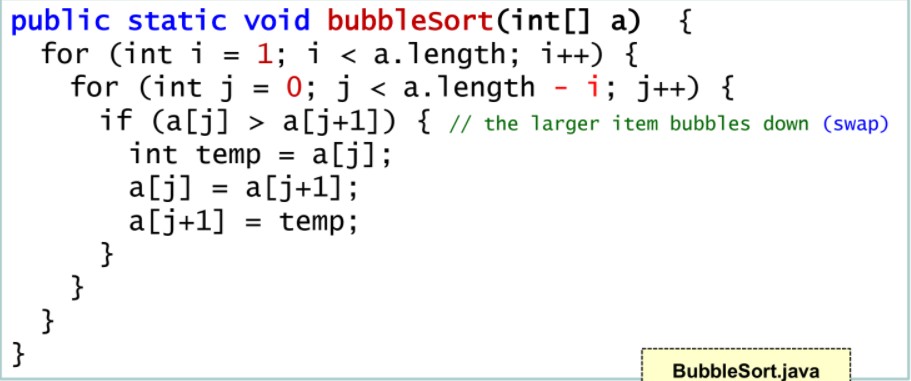
Tên thuật toán : Bubble Sort

Ý tưởng lý thuyết : Bubble Sort là thuật toán sắp xếp đơn giản nhất hoạt động bằng cách hoán đổi nhiều lần các phần tử liền kề nếu chúng không đúng thứ tự.

Cho mảng sau : [ 25, 5 , 1 , 11 ]



a.length = 4

Khi i = 1, i < a.length (4) ( true )

Khi j = 0, j < a.length – 1 (3) ( true ) a [ j ] = 25 > a [ j + 1 ] = 5 ( true ) swap ( a [ j ], a [ j + 1] )

a [ j ] 5, a [ j + 1 ] = 25

Mảng mới : [ 5, 25, 1, 11 ]

Khi j = 1, j < a.length – 1 (3) ( true ) a [ j ] = 25 > a [ j + 1 ] = 1 ( true ) swap ( a [ j ], a [ j + 1] )

a [ j ] 1, a [ j + 1 ] = 25

Mảng mới : [ 5, 1, 25, 11 ]

Khi j = 2, j < a.length – 1 (3) ( true ) a [ j ] = 25 > a [ j + 1 ] = 11 ( true ) swap ( a [ j ], a [ j + 1] )

a [ j ] = 11, a [ j + 1 ] = 25

Mảng mới : [ 5, 1, 11, 25 ]

Khi j = 3, j < a.length – 1 (3) ( false ) ( out loop ) Khi i = 2, i < a.length (4) (true)

Khi j = 0, j < a.length – 1 (3) (true) a [ j ] = 5 > a [ j + 1 ] = 1 (true) swap ( a [ j ], a [ j + 1] )

a [ j ] = 1, a [ j + 1 ] = 5

Mảng mới : [ 1, 5, 11, 25 ]

Khi j = 1, j < a.length – 1 (3) (true) a [ j ] = 5 > a [ j + 1 ] = 11 (false) Khi j = 2, j < a.length – 1 (3) (true) a [ j ] = 11 > a [ j + 1 ] = 25 (true)

Khi j = 3, j < a.length – 1 (3) (false) (out loop) Mảng mới : [ 1, 5, 11, 25 ]

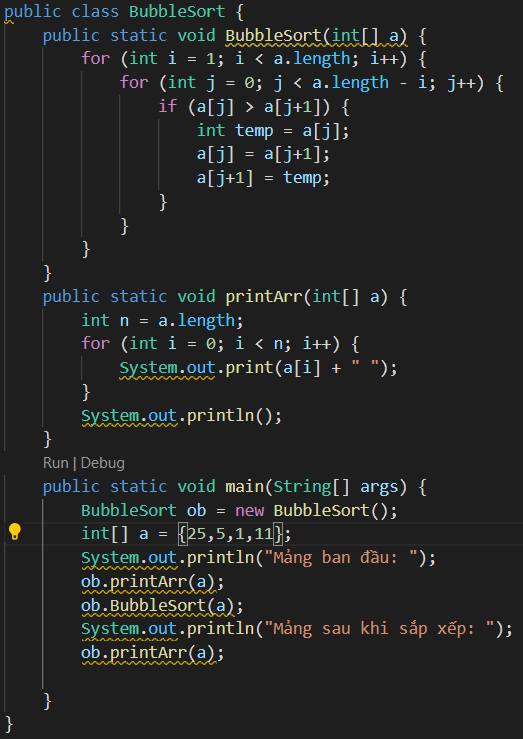
Khi i = 3, i < a.length (4) (true)

Khi j = 0, j < a.length – 1 (3) (true) a [ j ] = 1 > a [ j + 1 ] = 5 (false) Khi j = 1, j < a.length – 1 (3) (true) a [ j ] = 5 > a [ j + 1 ] = 11 (false)

Khi j = 3, j < a.length – 1 (3) (false) (out loop) Khi i = 4, i < a.length (4) (false) (out loop)

Mảng sau khi được sắp xếp : [ 1, 5, 11, 25 ]

Minh họa code demo java :



a [ j ]

a [ j ]



25

5

1

11

Loop 1 : i = 1

j = 0, a [ j ] > a [ j + 1], swap ( a [ j ], a [ j + 1 ] )

a [ j ]

a [ j + 1]



5

25

1

11

j = 1, a [ j ] > a [ j + 1 ] , swap ( a [ j ], a [ j + 1 ] )

a [ j ]

a [ j + 1]



5

1

25

11

j = 2, a [ j ] > a [ j + 1 ] , swap ( a [ j ], a [ j + 1 ] )

5

11

25

1

Loop 2 : i = 2

j = 0, a [ j ] > a [ j + 1], swap ( a [ j ], a [ j + 1 ] )

a [ j ]

a [ j + 1]



5

1

11

25

j = 1, a [ j ] > a [ j + 1 ] ( false )

5



a [ j ]

1

11

25

a [ j + 1]

j = 2, a [ j ] > a [ j + 1] ( false )

a [ j ]

a [ j + 1]



1

5

11

25

Loop 2 : i = 3

j= 0, a [ j ] > a [ j + 1 ] ( false )

a [ j ]

a [ j + 1]



1

5

11

25

j = 1, a [ j ] > a [ j + 1 ] ( false )

a [ j ]

a [ j + 1]



1

5

11

25

j = 2, a [ j ] > a [ j + 1 ] ( false )

a [ j ]

a [ j + 1]



1

5

11

25

Loop 3 : i = 4 > a.length ( 4 ) ( false ) Mảng mới sau khi được sắp xếp

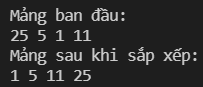
1

11

25

5

Kết quả sau khi chạy code :

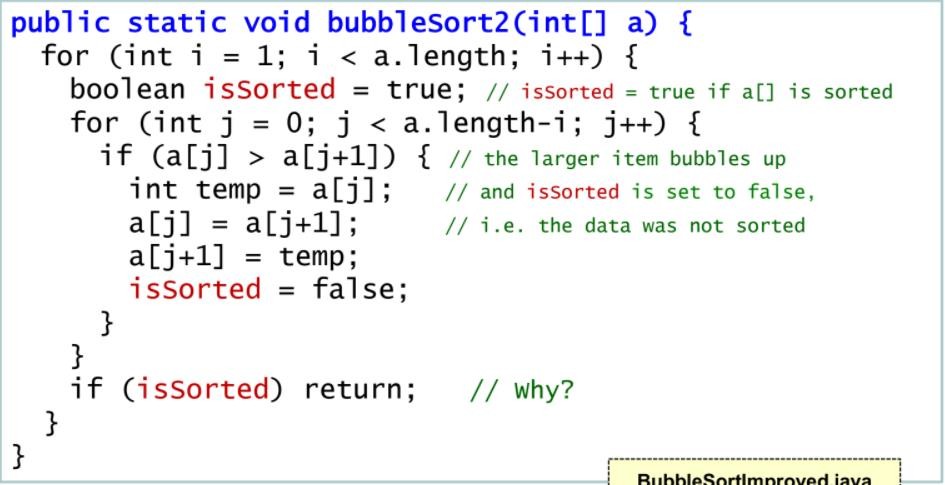


Độ phức tạp của thuật toán :

Worse case : O(n^2), trường hợp xấu nhất xảy ra khi mảng được sắp xếp ngược lại Average Case Time Complexity : O(n^2)

Best case : O(n), trường hợp tốt nhất xảy ra khi mảng đã được sắp xếp

Thuật toán Improved Bubble Sort



Mảng ban đầu : [ 25, 5, 1, 11 ]

a.length = 4

Loop 1 : i = 1, i < a.length ( 1 < 4 ) ( true) isSorted = true;

j = 0, j < a.length - i ( 0 < 4 – 1 = 3) (true) a [ j ] > a [ j + 1 ] (25 > 5 ) ( true )

swap ( a [ j ] , a [ j + 1 ] ; isSorted = false;

Mảng mới : [ 5, 25, 1, 11 ]

j = 1, j < a.length - i ( 1 < 4 – 1 = 3) (true) a [ j ] > a [ j + 1 ] (25 > 1 ) ( true )

swap ( a [ j ] , a [ j + 1 ] ; isSorted = false;

Mảng mới : [ 5, 1, 25, 11 ]

j = 2, j < a.length - i ( 1 < 4 – 1 = 3) (true)

a [ j ] > a [ j + 1 ] (25 > 11 ) ( true )

swap ( a [ j ] , a [ j + 1 ] ; isSorted = false;

Mảng mới : [ 5, 1, 11, 25 ]

j = 3, j < a.length - i ( 1 < 4 – 1 = 3) (false) Out loop;

Mảng sau khi xong vòng lặp đầu : [ 5, 1, 11, 25 ] Loop 2 : i = 2, i < a.length ( 2 < 4 ) ( true) isSorted = true;

j = 0, j < a.length - i ( 0 < 4 – 2 = 2) (true) a [ j ] > a [ j + 1 ] (5 > 1 ) ( true )

swap ( a [ j ] , a [ j + 1 ] ; isSorted = false;

Mảng mới : [ 1, 5, 11, 25 ]

j = 1, j < a.length - i ( 1 < 4 – 2 = 2) (true) a [ j ] > a [ j + 1 ] (25 > 1 ) ( false )

j = 2, j < a.length - i ( 1 < 4 – 2 = 2) (false) Out loop

Mảng mới : [ 1, 5, 11, 25 ]

Mảng sau khi xong vòng lặp thứ hai : [ 1, 5, 11, 25 ] Loop 3 : i = 3, i < a.length ( 3 < 4 ) ( true)

isSorted = true;

j = 0, j < a.length - i ( 0 < 4 – 3 = 1) (true) a [ j ] > a [ j + 1 ] (1 > 5 ) ( false )

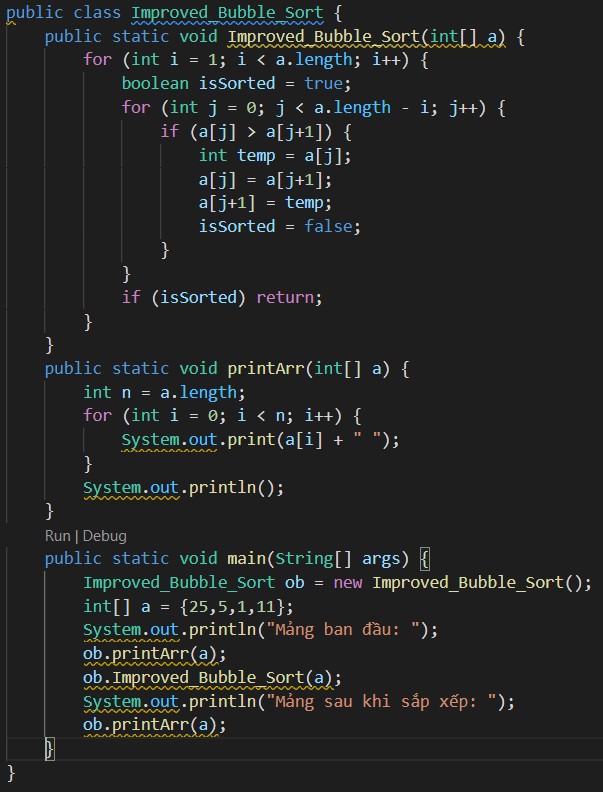
Out loop;

Mảng sau khi xong vòng lặp thứ ba : [ 1, 5, 11, 25 ] Loop 4 : i = 4, i < a.length ( 4 < 4 ) ( false)

Out loop; isSorted = true;

Trả về mảng đã được sắp xếp thành công : [ 1, 5, 11, 25 ]

Code demo java :



isSorted = true

a [ j ]

a [ j ]



25

5

1

11

Loop 1 : i = 1

isSorted = true

j = 0, a [ j ] > a [ j + 1], swap ( a [ j ], a [ j + 1 ] )

isSorted = false

a [ j + 1]

a [ j ]



5

25

1

11

j = 1, a [ j ] > a [ j + 1 ] , swap ( a [ j ], a [ j + 1 ] )

a [ j ]

isSorted = false

a [ j + 1]



5

1

25

11

j = 2, a [ j ] > a [ j + 1 ] , swap ( a [ j ], a [ j + 1 ] )

5

11

25

1

Loop 2 : i = 2

isSorted = true

j = 0, a [ j ] > a [ j + 1], swap ( a [ j ], a [ j + 1 ] )

isSorted = false

a [ j ]

a [ j + 1]



5

1

11

25

j = 1, a [ j ] > a [ j + 1 ] ( false )

1

5

11

25



a [ j ]

a [ j + 1]

isSorted = false

j = 2, a [ j ] > a [ j + 1] ( false )

isSorted = false

a [ j + 1]

a [ j ]



1

5

11

25

Loop 2 : i = 3

isSorted = true

j= 0, a [ j ] > a [ j + 1 ] ( false )

a [ j ]

a [ j + 1]



1

5

11

25

j = 1, a [ j ] > a [ j + 1 ] ( false )

a [ j ]

a [ j + 1]



1

5

11

25

j = 2, a [ j ] > a [ j + 1 ] ( false )

a [ j ]

a [ j + 1]



1

5

11

25

Loop 3 : i = 4 > a.length ( 4 ) ( false )

isSorted = true

Mảng mới sau khi được sắp xếp

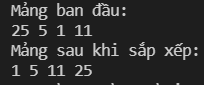
1

11

25

5

Kết quả :



Độ phức tạp của thuật toán :

Worse case : O(n^2), trường hợp xấu nhất xảy ra khi mảng được sắp xếp ngược lại Average Case Time Complexity : O(n^2)

Best case : O(n), trường hợp tốt nhất xảy ra khi mảng đã được sắp xếp