Bài 8.2: Thuật toán bubble sort

- ✓ Mục đích sử dụng
- ✓ Sắp xếp tăng dần
- ✓ Sắp xếp giảm dần
- ✓ Bubble sort tối ưu
- ✓ Bài tập thực hành

Mục đích sử dụng

- Sắp xếp là đặt các phần tử của một tập hợp theo thứ tự của một hoặc một số tiêu chí nào đó.
- Đây là chức năng thiết yếu trong hầu hết các ứng dụng, phần mềm.
- Giúp tối ưu chương trình, chuẩn hóa dữ liệu đầu vào, tạo đầu ra dễ đọc hiểu với con người.
- Ví dụ cần sắp xếp danh sách nhân viên theo lương giảm dần.
- Ví dụ cần sắp xếp sản phẩm của trang bán hàng theo thứ tự giá giảm dần…
- Thuật toán sắp xếp nổi bọt có độ phức tạp O(n^2)

Sắp xếp tăng dần

- Sau đây là mã giả của thuật toán sắp xếp nổi bọt theo thứ tự tăng dần của các phần tử:

```
void bubbleSort(arr[]) { // arr-mảng 1 chiều, n: số phần tử của mảng
    n = arr.length // lấy số phần tử của mảng
        for (i from 0 to n - 2) { // cho i chạy từ 0 đến n-2
            for (j from n - 1 to i + 1) { // cho j chạy từ n-1 đến i+1
                if (arr[j - 1] > arr[j]) { // đổi chỗ hai phần tử
                    tmp = arr[j - 1];
                    arr[j - 1] = arr[j];
                    arr[j] = tmp;
                } // end if
            } // end inner for loop
        } // end outer for loop
} // end function
```

Sắp xếp tăng dần

- Triển khai của thuật toán sắp xếp bubble sort tăng dần:

```
public static <T extends Comparable<T>> void bubbleSort(T[] arr, int n) {
    for (int i = 0; i <= n - 2; i++) {
        for (int j = n - 1; j >= i + 1; j--) {
            if (arr[j - 1].compareTo(arr[j]) > 0) {
                T tmp = arr[j - 1];
                arr[j - 1] = arr[j];
                arr[j] = tmp;
            }
        }
    }
}
```

Sắp xếp giảm dần

- Mã giả của thuật toán sắp xếp bubble sort giảm dần:

```
void bubbleSort(arr[]) { // arr-mảng 1 chiều, n: số phần tử của mảng
    n = arr.length // lấy số phần tử của mảng
        for (i from 0 to n - 2) { // cho i chạy từ 0 đến n-2
            for (j from n - 1 to i + 1) { // cho j chạy từ n-1 đến i+1
                if (arr[j - 1] < arr[j]) { // đổi chỗ hai phần tử
                    tmp = arr[j - 1];
                    arr[j - 1] = arr[j];
                    arr[j] = tmp;
                } // end if
            } // end inner for loop
        } // end outer for loop
} // end function
```

Sắp xếp giảm dần

- Mã thật của thuật toán sắp xếp bubble sort giảm dần:

```
public static <T extends Comparable<T>> void bubbleSortDESC(T[] arr, int n) {
    for (int i = 0; i <= n - 2; i++) {
        for (int j = n - 1; j >= i + 1; j--) {
            if (arr[j - 1].compareTo(arr[j]) < 0) {
                T tmp = arr[j - 1];
                arr[j - 1] = arr[j];
                arr[j] = tmp;
            }
        }
    }
}
```

Bubble sort tối ưu

- Trong trường hợp nếu lần duyệt nào đó với một biến chạy i cố định mà vòng lặp bên trong với biến chạy j kết thúc, không xảy ra việc đổi chỗ cặp phần tử nào tức là lúc này tập hợp đã được sắp xếp đúng thứ tự.
- Do đó ta có thể dùng một biến đánh dấu để kiểm soát việc lặp của trường hợp này.
- Nếu vòng lặp thứ i nào đó kết thúc mà biến đánh dấu cho thấy không xảy ra việc đổi chỗ của bất kì cặp phần tử nào, ta break khỏi vòng lặp, kết thúc việc sắp xếp.
- Kết quả ta nhận được mảng đã sắp xếp.

Bubble sort tối ưu

- Mã giả của thuật toán sắp xếp nổi bọt tối ưu:

```
void bubbleSortOpt(arr[]) {              // thuật toán tối ưu
    n = arr.length;
    isSwapped = true
    i = n - 1
    while (i > 0) {
          isSwapped = false
          for (j from 0 to i - 1) {
                if (arr[j] > arr[j + 1]) {
                      swap(arr[j], arr[j + 1])
                      isSwapped = true
                }
          }
          if (isSwapped == false) {      // nếu không xảy ra sự đổi chỗ
                break                    // kết thúc việc sắp xếp
          }
          else {                         // ngược lại
                i--                      // tiếp tục sắp xếp
          }
    }
}
```

Bubble sort tối ưu

- Mã thật của thuật toán sắp xếp nổi bọt tối ưu:

```
public static <T extends Comparable<T>> void bubbleSortOpt(T[] arr, int n) {
    boolean isSwapped;
    int i = n - 1;
    while (i > 0) {
        isSwapped = false;
        for (int j = 0; j <= i - 1; j++) {
            if (arr[j].compareTo(arr[j + 1]) > 0) {
                T tmp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = tmp;
                isSwapped = true;
            }
        }
        if (!isSwapped) {
            break;
        } else {
            i--;
        }
    }
}
```

Nội dung tiếp theo

Thuật toán sắp xếp chọn – Selection sort