 :  T
 Posizione:  2

 Elemento corrente vet( 4):  2.5
 Trovato valore   2.5 :  T
 Posizione:  4

 Elemento corrente vet( 5):  2.5
 Trovato valore   2.5 :  T
 Posizione:  5

 Elemento corrente vet( 6):  1.70000005
 Trovato valore   2.5 :  T
 Posizione:  5
 Valori letti:
  1.
  2.5
  3.20000005
  2.5
  2.5
  1.70000005

 Il valore   2.5 e' presente in posizione  5
 Premere ENTER per terminare
```

```
PROGRAM CercaValore2
  ! Lettura di un array di valori reali
  ! e ricerca di un valore nell'array.
  ! Il programma fornisce in uscita il risultato
  ! della ricerca e la posizione del valore
  ! (se presente)

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,pos
  REAL :: vet,val
  LOGICAL :: trovato

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  ! Lettura del valore da cercare
  WRITE(*,*)"Valore da cercare:"
  READ(*,*) val
```

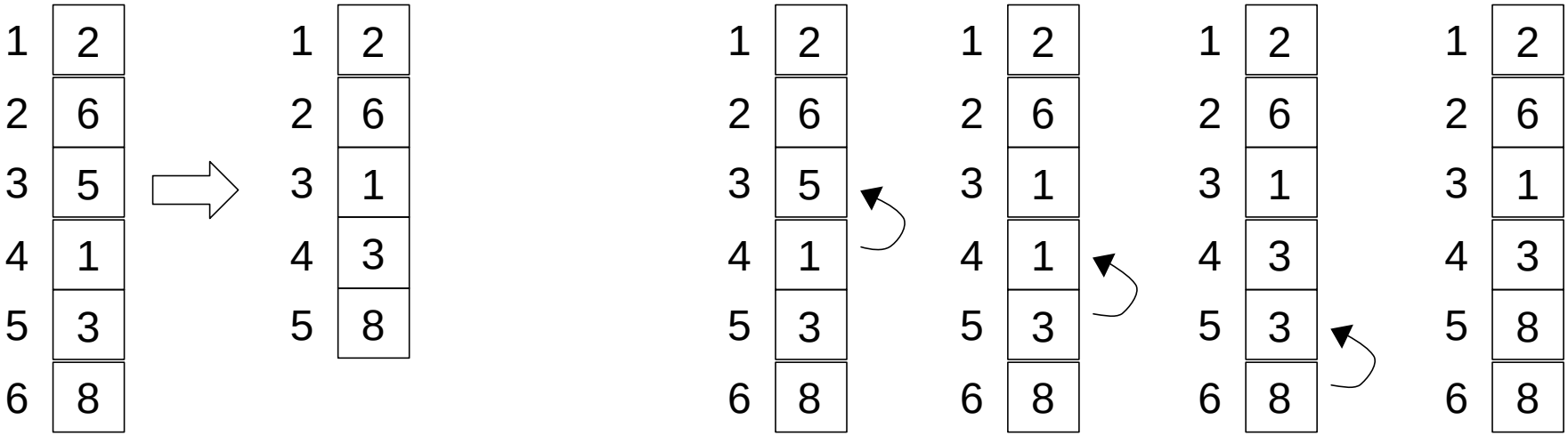
```
  ! Ricerca del valore
  trovato=.FALSE.
  pos=0
  i=1

  DO WHILE((i<=riemp).AND.(.NOT. trovato))
    IF(vet(i)==val) THEN
      trovato=.TRUE.
      pos=i
    END IF
    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)
    WRITE(*,*)"Trovato valore ",val," : ", trovato
    WRITE(*,*)"Posizione: ",pos

    ! Aggiorna variabile di ciclo
    i=i+1
END DO

  ! Stampa risultati
  WRITE(*,*)"Valori letti: "
  DO i=1,riemp
    WRITE(*,*) vet(i)
  END DO

  IF(riemp>0) THEN
    IF(trovato) THEN
      WRITE(*,*)"Il  valore  ",val,"  e'  presente  in
posizione ",pos
    ELSE
      WRITE(*,*)"Il valore ",val," non e' presente"
    END IF
  ELSE
    WRITE(*,*)"Nessun valore fornito in input"
  END IF

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
Quanti elementi ?
6
 Valore dell'elemento  1 :
1.0
 Valore dell'elemento  2 :
2.5
 Valore dell'elemento  3 :
3.7
 Valore dell'elemento  4 :
1.6
 Valore dell'elemento  5 :
2.3
 Valore dell'elemento  6 :
7.4
 Valore da cercare:
1.6

Elemento corrente vet( 1):  1.
Trovato valore   1.60000002 :  F
 Posizione:  0

 Elemento corrente vet( 2):  2.5
 Trovato valore   1.60000002 :  F
 Posizione:  0

 Elemento corrente vet( 3):  3.70000005
 Trovato valore   1.60000002 :  F
 Posizione:  0

 Elemento corrente vet( 4):  1.60000002
 Trovato valore   1.60000002 :  T
 Posizione:  4
 Valori letti:
  1.
  2.5
  3.70000005
  1.60000002
  2.29999995
  7.4000001
 Il valore   1.60000002 e' presente in posizione  4
 Premere ENTER per terminare
```

Eliminazione di un valore da un array

Nel caso si debba eliminare un certo valore da un array, i passi da compiere sono:

- ricerca nell'array per identificare la posizione del valore (se presente)
- cancellazione logica tramite scorrimento di una posizione verso l'alto dei successivi elementi dell'array



```
PROGRAM EliminaValore
  ! Lettura di un array di valori reali
  ! ed eliminazione di un valore dall'array.
  ! Il programma fornisce in uscita il risultato
  ! della ricerca e l'array aggiornato

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,pos,k
  REAL :: vet,val
  LOGICAL :: trovato

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  ! Lettura del valore da cercare
  WRITE(*,*)"Valore da cercare:"
  READ(*,*) val

  ! Ricerca del valore
  trovato=.FALSE.
  pos=0
  i=1

  DO WHILE((i<=riemp).AND.(.NOT. trovato))
    IF(vet(i)==val) THEN
      trovato=.TRUE.
      pos=i
    END IF

    ! Aggiorna variabile di ciclo
    i=i+1
  END DO
```

```
  IF(trovato) THEN
    ! Eliminazione

    ! Stampa di controllo
    WRITE(*,*)"Valore",val,"trovato in posizione",pos

    ! 1. Scorrimento degli elementi
    DO i=pos,riemp-1
      vet(i)=vet(i+1)
      ! Stampa di controllo
      WRITE(*,*)"Copia di vet(",i+1,") in vet(",i,")"
      WRITE(*,*) (vet(k),k=1,riemp)

    END DO

    ! 2. Aggiornamento del riempimento
    riemp=riemp-1

    ! Stampa dell'array modificato
    WRITE(*,*)"Array modificato: "
    DO i=1,riemp
      WRITE(*,*) vet(i)
    END DO

  ELSE
    WRITE(*,*)"Valore     ",val,"     non     presente
nell'array"
  END IF

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

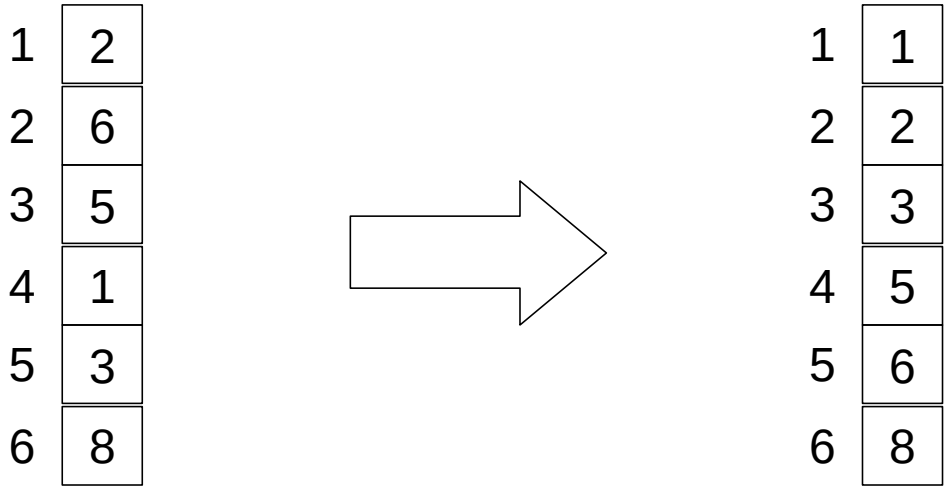
```
Quanti elementi ?
6
 Valore dell'elemento  1 :
1.0
 Valore dell'elemento  2 :
2.7
 Valore dell'elemento  3 :
9.2
 Valore dell'elemento  4 :
1.5
 Valore dell'elemento  5 :
4.6
 Valore dell'elemento  6 :
7.5
 Valore da cercare:
9.2

 Valore   9.19999981 trovato in posizione  3

 Copia di vet( 4) in vet( 3)
  1.0  2.70000005  1.5  1.5  4.5999999  7.5

 Copia di vet( 5) in vet( 4)
  1.0  2.70000005  1.5  4.5999999  4.5999999  7.5

 Copia di vet( 6) in vet( 5)
  1.0  2.70000005  1.5  4.5999999  7.5  7.5

 Array modificato:
  1.0
  2.70000005
  1.5
  4.5999999
  7.5
 Premere ENTER per terminare
```

Ordinamento di un array

L'operazione di ordinamento consiste in una permutazione degli elementi nell'array in modo che, al termine dell'ordinamento, la disposizione degli elementi nell'array rispetti un ordine specifico (p.es. crescente).

Lo scopo dell'ordinamento è di facilitare successive ricerche di elementi nell'array che è stato ordinato.

Sono stati proposti numerosi algoritmi di ordinamento, con caratteristiche diverse. Non esiste l'algoritmo di ordinamento ottimo.

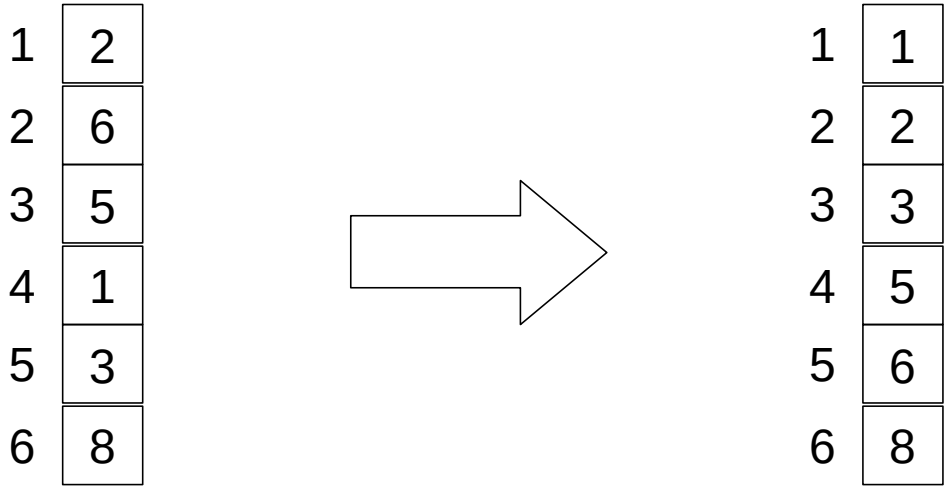


Ordinamento di un array

L'operazione di ordinamento consiste in una permutazione degli elementi nell'array in modo che, al termine dell'ordinamento, la disposizione degli elementi nell'array rispetti un ordine specifico (p.es. crescente).

Lo scopo dell'ordinamento è di facilitare successive ricerche di elementi nell'array che è stato ordinato.

Sono stati proposti numerosi algoritmi di ordinamento, con caratteristiche diverse. Non esiste l'algoritmo di ordinamento ottimo.



Ordinamento per selezione (select sort)

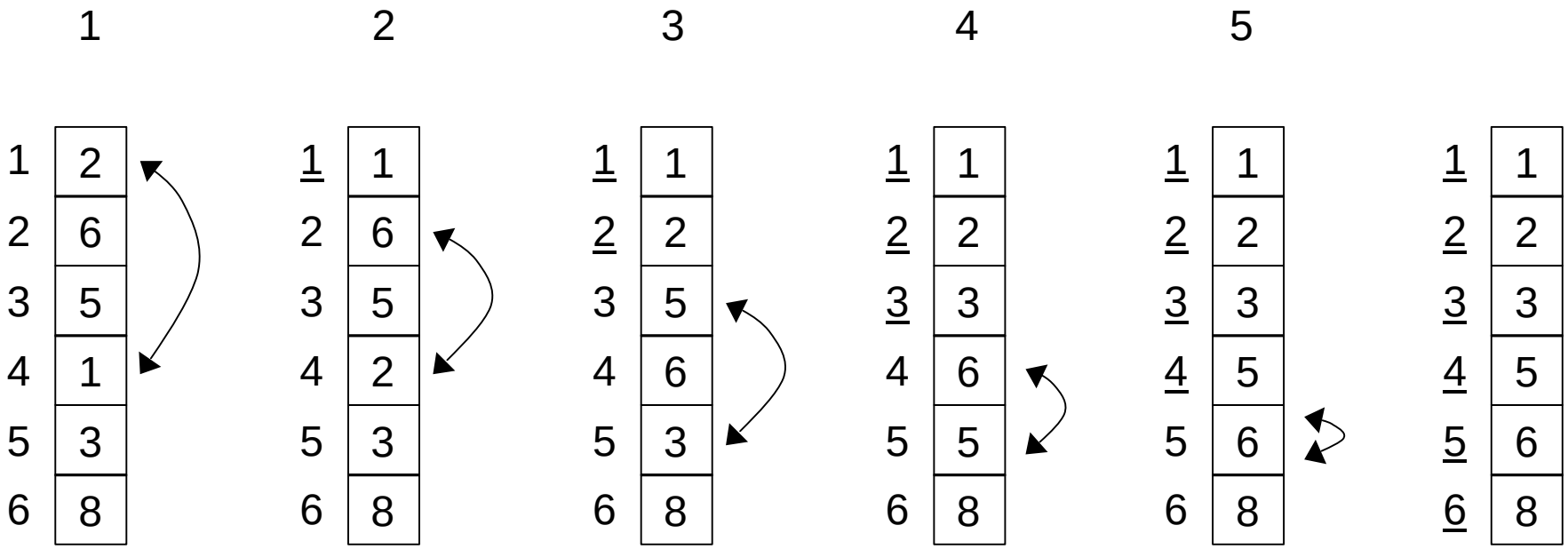
Questo algoritmo si basa sul seguente principio:

dato un array vet di N elementi, si determina l'elemento minimo tra vet(1), vet(2), … , vet(N) e lo si scambia con il primo elemento

Queste operazioni vengono poi ripetute su N-1 elementi a partire da vet(2), poi su N-2 elementi a partire da vet(3), …, su 2 elementi a partire da vet(N-1).

```
DO i=1,N-1
    determina l'elemento minimo in vet(i),…,vet(N)
    e scrivi in k la sua posizione

    scambia vet(i) e vet(k)

END DO
```

Ordinamento per selezione (select sort)



2 **Parte ordinata**

3 **Parte disordinata**

```
PROGRAM OrdinaSelect
  ! Lettura di un array di valori interi
  ! ed ordinamento degli elementi in
..! ordine crescente.
  ! Il programma fornisce in uscita l'array ordinato

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,j,k,posmin
  INTEGER :: vet,min,appo

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  ! Ciclo esterno
  DO i=1,riemp-1
    ! Stampa di controllo
    WRITE(*,*)"Ciclo esterno - indice: ",i
    posmin=i
    min=vet(i)

    ! Ciclo interno
    ! Ricerca del minimo tra i e riemp
    DO j=i+1,riemp
      IF(vet(j)<min) THEN
        min=vet(j)
        posmin=j
      END IF
    END DO

    ! Stampa di controllo
    WRITE(*,*)"Minimo trovato in pos. ",posmin
```

```
    ! Aggiornamento dell'elemento i
    IF(posmin/=i) THEN
      ! Stampa di controllo
      WRITE(*,*)"Scambio  dell'elemento  ",i,"  con
l'elemento ",posmin
      appo=vet(i)
      vet(i)=vet(posmin)
      vet(posmin)=appo
    END IF

    ! Stampa di controllo
    WRITE(*,*)"Array al termine del passo ",i," :"
    WRITE(*,*) (vet(k),k=1,riemp)
    WRITE(*,*)
  END DO ! Fine ciclo esterno

  ! Stampa dell'array ordinato
  WRITE(*,*)"Array ordinato: "
  DO i=1,riemp
    WRITE(*,*) vet(i)
  END DO

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
Quanti elementi ?
6
 Valore dell'elemento  1 :
5
 Valore dell'elemento  2 :
1
 Valore dell'elemento  3 :
6
 Valore dell'elemento  4 :
2
 Valore dell'elemento  5 :
4
 Valore dell'elemento  6 :
3

Ciclo esterno - indice:  1
 Minimo trovato in pos.  2
 Scambio dell'elemento  1 con l'elemento  2
 Array al termine del passo  1 :
 1 5 6 2 4 3

 Ciclo esterno - indice:  2
 Minimo trovato in pos.  4
 Scambio dell'elemento  2 con l'elemento  4
 Array al termine del passo  2 :
 1 2 6 5 4 3

 Ciclo esterno - indice:  3
 Minimo trovato in pos.  6
 Scambio dell'elemento  3 con l'elemento  6
 Array al termine del passo  3 :
 1 2 3 5 4 6

 Ciclo esterno - indice:  4
 Minimo trovato in pos.  5
 Scambio dell'elemento  4 con l'elemento  5
 Array al termine del passo  4 :
 1 2 3 4 5 6

 Ciclo esterno - indice:  5
 Minimo trovato in pos.  5
 Array al termine del passo  5 :
 1 2 3 4 5 6
```

```
Array ordinato:
1
2
3
4
5
6
Premere ENTER per terminare
```

```
PROGRAM CercaValore3
  ! Lettura di un array ordinato di valori interi
  ! e ricerca di un valore nell'array. Il programma
  ! fornisce in uscita il risultato della ricerca
  ! e la posizione del valore (se presente)

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,pos
  INTEGER :: vet,val
  LOGICAL :: trovato,finito

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

  WRITE(*,*)
  ! Lettura del valore da cercare
  WRITE(*,*)"Valore da cercare:"
  READ(*,*) val

  ! Ricerca del valore
  trovato=.FALSE.
  finito=.FALSE.
  pos=0
  i=1
```

```
! Ricerca del valore
  trovato=.FALSE.
  finito=.FALSE.
  pos=0
  i=1

  DO WHILE((i<=riemp).AND.(.NOT. trovato).AND.
           (.NOT. finito))
    IF(vet(i)==val) THEN
      trovato=.TRUE.
      pos=i
    ELSE IF(vet(i)>val) THEN
      finito=.TRUE.
    END IF

    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)
    WRITE(*,*)"Trovato valore ",val," : ", trovato
    WRITE(*,*)"Posizione: ",pos
    WRITE(*,*)"Finito: ",finito

    ! Aggiorna variabile di ciclo
    i=i+1

  END DO

IF(riemp>0) THEN
    IF(trovato) THEN
      WRITE(*,*)"Il  valore  ",val,"  e'  presente  in
posizione ",pos
    ELSE
      WRITE(*,*)"Il valore ",val," non e' presente"
    END IF
  ELSE
    WRITE(*,*)"Nessun valore fornito in input"
  END IF

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
 Quanti elementi ?
6
 Valore dell'elemento  1 :
2
 Valore dell'elemento  2 :
4
 Valore dell'elemento  3 :
5
 Valore dell'elemento  4 :
7
 Valore dell'elemento  5 :
10
 Valore dell'elemento  6 :
13

 Valore da cercare:
6

 Elemento corrente vet( 1): 2
 Trovato valore  6 :  F
 Posizione:  0
 Finito:  F

 Elemento corrente vet( 2): 4
 Trovato valore  6 :  F
 Posizione:  0
 Finito:  F

 Elemento corrente vet( 3): 5
 Trovato valore  6 :  F
 Posizione:  0
 Finito:  F

 Elemento corrente vet( 4): 7
 Trovato valore  6 :  F
 Posizione:  0
 Finito:  T

 Il valore  6 non e' presente
 Premere ENTER per terminare
```

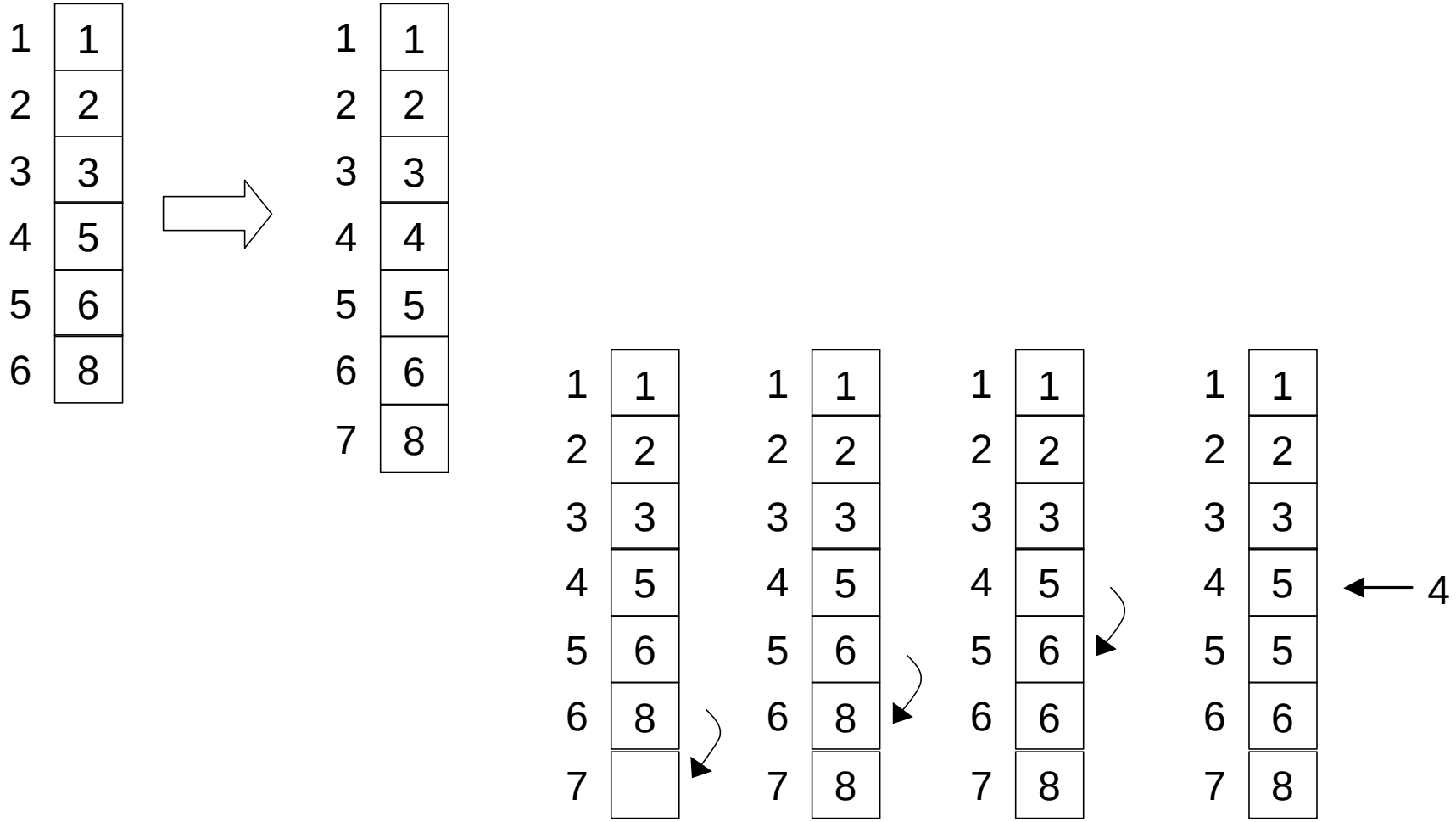
Inserimento di un valore in un array ordinato

L'inserimento di un valore in un array ordinato deve mantenere l'ordine degli elementi. Per questo motivo, non è possibile il semplice accodamento del nuovo valore a quelli già presenti, ma è necessario modificare la disposizione degli elementi già presenti per fare in modo che il nuovo valore possa essere inserito nella giusta posizione.

L'inserimento si realizza in due passi:

- individuare la posizione che deve assumere il nuovo valore all'interno dell'array per mantenere l'ordine dell'insieme dei valori
- rendere disponibile quella posizione facendo scorrere tutti gli elementi da quella posizione in poi di un posto in basso e infine scrivere il nuovo valore nella posizione resa disponibile

Inserimento di un valore in un array ordinato



```
PROGRAM InsertOrdinato
  ! Lettura di un array ordinato di valori interi
  ! ed inserimento nell'array di un nuovo valore.
  ! Il programma produce in uscita l'array modificato

  IMPLICIT NONE
  INTEGER :: maxnum,riemp,i,j,k
  INTEGER :: vet,val
  LOGICAL :: finito

  PARAMETER(maxnum=10)
  DIMENSION vet(maxnum)

  WRITE(*,*)"Quanti elementi ?"
  READ(*,*) riemp

  ! Lettura degli elementi dell'array
  DO i=1,riemp
    WRITE(*,*)"Valore dell'elemento ",i," :"
    READ(*,*) vet(i)
  END DO

! Lettura del valore da inserire
  WRITE(*,*)"Valore da inserire:"
  READ(*,*) val

  ! Ricerca del valore
  finito=.FALSE.
  i=1

  DO WHILE((i<=riemp).AND.(.NOT. finito))
    ! Stampa di controllo.
    WRITE(*,*)
    WRITE(*,*)"Elemento corrente vet(",i,"):",vet(i)

    IF(vet(i)>=val) THEN
      finito=.TRUE.
    ELSE
      i=i+1
    END IF

    ! Stampa di controllo.
    WRITE(*,*)"Finito: ",finito
  END DO
```

```
  ! Inserimento

  ! 1. Scorrimento degli elementi
  DO j=riemp,i,-1
    vet(j+1)=vet(j)
    ! Stampa di controllo
    WRITE(*,*)
    WRITE(*,*) "Copia di vet(",j,") in vet(",j+1,")"
    WRITE(*,*) "Array: ",(vet(k),k=1,riemp+1)
  END DO

  ! 2. Scrittura del valore
  vet(i)=val

  ! 3. Aggiornamento del riempimento
  riemp=riemp+1

  WRITE(*,*)

  WRITE(*,*) "Array modificato: "
  WRITE(*,*) (vet(i),i=1,riemp)

  WRITE(*,*) "Premere ENTER per terminare"
  READ(*,*)

END PROGRAM
```

```
Quanti elementi ?
6
 Valore dell'elemento  1 :
2
 Valore dell'elemento  2 :
4
 Valore dell'elemento  3 :
5
 Valore dell'elemento  4 :
7
 Valore dell'elemento  5 :
10
 Valore dell'elemento  6 :
13

Valore da inserire:
6

 Elemento corrente vet( 1): 2
 Finito:  F

 Elemento corrente vet( 2): 4
 Finito:  F

 Elemento corrente vet( 3): 5
 Finito:  F

 Elemento corrente vet( 4): 7
 Finito:  T

 Copia di vet( 6) in vet( 7)
 Array:  2 4 5 7 10 13 13

 Copia di vet( 5) in vet( 6)
 Array:  2 4 5 7 10 10 13

 Copia di vet( 4) in vet( 5)
 Array:  2 4 5 7 7 10 13

 Array modificato:
 2 4 5 6 7 10 13

 Premere ENTER per terminare
```

```
Quanti elementi ?
6
 Valore dell'elemento  1 :
2
 Valore dell'elemento  2 :
4
 Valore dell'elemento  3 :
5
 Valore dell'elemento  4 :
7
 Valore dell'elemento  5 :
10
 Valore dell'elemento  6 :
13

Valore da inserire:
1

 Elemento corrente vet( 1): 2
 Finito:  T

 Copia di vet( 6) in vet( 7)
 Array:  2 4 5 7 10 13 13

 Copia di vet( 5) in vet( 6)
 Array:  2 4 5 7 10 10 13

 Copia di vet( 4) in vet( 5)
 Array:  2 4 5 7 7 10 13

 Copia di vet( 3) in vet( 4)
 Array:  2 4 5 5 7 10 13

 Copia di vet( 2) in vet( 3)
 Array:  2 4 4 5 7 10 13

 Copia di vet( 1) in vet( 2)
 Array:  2 2 4 5 7 10 13

 Array modificato:
 1 2 4 5 7 10 13
 Premere ENTER per terminare
```

```
Quanti elementi ?
6
 Valore dell'elemento  1 :
2
 Valore dell'elemento  2 :
4
 Valore dell'elemento  3 :
5
 Valore dell'elemento  4 :
7
 Valore dell'elemento  5 :
10
 Valore dell'elemento  6 :
13

Valore da inserire:
15
Elemento corrente vet( 1): 2
 Finito:  F

 Elemento corrente vet( 2): 4
 Finito:  F

 Elemento corrente vet( 3): 5
 Finito:  F

 Elemento corrente vet( 4): 7
 Finito:  F

 Elemento corrente vet( 5): 10
 Finito:  F

 Elemento corrente vet( 6): 13
 Finito:  F

 Array modificato:
 2 4 5 7 10 13 15
 Premere ENTER per terminare
```

```
Quanti elementi ?
0

Valore da inserire:
6

Array modificato:
 6

 Premere ENTER per terminare
```