Sau đây là lưu đồ tổng thể của thuật toán selection sort

Để hiểu rõ cái lưu đồ này thì tôi sẽ cho luôn ví dụ là n = 3 5 8 1 4

Tương ứng là n có 5 phần tử từ n0 cho đến n4

Chúng ta sẽ đặt bức tường bắt đầu từ tận cùng bên trái của dãy số và bắt đầu so sánh từng phần tử trong bức tường tính từ trái sang phải bắt đầu từ n0

Nếu điều kiện so sánh trả về là true nghĩa là trong bức tường còn có từ 2 phần tử trở lên thì nó sẽ tìm số nhỏ nhất từ bức tường và sẽ hoán vị số đó cho cái phần tử đầu tiên tính từ bức tường.

Ví dụ như ở ví dụ trên thì n0 đang bằng 3 nhưng n3 lại bằng 1 cho nên nó hoán vị vị trí của 3 cho 1 nghĩa là n0 lúc này có giá trị bằng 1 và n3 có giá trị bằng 3

Sau đó bức tường sẽ dịch chuyển sang bên phải chính xác 1 đơn vị phần tử và quay trở lại cái vòng lặp ban đầu lại bắt đầu compare cho đến khi bức tường dịch chuyển đến sau phần tử thứ n i-1hay ở trong ví dụ trên thì nó dịch chuyển đến trước phần tử n4 thì lúc này điều kiện trả về sẽ là false

Bởi vì trong bức tường chỉ còn một phần tử và điều kiện compare là bắt buộc có 2 phần tử trở lên. Cho nên lúc này khi giá trị trả về là false thì nó sẽ đưa ra cái mảng đã được sắp xếp theo thứ tự từ nhỏ cho đến lớn và kết thúc vòng lặp. Như ở ví dụ trên thì mảng mới sẽ là 1 3 4 5 8.

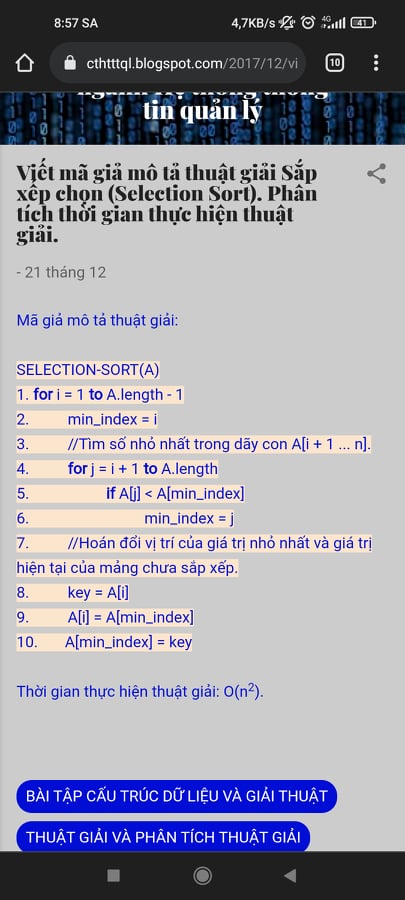
Vậy ví dụ như nếu mảng của chúng ta có 2 phần tử trở lên trùng nhau thì sao? Thì nó vẫn sẽ sắp xếp như bình thường chỉ có điều các phần tử trùng nhau sẽ được xếp cạnh nhau.

//Thì thuật toán này có những ưu điểm đó là đơn giản, dễ thực hiện và có số lần hoán vị ít nhưng vì đơn giản nên cũng có những nhược điểm đó là chỉ được áp dụng trong các trường hợp có số lượng phần tử cần so sánh ít, một hạn chế nữa là nó không nhận biết được mảng đã sắp xếp. Cho nên nó được ưa dùng cho những chương trình đơn giản.

Đó là toàn bộ nội dung về thuật toán selection sort. Cảm ơn cô và các bạn đã lắng nghe nhóm 1 chúng em//

Ưu và nhược điểm

* Số lần so sánh trong trường hợp tốt nhất là n(n-1)/2
* Số lần so sánh trong trường hợp xấu nhất là 3n(n-1)/2



1.

The following is the overall flowchart of the selection sort algorithm

To understand this flowchart, I will give an example of n = 3 5 8 1 4

Correspondingly, n has 5 elements from n0 to n4

2.

We will place the wall starting at the left end of the sequence and start comparing each element in the wall from left to right starting at n0.

3 + 4. If the comparison condition returns true, it means that there are 2 or more elements in the wall, then it will find the smallest number from the wall and will transpose that number to the first element from the wall.

In the above example, n0 is equal to 3 but n3 is equal to 1, so it transposes the position of 3 for 1, meaning that n0 now has the value of 1 and n3 has the value of 3.

5 + 6.The wall will then shift to the right exactly 1 element and go back to the original loop that starts comparing again until the wall moves after the n-1 element or in the example above, it moves before the element n4, then the return condition will be false

7.Because there is only one element in the wall and the compare condition is required to have 2 or more elements. So now when the return value is false, it will output the sorted array from smallest to largest and end the loop. As in the above example, the new array will be 1 3 4 5 8.

So for example, what if our array has 2 or more identical elements? Then it will still sort as usual, only the duplicate elements will be placed next to each other.