Bài 9.5: Thuật toán sắp xếp của Shell

- ✓ Tổng quan về thuật toán
- ✓ Thuật toán sắp xếp shell sort
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Tổng quan về thuật toán

- Là thuật toán cải tiến của thuật toán sắp xếp chèn, được phát minh bởi Donald Shell vào năm 1959.
- Thuật toán cho phép tránh việc phải dịch chuyển các phần tử ở khoảng cách xa.
- Việc sắp xếp bắt đầu bằng cách tráo đổi các cặp phần tử ở khoảng cách xa nhất trước, sau đó giảm dần khoảng cách, lặp lại cho đến khi khoảng cách bằng 1.
- Mức độ tối ưu của thuật toán phụ thuộc vào chuỗi khoảng cách được chọn để so sánh và tráo đổi các cặp phần tử.
- Với nhiều biến thể của thuật toán này, việc xác định độ phức tạp thuật toán còn là vấn đề bỏ ngỏ.

Tổng quan về thuật toán

- Bài này ta sẽ triển khai thuật toán shell sort với công thức tính khoảng cách Knuth: d = [(3^k-1)/2] với k nguyên nằm trong đoạn [1, N/3].
- Độ phức tạp của thuật toán trong trường hợp này là O(n^1.5).

Mã giả của thuật toán

➢ Mã giả của thuật toán shell sort:

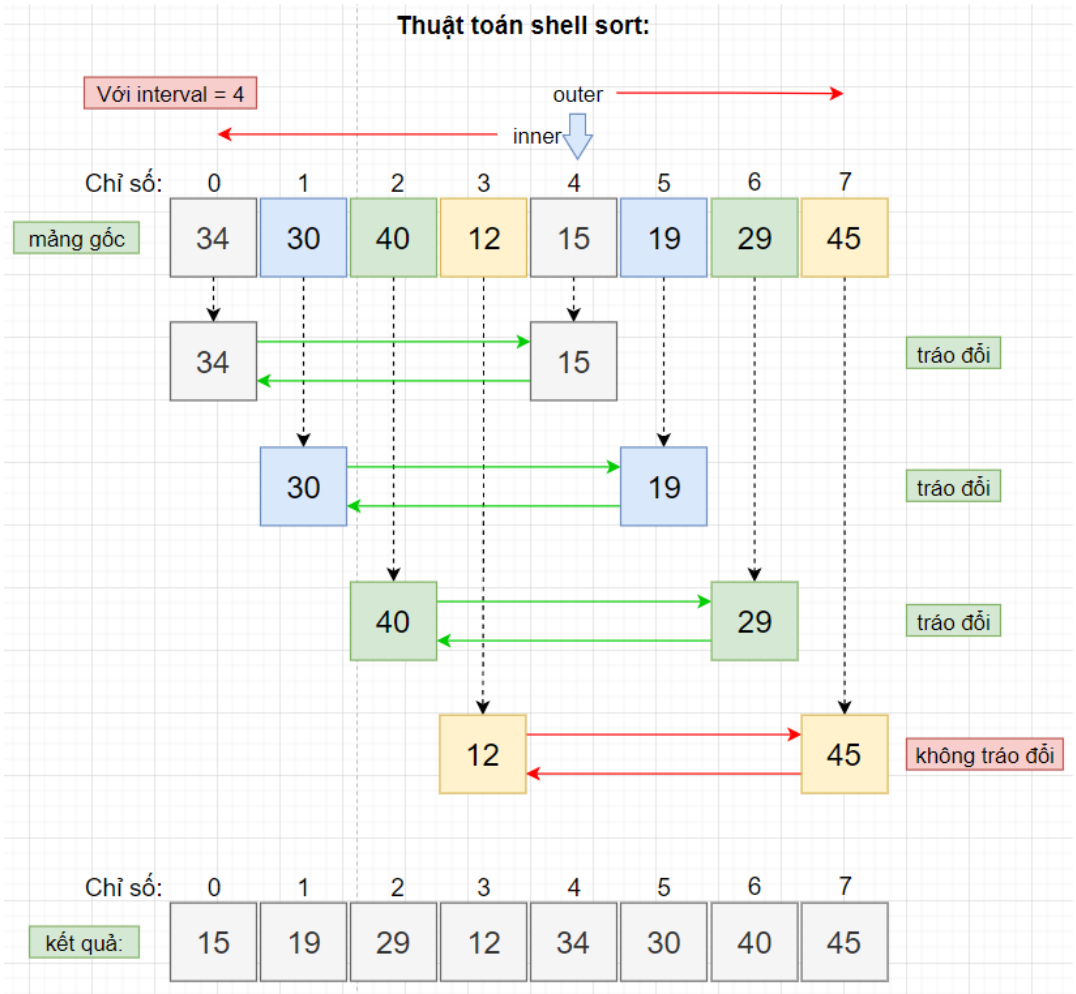
```
// thuật toán sắp xếp shell sort
// arr: mảng đầu vào
// n: số phần tử mảng
function shellSort(arr[], n):
    interval = 1 // khoảng cách giữa 2 phần tử trong mảng
    while(interval < n / 3): // tìm khoảng cách max
        interval = interval * 3 + 1
    while(interval > 0):
        for(outer từ interval tới n - 1):
            target = arr[outer] // phần tử đang xét
            inner = outer;
            while(inner > interval - 1 && arr[inner - interval] >= target):
                arr[inner] = arr[inner - interval] // tráo đổi phần tử
                inner = inner - interval // nhảy sang vị trí mới
            arr[inner] = target // cập nhật phần tử tại vị trí đầu bên trái
        interval = (interval - 1) / 3 // cập nhật khoảng cách
```

Mã thật của thuật toán

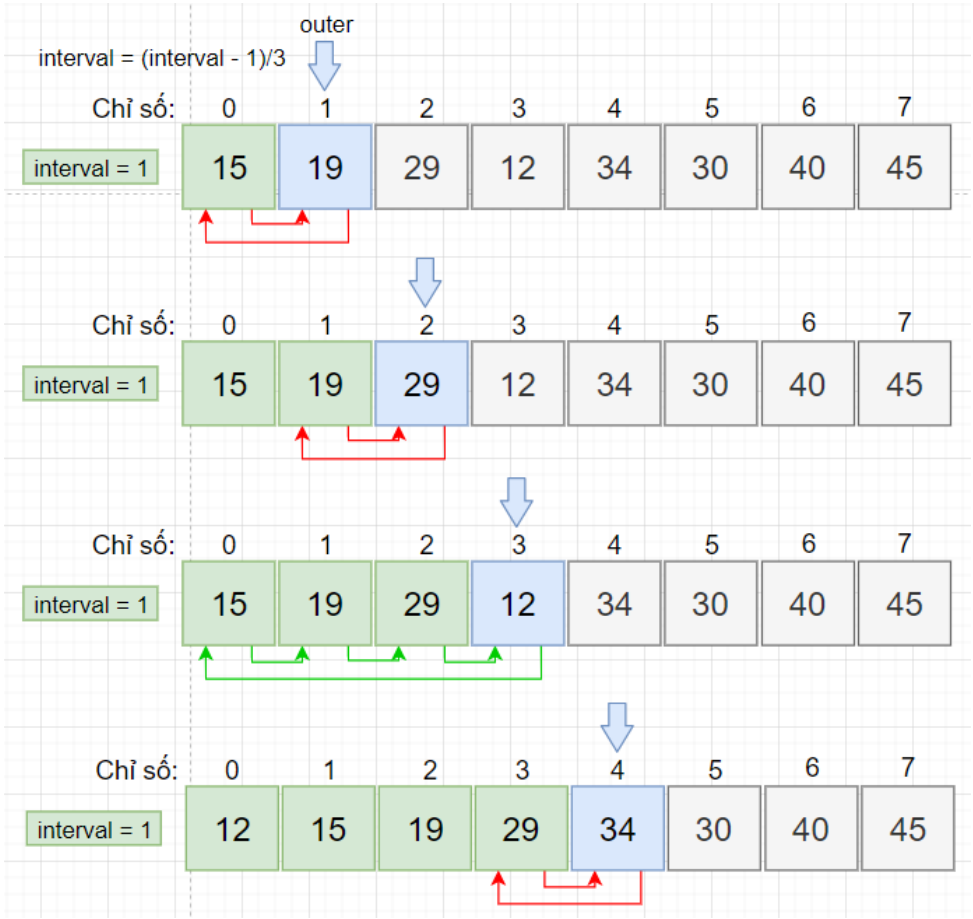
➢ Mã thật của thuật toán shell sort:

```
// thuật toán sắp xếp shell
template<class T> void shellSort(T* arr, int size) {
    int interval = 1;
    while (interval < size / 3) {
        interval = interval * 3 + 1;
    }
    while (interval > 0) {
        for (int outer = interval; outer < size; outer++)
        {
            T target = arr[outer];
            int inner = outer;
            while (inner > interval - 1 && arr[inner - interval] >= target) {
                arr[inner] = arr[inner - interval];
                inner = inner - interval;
            }
            arr[inner] = target;
        }
        interval = (interval - 1) / 3;
    }
}
```

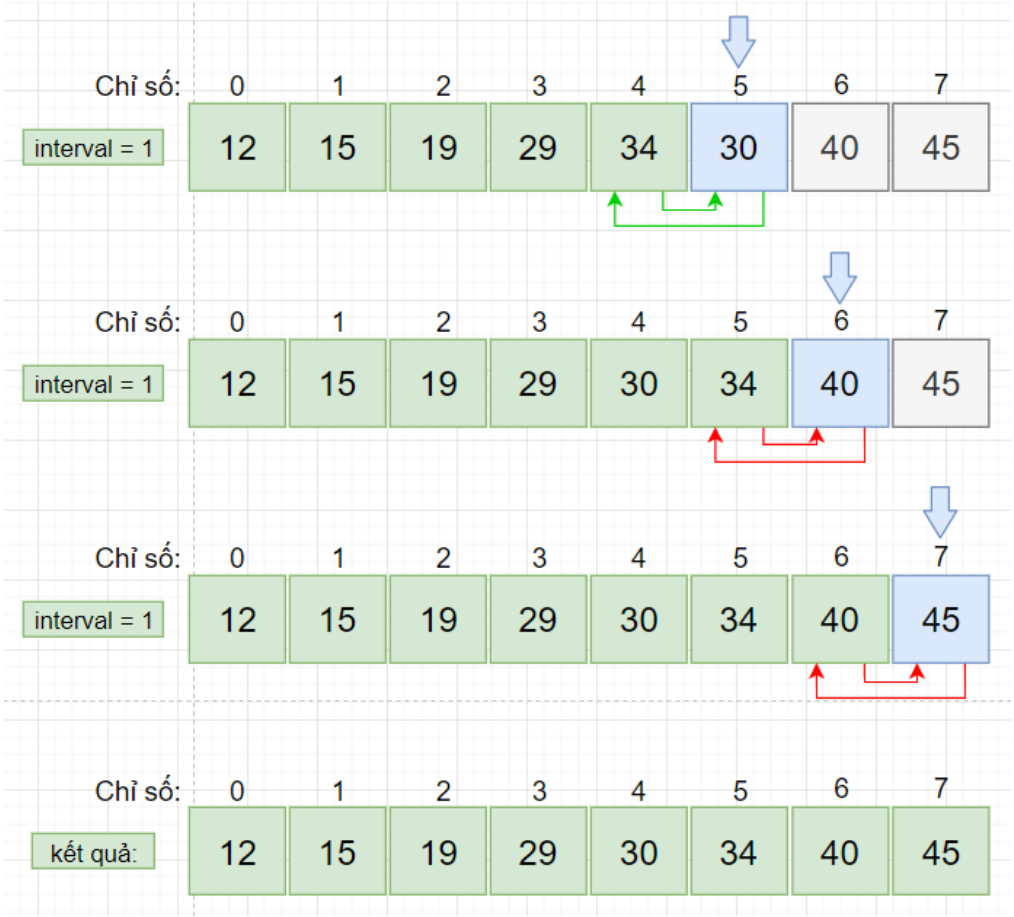
Minh họa thuật toán



Minh họa thuật toán



Minh họa thuật toán



Nội dung tiếp theo

Thuật toán sắp xếp merge sort