Titolo unità didattica: Ordinamento di array, parte I [11]

Titolo modulo : Algoritmo di ordinamento per inserimento [01-T]

Approccio incrementale all'ordinamento

Argomenti trattati:

- ✓ porzioni ordinate di array
- ✓ algoritmo di ordinamento per inserimento
- ✓ analisi dell'efficienza

Prerequisiti richiesti: P1-07-03-T

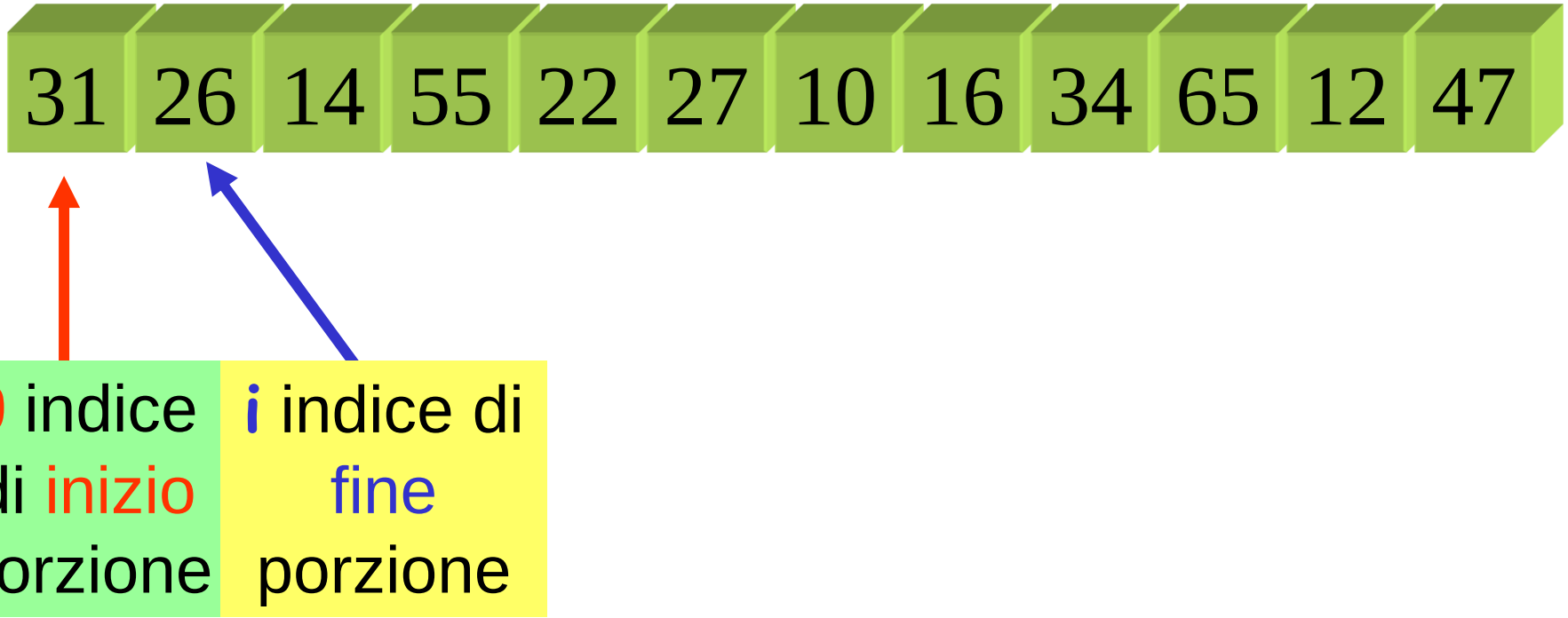
problema: (**sorting**)
ordinamento (senso crescente) dei valori di un array 1D

dati di input: l'array `a` (da ordinare), il size `n`

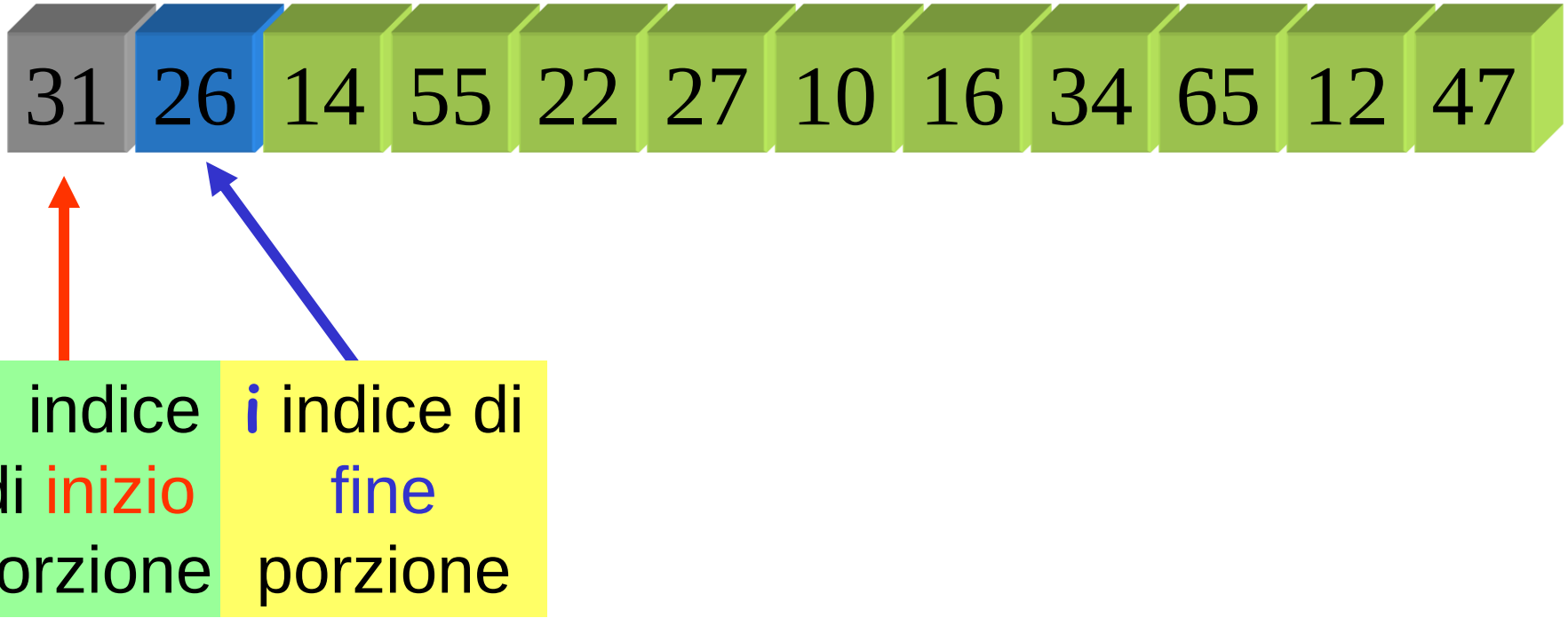
dato di output: l'array `a` (ordinato) (*in place*)

idea incrementale:

- ✓ ordinare i primi **2** elementi (porzione `0..1`);
- ✓ ordinare i primi **3** elementi (porzione `0..2`), inserendo il **terzo** elemento nella posizione corretta rispetto ai precedenti due elementi;
- ✓ ordinare i primi **4** elementi (porzione `0..3`), inserendo il **quarto** elemento nella posizione corretta rispetto ai precedenti tre elementi;
- ✓ e così via ….

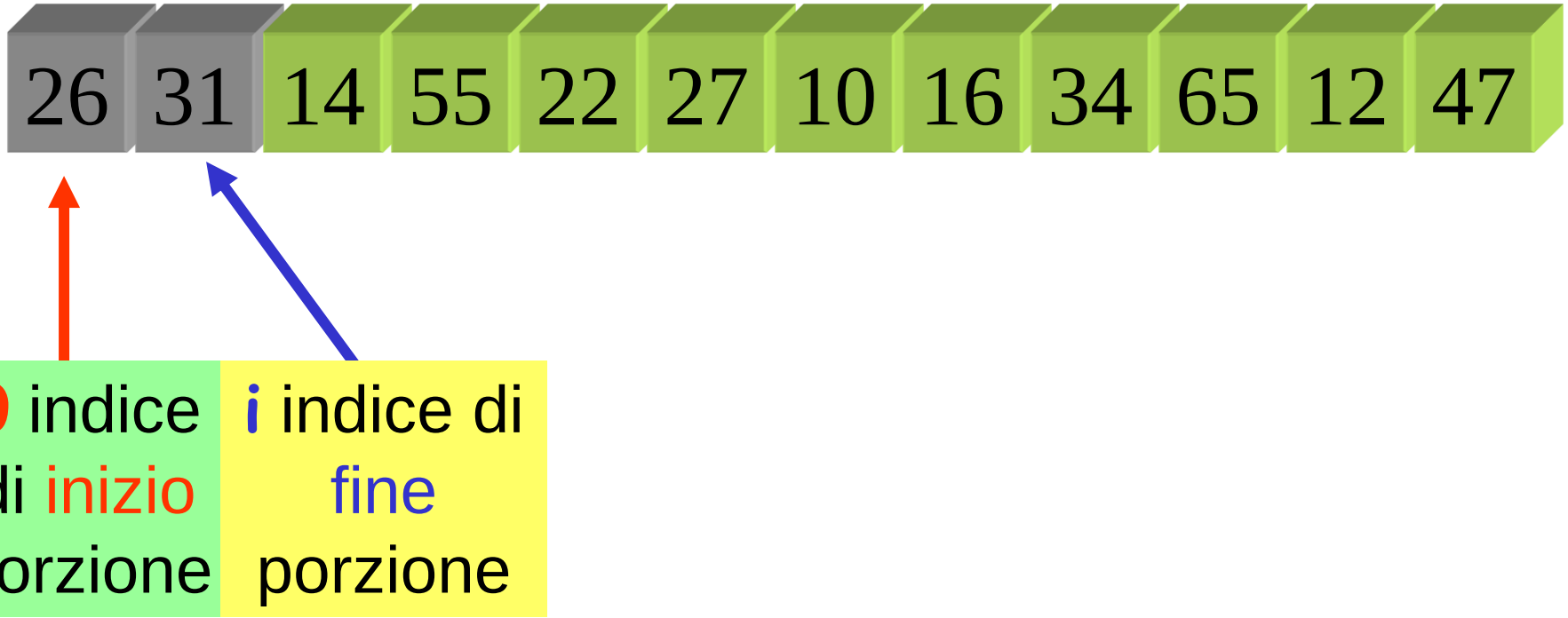



```
for (i=1; i<n; i++) {
  ordinare la porzione 0..i dell'array
 }
```




```
for (i=1; i<n; i++) {
    inserire l'i-simo elemento in posizione
    corretta nella porzione 0..i in modo
    che la porzione 0..i sia ordinata
}
```

i=1




```
for (i=1; i<n; i++) {
    inserire l'i-simo elemento in posizione
    corretta nella porzione 0..i in modo
    che la porzione 0..i sia ordinata
}
```

i=1

0 indice di inizio porzione

i indice di fine porzione

```
for (i=1; i<n; i++) {
    inserire l'i-simo elemento in posizione
    corretta nella porzione 0..i in modo
    che la porzione 0..i sia ordinata
}
```

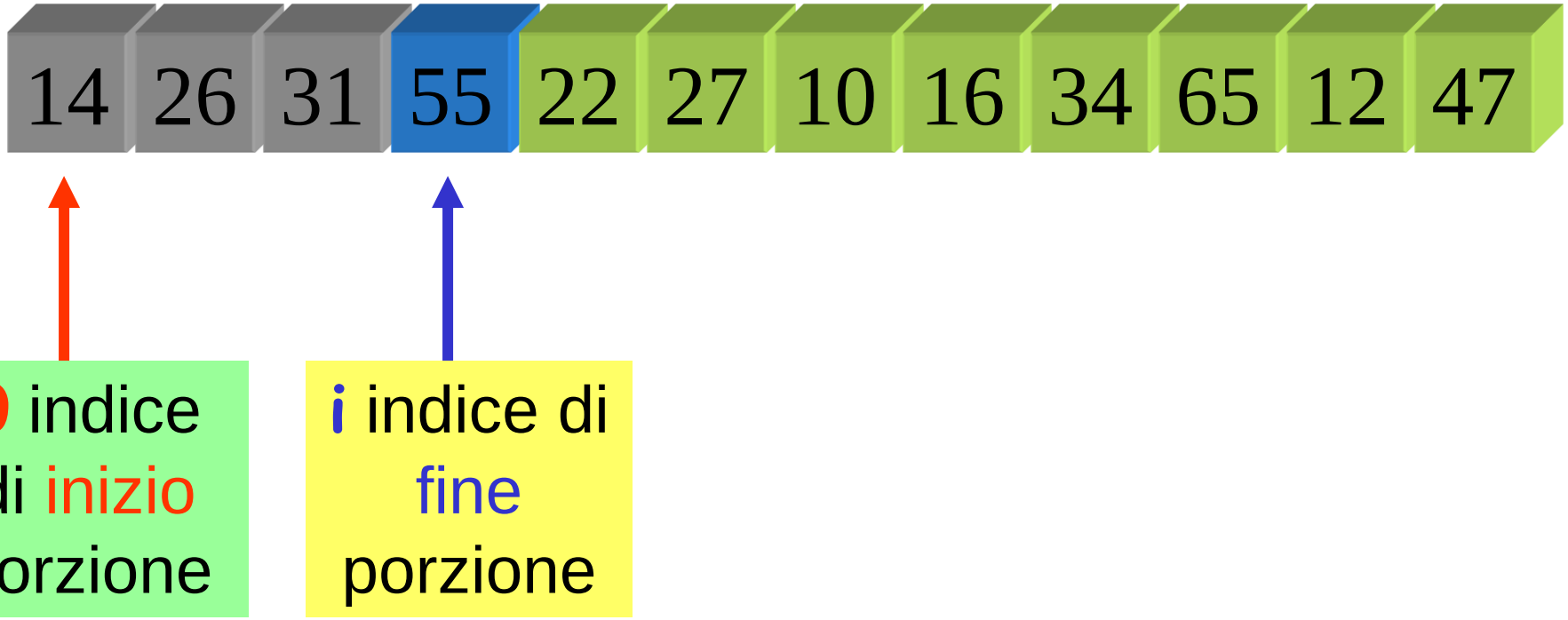
i=2

0 indice di inizio porzione

i indice di fine porzione

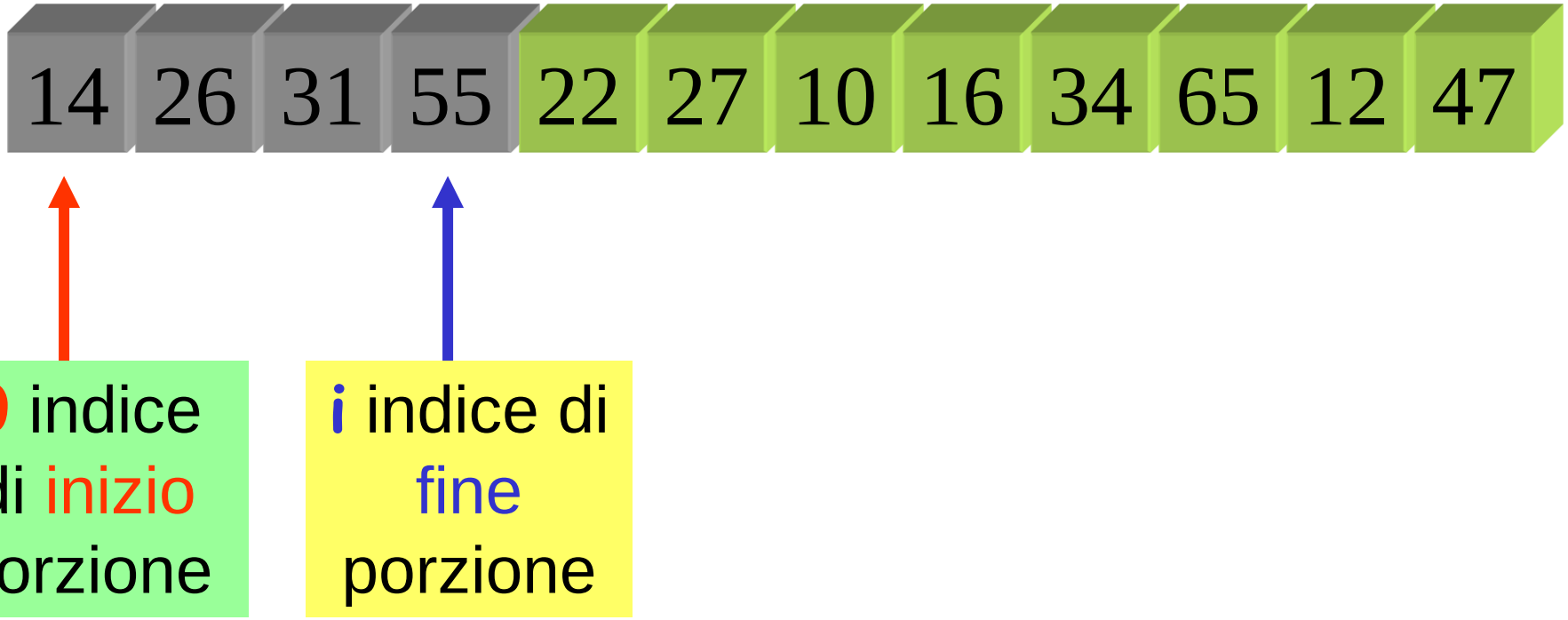
```
for (i=1; i<n; i++) {
    inserire l'i-simo elemento in posizione
    corretta nella porzione 0..i in modo
    che la porzione 0..i sia ordinata
}
```

i=2



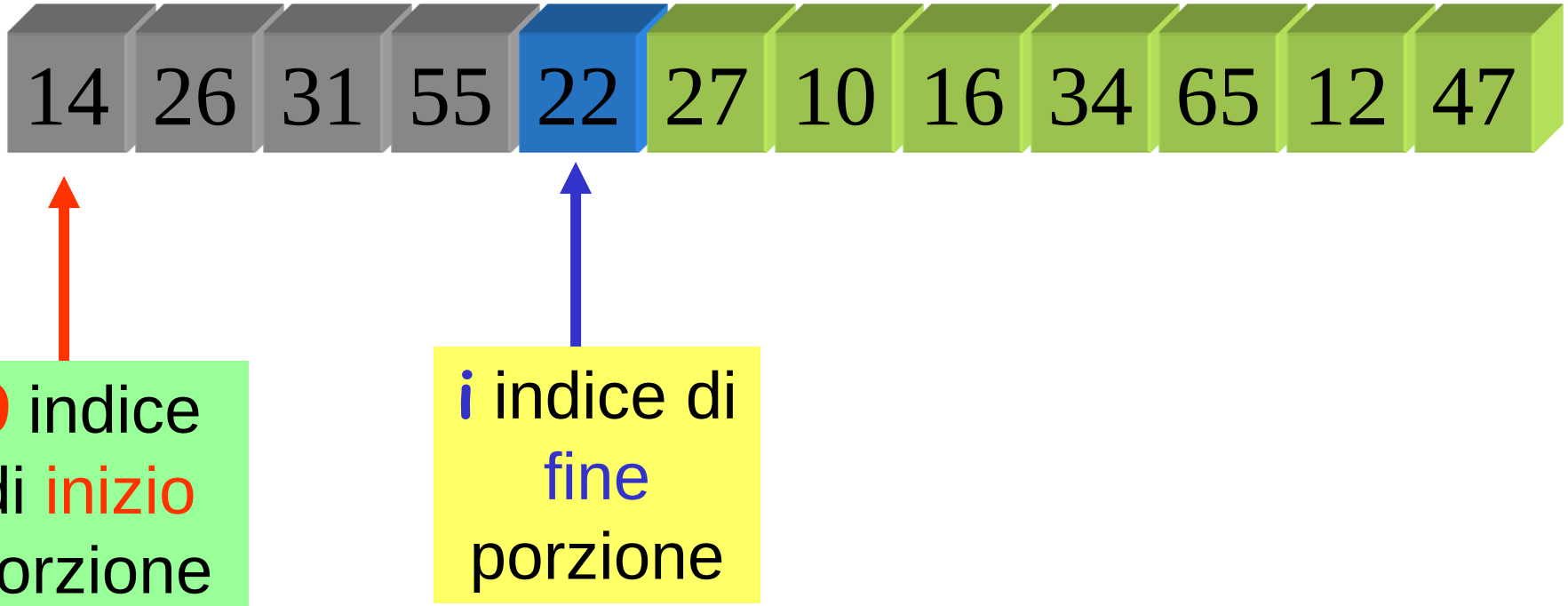

```
for (i=1; i<n; i++) {
    inserire l'i-simo elemento in posizione
    corretta nella porzione 0..i in modo
    che la porzione 0..i sia ordinata
}
```

i=3



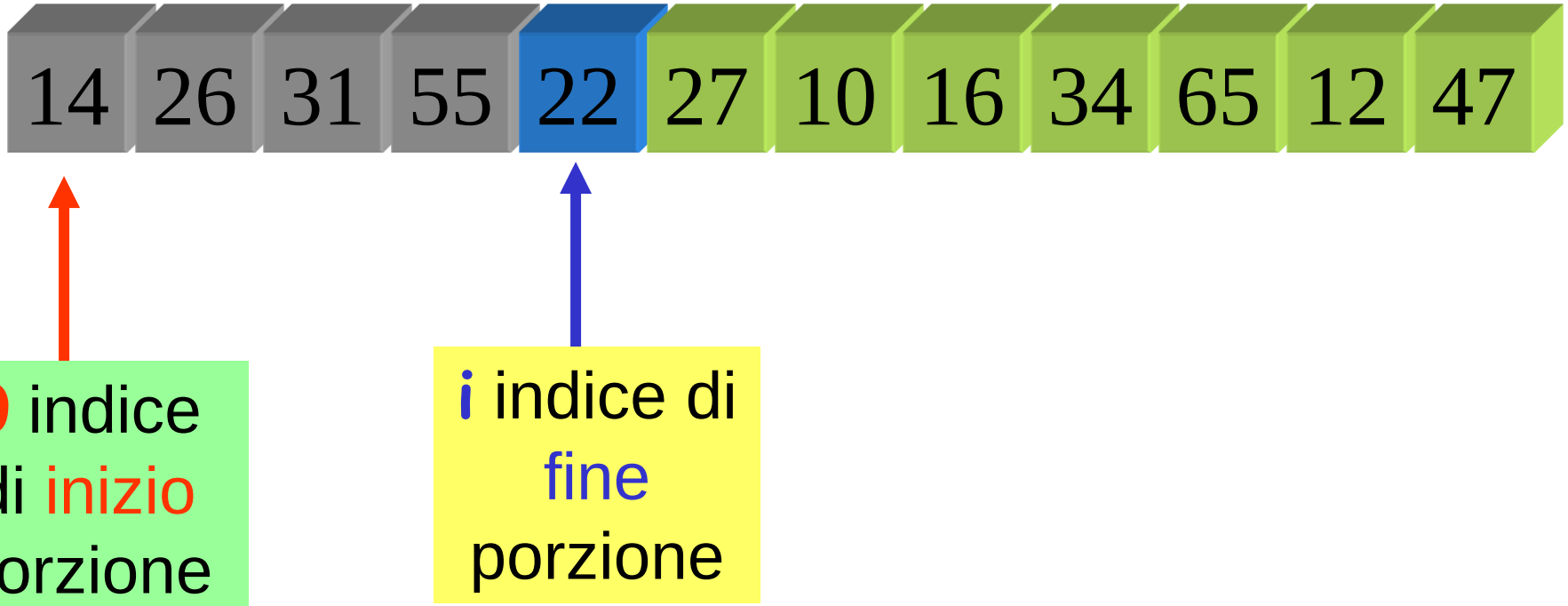

```
for (i=1; i<n; i++) {
    inserire l'i-simo elemento in posizione
    corretta nella porzione 0..i in modo
    che la porzione 0..i sia ordinata
}
```

i=3

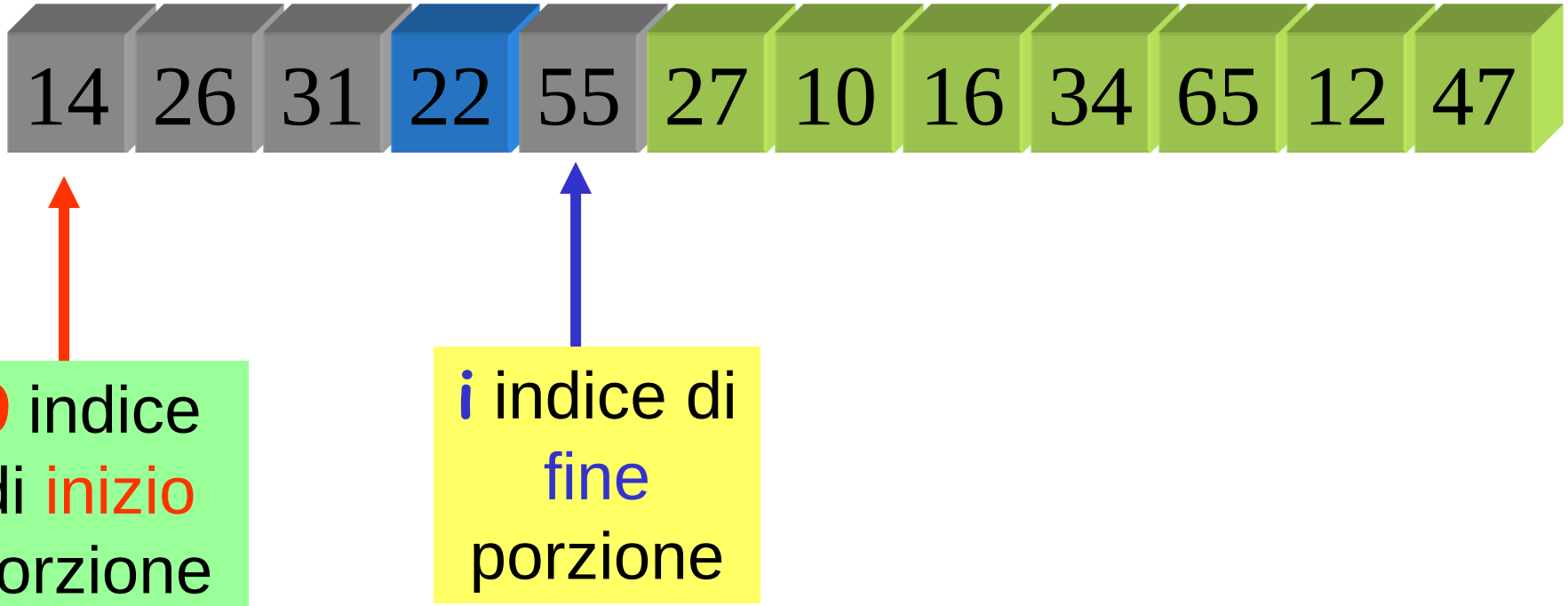



```
for (i=1; i<n; i++) {
    inserire l'i-simo elemento in posizione
    corretta nella porzione 0..i in modo
    che la porzione 0..i sia ordinata
}
```

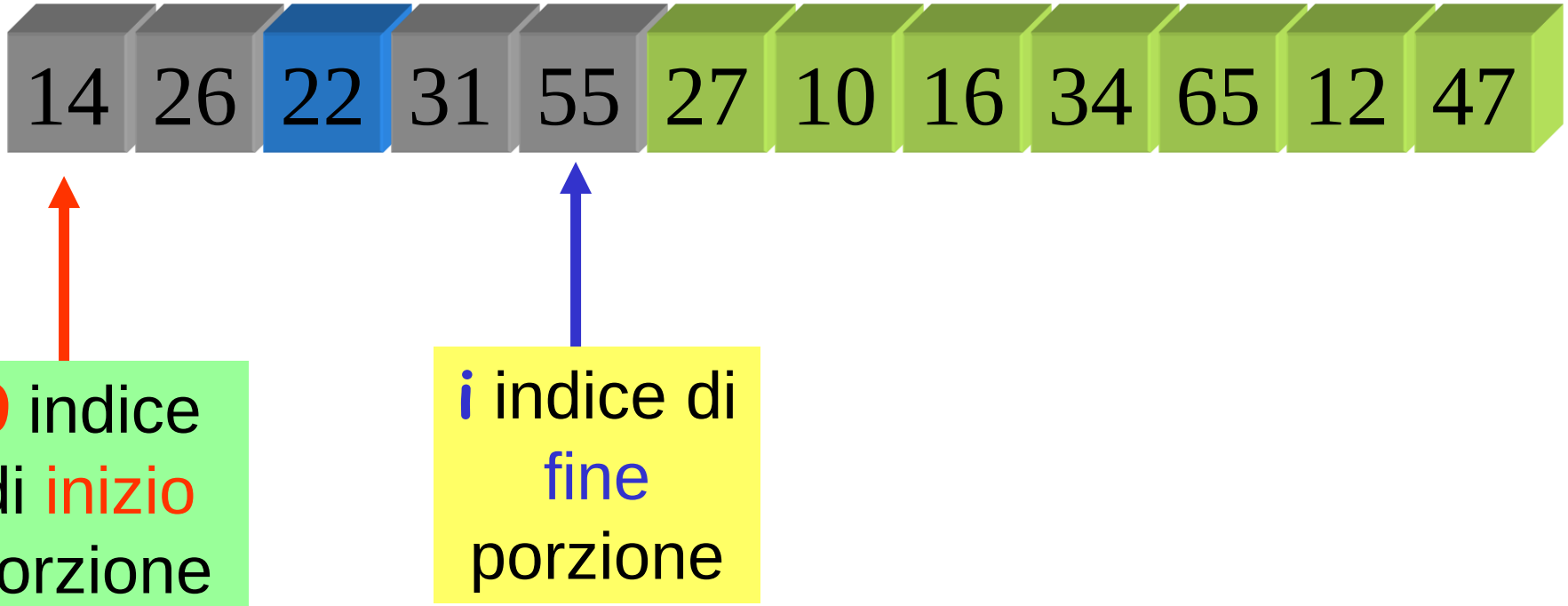
i=4



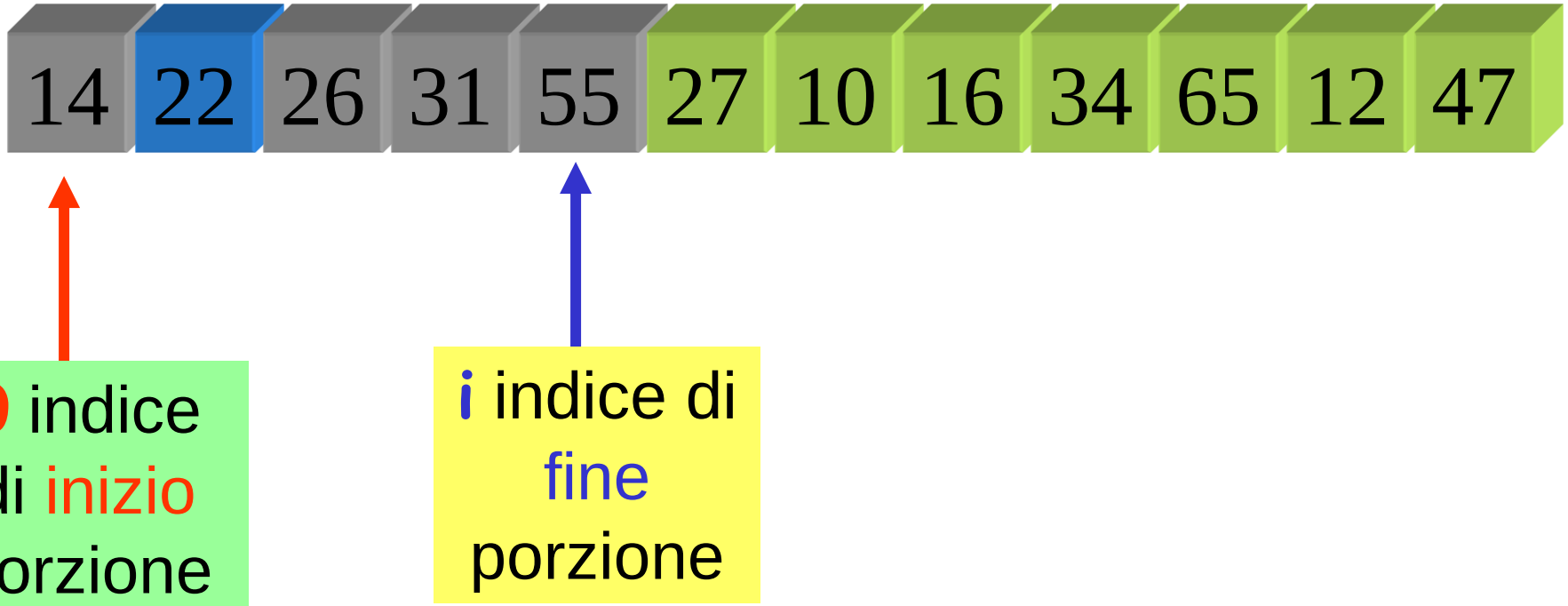

```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```



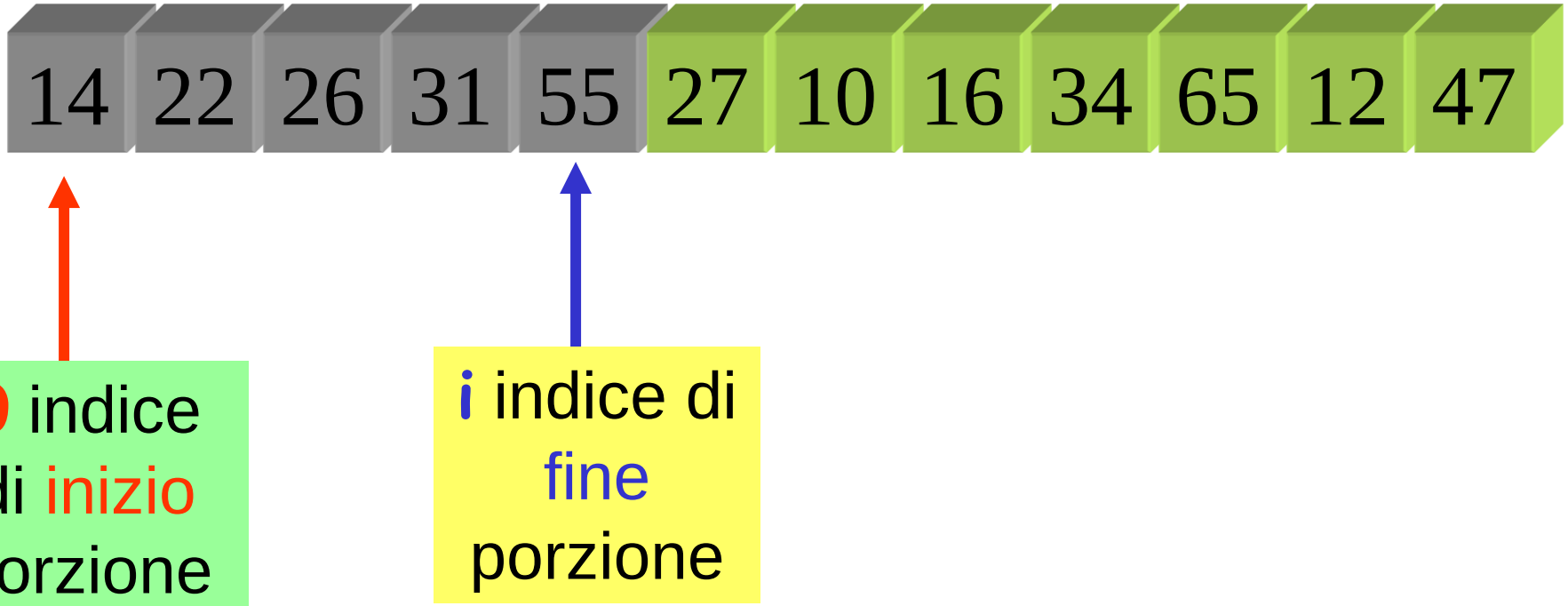

```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```



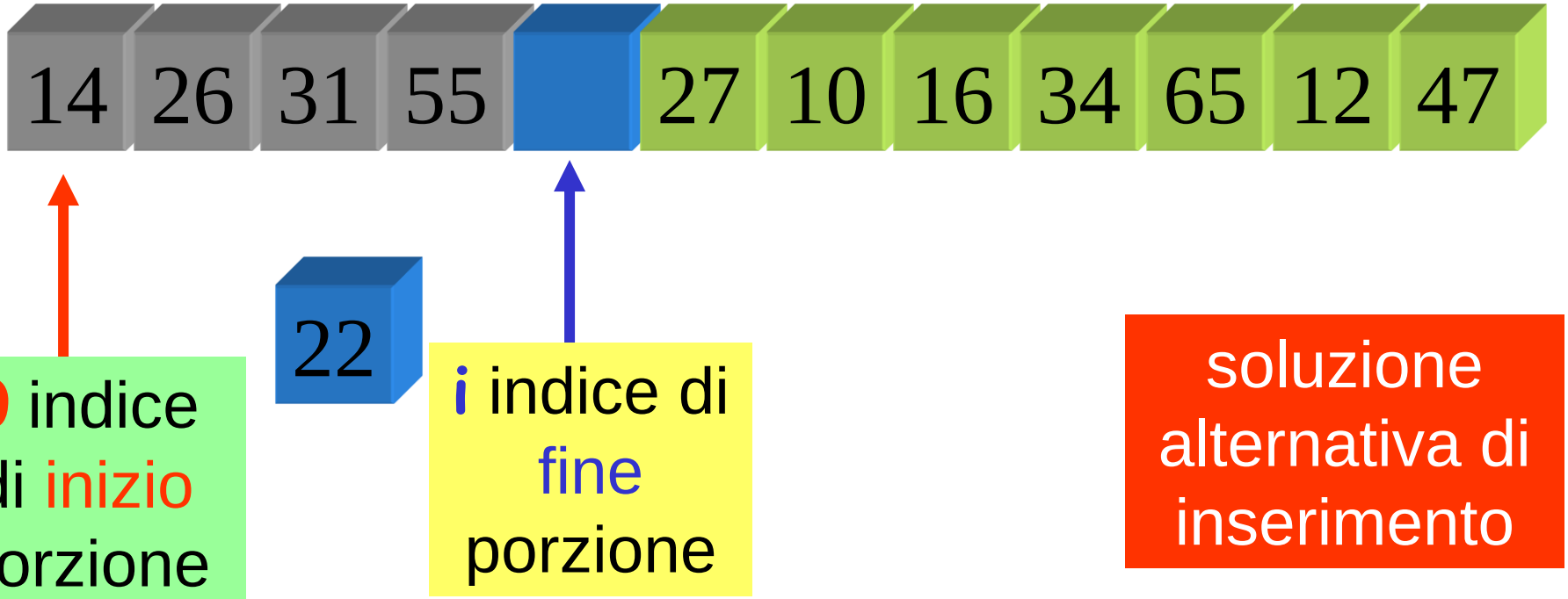

```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```



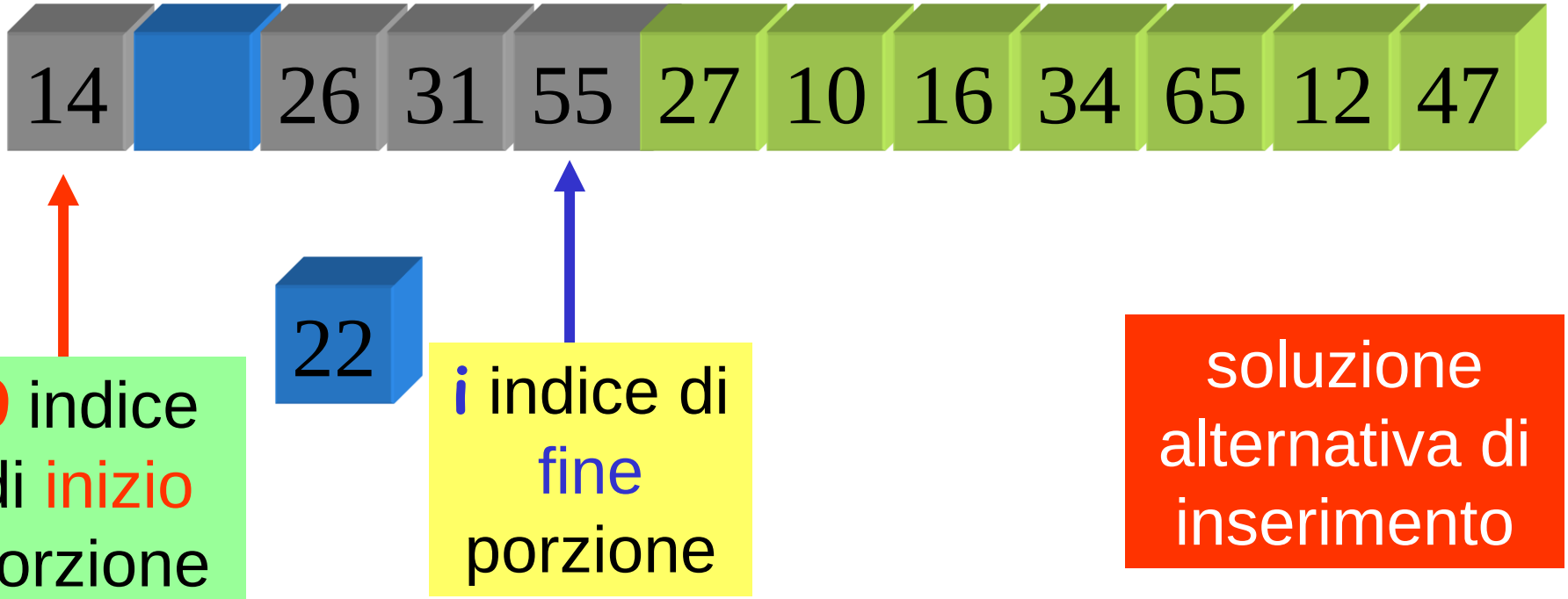

```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```




```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```




```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```




```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```

0 indice di inizio porzione

i indice di fine porzione

soluzione alternativa di inserimento

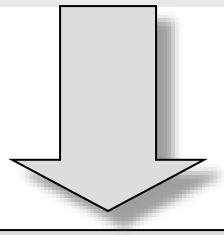
```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```

costrutto ripetitivo: 1 ciclo **for** e 1 ciclo **while** innestati

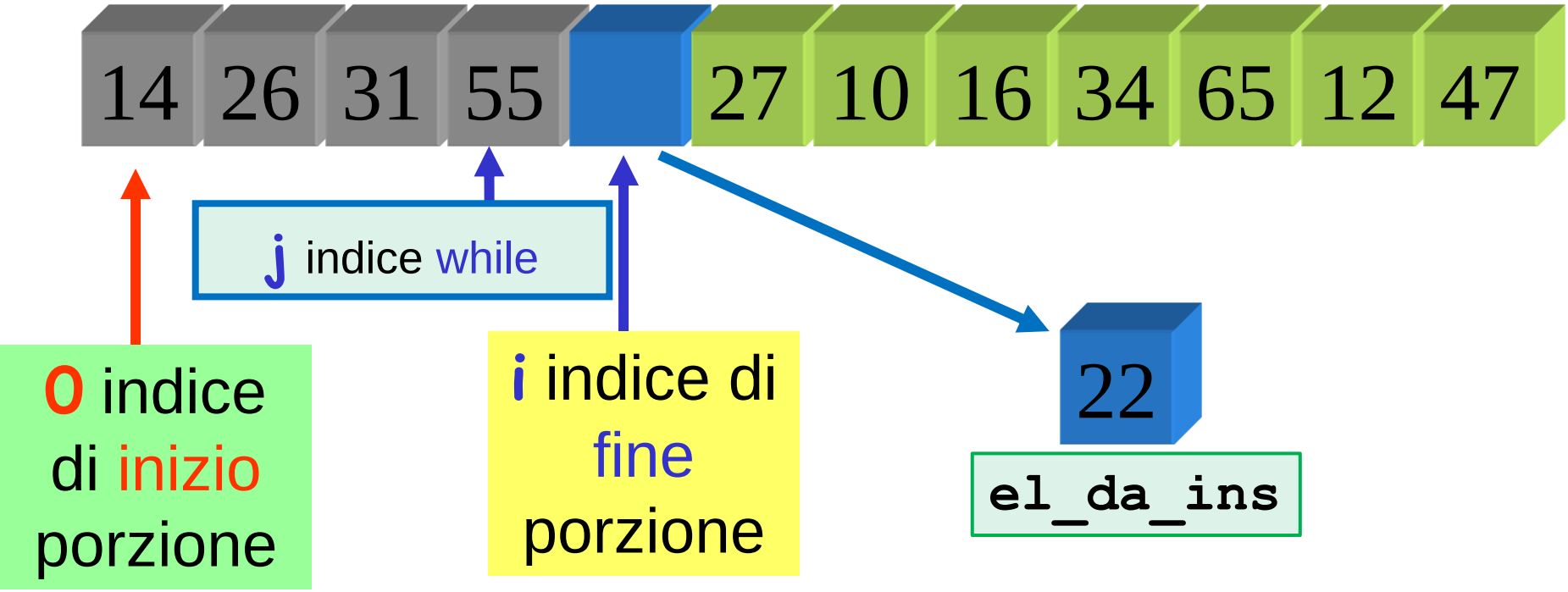
operazione ripetuta (al generico passo **i**, ciclo **for** esterno)**:**

inserire l'**i**-simo elemento nella posizione corretta, per rendere ordinata la porzione (**0..i**)

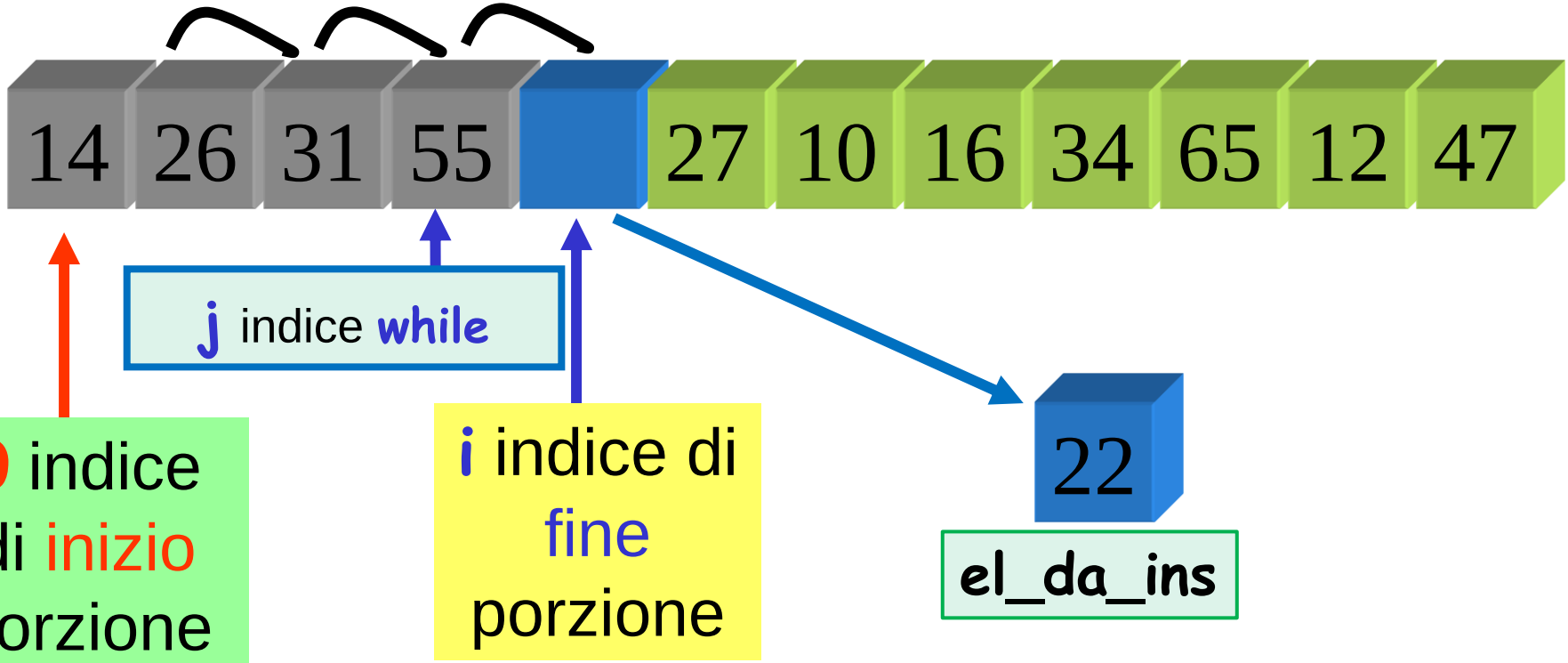
[l'inserimento richiede un ciclo **while**]



algoritmo di ordinamento **per inserimento** (**insertion sort**)




```
for (i=1; i<n; i++) {
    while l' i-simo elemento non è nella posizione corretta
          scambiarlo con l'elemento a sinistra
}
```




```
for (i=1; i<n; i++) {
  while (el_da_ins < a[j]                    ) {
    spostare a destra a[j]
    j--
  }
  inserire el_da_ins
}
```

```
void ord_inser (char a[], int n) {
 int i, j;
 char el_da_ins;
 for (i=1; i < n; i++) {
   el_da_ins = a[i] ;
   j = i-1 ;
   while (j >= 0 && el_da_ins < a[j])
       { a[j+1] = a[j] ;
         j = j-1 ;
       }
   a[j+1] = el_da_ins ;
 }
}
```

```
for (i=1;i<n;i++)
    while …
        al più i volte
```

caso peggiore:

i	confronti	
1	1	+
2	2	+
3	3	+
…		+
n-1	n-1	
	n(n-1)/2	

da ora in poi non usiamo più **in:** , **out:** e **inout:** nell'intestazioni delle **void function**

```
void ord_inser (char a[], int n) {
 int i, j;
 char el_da_ins;
 for (i=1; i < n; i++) {
    el_da_ins = a[i] ;
    j = i-1 ;
    while (j >= 0 && el_da_ins < a[j])
        { a[j+1] = a[j] ;
          j = j-1 ;
        }
    a[j+1] = el_da_ins ;
 }
}
```

n*(n-1)/2
confronti
(tra elementi dell'array)
al più

caso peggiore:

i	scambi
1	1 +
2	2 +
3	3 +
…	+
n-1	n-1
	n(n-1)/2

n*(n-1)/2
scambi [spostamenti]
(tra elementi dell'array)
al più

algoritmo di ordinamento **per inserimento** (**insertion sort**)

$T(n) = n*(n-1)/2$

confronti

$T(n) = n(n-1)/2$

(spostamenti)

(tra elementi dell'array)

al più

$T(n) = O(n^2)$

confronti e scambi

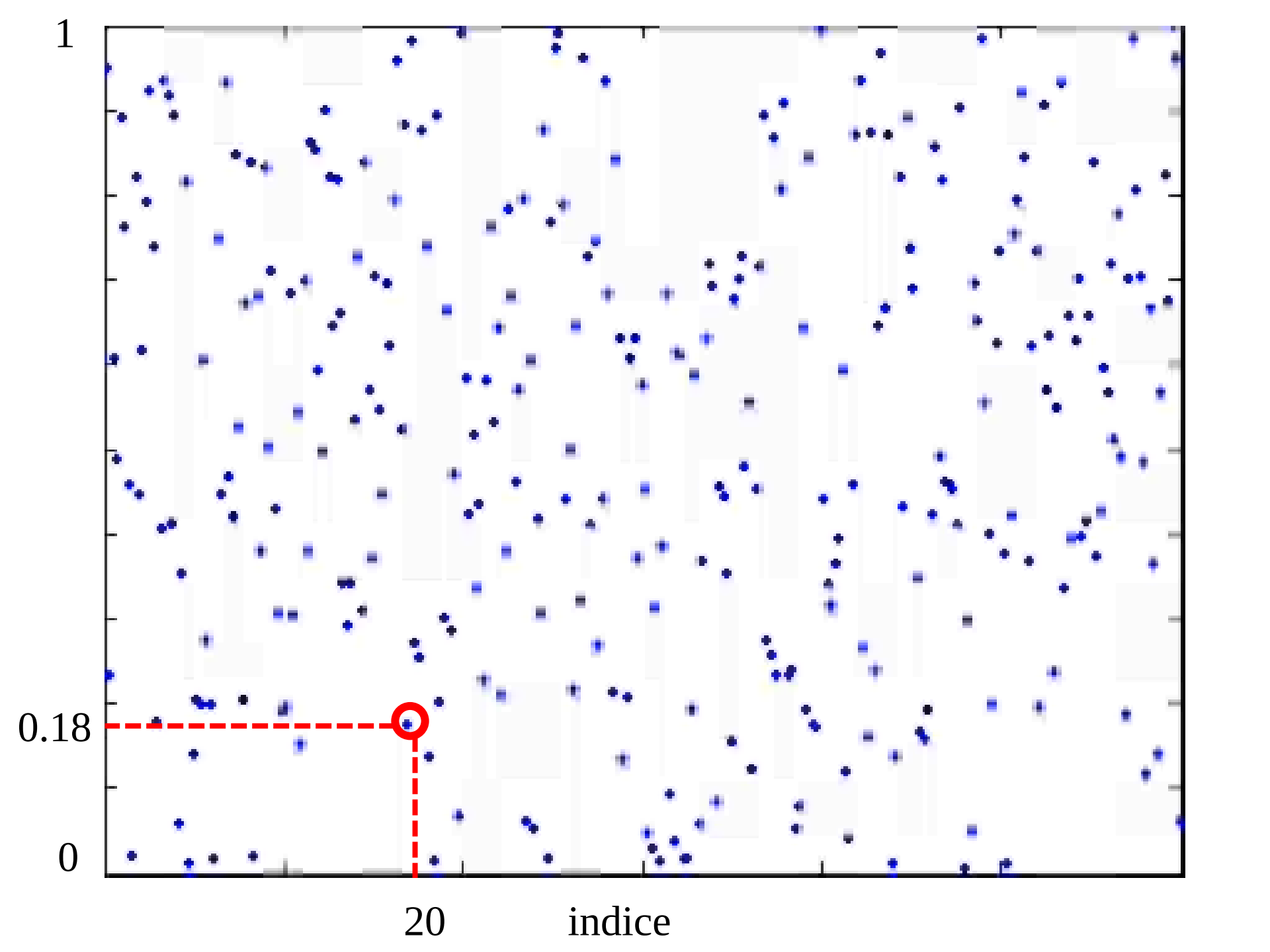
nel caso peggiore

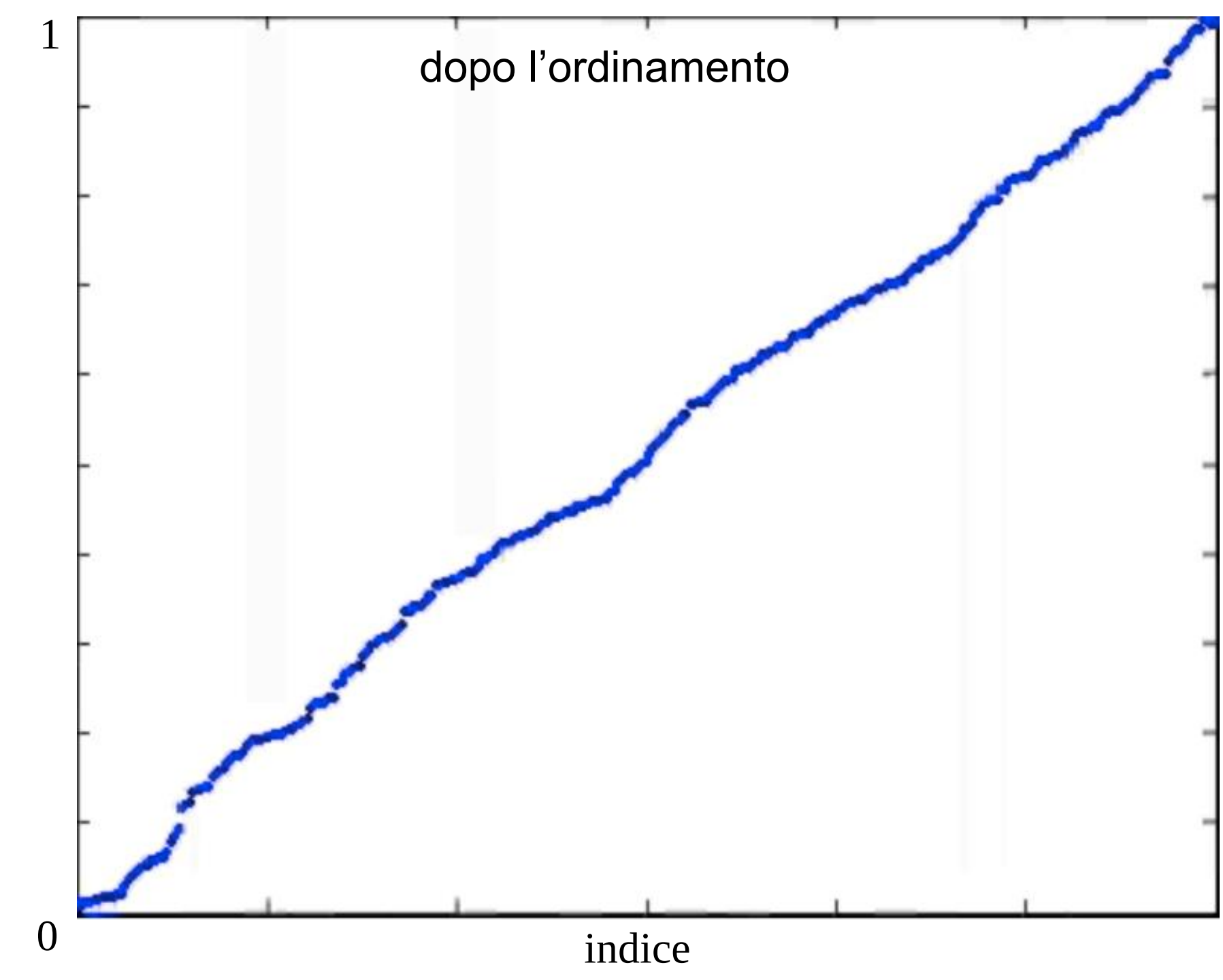
caso peggiore: array ordinato in senso decrescente

caso migliore: array già ordinato

$n-1$ **confronti**

0 **scambi**


1
0.18
0
20
indice


1
dopo l’ordinamento
0
indice

Link video ordinamento per inserimento

per visualizzare il video andare in piattaforma di e-learning e scaricare il file ordinsmov.avi

```
void ord_inser_da_destra (char a[], int n) {
 int i, j;
 char el_da_ins;
 for (i=n-2; i >= 0; i--) {
    el_da_ins = a[i] ;
    j = i+1 ;
    while (j < n && el_da_ins > a[j])
        { a[j-1] = a[j] ;
          j = j+1 ;
        }
    a[j-1] = el_da_ins ;
 }
}
```