**ASSIGNMENT 01: Giải thích kết quả thực nghiệm**

Array random: [9.0, 3.0, 5.0, 6.0, 1.0, 2.0, 4.0]

Array sắp xếp (thấp 🡪 cao): [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 9.0]

Array sắp xếp ngược (cao 🡪 thấp): [9.0, 6.0, 5.0, 4.0, 3.0, 2.0, 1.0]

1. **Bubble sort (Sắp xếp nổi bọt):**

* Độ phức tạp tổng quát: 0(n2)
* Bubble sort có 2 vòng lặp, vòng lặp trong có độ phức tạp 0(n)
* Worst case:
* Trường hợp xấu nhất,array được sắp xếp ngược, vòng ngoài chạy 0(n) times
* Độ phức tạp của thuật toán: 0(n\*n) = 0(n2)
* Best case:
* Trường hợp tốt nhất, array đã được sắp xếp , thuật toán có độ phức tạp 0(n)
* Average case:
* Trường hợp random array, độ phức tạp thuật toán: 0(n2)
* Ưu điểm:
* Bubble sort chỉ cần rất ít memory
* Được sử dụng với những data bé, chỉ cần vài dòng code
* Với Best case, bubble sort có độ phức tạp là 0(n) trong khi các thuật toán khác có độ phức tạp lớn hơn
* Nhược điểm:
* Không hiệu quả với những dữ liệu lớn

1. **Selection sort:**

* Bởi vì thuật toán chúng ta yêu cầu tìm phần tử nhỏ nhất, nên phải scan cả array để tìm phần tử bé nhất đó
* Selection sort có 2 vòng lặp, mỗi vòng có độ phức tạp 0(n2) 🡪 Độ phức tạp tổng quát: 0(n2)
* Worst case:
* Trường hợp xấu nhất, array đã được sắp xếp ngược, thuật toán có độ phức tạp: 0(n2)
* Best case:
* Array được sắp xếp đúng thứ tự, độ phức tạp: 0(n2)
* Average case:
* Array random, độ phức tạp: 0(n2)
* Ưu điểm:
* Dùng cho các dữ liệu nhỏ, random array
* Nhược điểm:
* Không hiệu quả với các dữ liệu lớn

1. **Insertion sort:**

* Worst case:
* Trường hợp xấu nhất, array đã được sắp xếp ngược, thuật toán có độ phức tạp: 0(n2)
* Best case:
* Array được sắp xếp đúng thứ tự, độ phức tạp: 0(n)
* Average case:
* Array random, độ phức tạp: 0(n2)
* Ưu điểm:
* Dùng cho các dữ liệu nhỏ, random array
* Đơn giản
* Nhược điểm:
* Không hiệu quả với các dữ liệu lớn